UGAN: Underwater Image Restoration using Generative Adversarial Networks

Cameron Fabbri

Information Directorate

Air Force Research Laboratory

Rome, NY, USA.

cameron.fabbri@us.af.mil

Md Jahidul Islam

University of Minnesota- Twin Cities

MN, USA.

islam034@umn.edu

Junaed Sattar

University of Minnesota- Twin Cities

MN, USA.

junaed@umn.edu

Abstract—Autonomous underwater robots often rely on visual input for decision making due to its non-intrusive and passive nature. However, due to many factors such as light refraction, particles in the water, and color distortion, images are often times very noisy. This paper propose a method using Generative Adversarial Networks (GANs) to denoise underwater images, and show that these images provide both increased accuracy for an underwater tracking algorithm, as well as a more visually appealing image. Furthermore, we show how recently proposed methods are able to generate a dataset for the purpose of underwater image reconstruction.

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. Vision is a commonly used sensor in autonomous underwater robots due to its non-intrusive, passive, and energy effecient nature. The monitoring of coral reefs [1], deep ocean exploration [2], and mapping of the seabed are all tasks suitable for autonomous robots because they provide safety by taking the risk instead of a human. Despite the advantages vision provides, many underwater environments can be quite noisy due to light refraction and particles present in the water. Because red wavelengths are quickly absorbed by water, images tend to have a green or blue hue to them. As you go deeper, this worsens as more and more red wavelengths are being absorbed. This extremely non-linear distortion has many factors such as the amount of light present (overcast vs sunny or depth), 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.

As color and illumination begin to change with the depth, vision based algorithms must 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, as well as the high amount of noise introduced, many algorithms may perform poorly in these different domains. Figure 1 shows the high variability 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 particles removed. By performing a many to one mapping of these domains from underwater to not underwater (what the image would look like in the air), algorithms that have
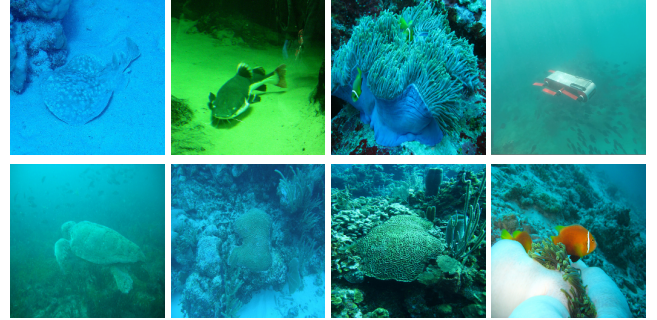

Fig. 1. Sample underwater images displaying the diversity of distortion that can occur. **Maybe say something about how some images lost all color, but some kept some color for close objects.**

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. Often times, these networks recquire large amounts of data, either labeled or paired with ground truth. For the problem of automatically colorizing grayscale images, paired training data is essentially free 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 visual effect lack ground truth. We use the recently proposed CycleGAN [3], which learns to translate an image from domain X to domain Y, as a way to generate a paired dataset. By letting X be a set of undistorted underwater images, 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 [4], [5], most are focused on the task of grayscale to color. The work of [6] 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 [7], 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 [8], style transfer [9], and image-to-image translation [10], [3]. 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 [10], we use image pairs from two domains an input and ground truth.

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 [3] 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 [11] 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:
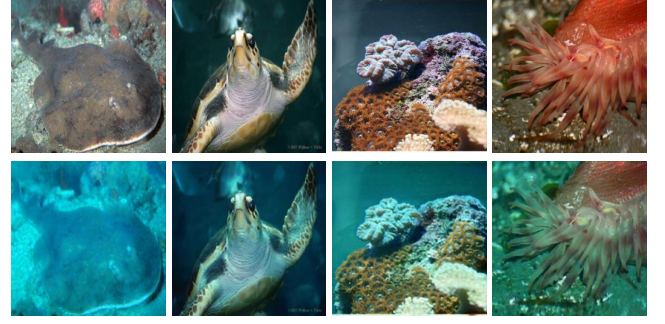

Fig. 2. Paired samples of ground truth and distorted images generated by CycleGAN. Top row: Ground truth. Bottom row: Generated samples.

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

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 [12], [13], [14], [15]. We focus on the Wasserstein GAN (WGAN) [13] formulation, which proposes to use the Earth-Mover or *Wasserstein-1* distance W by constructing a value function using the Kantorovich-Rubinstein duality [16]. 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) [14], which instead of clipping network weights like in [13], ensures the Lipschitz constraint by enforcing a soft constraint on the gradient norm of the discriminator's output with respect to its input. Following [14], our new objective then becomes

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

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 λ_{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

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

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

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

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 [17]. 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

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

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

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

D. Network Architecture and Training Details

Our generator network is a fully convolutional encoder-decoder, similar to the work of [10], which is designed as a "U-Net" [18] 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 4×4 with stride 2. Convolutions in the encoder portion of the network are followed by batch normalization [19] and a leaky ReLU activation with slope 0.02, while transpose convolutions in the decoder are followed by a ReLU activation [20] (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 [21], 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 [14] recommend layer normalization [22], but we found no significant improvements. In all of our experiments, we use $\lambda_1 = 100$, $\lambda_{GP} = 10$, batch size of 32, Adam Optimizer [23] with learning rate $1e-4$, and train for 100 epochs. 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 [24]. [1]

IV. EXPERIMENTS

A. Datasets

We used several subsets of Imagenet [25] for training and evaluation of our methods. We also evaluate a tracking algorithm on a diving video taken from Youtube. 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.

B. Evaluation

We trained UGAN and UGAN-P on the image pairs generated by CycleGAN, and evaluated entirely 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 $[-1, 1]$.

While many of the distorted images in the test set contain a blue or green hue, 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. Figure X shows a sample of such an image, which implies it does not simply subtract a variable amount of green or blue from the entire image.

C. Comparison to CycleGAN

note also this is on cyclegan's training set - retraining now with a test set in mind....should be quick. 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. Second, show some general results. Explain how our network is able to do three very important things: 1) Restore an image to its natural colors with varying amounts of distortion, i.e. blue hue, green hue, aka not just one type of color distortion. 2) The network does not just subtract a certain hue from the entire image. Show how in the clownfish image, part of the image has good colors, while part of it doesn't. Our network is able to leave correct colors. Third, show that even images taken towards the water's surface, either above or below, are able to be restored correctly.

Third mention that since CycleGAN has a cycle consistent nature, compare their results. Then from there go into how ours are better by showing results from Canny 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 [26] 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 [27].

Fig. 4 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.

REFERENCES

[1] F. Shkurti, A. Xu, M. Meghjani, J. C. G. Higuera, Y. Girdhar, P. Giguere, B. B. Dey, J. Li, A. Kalmbach, C. Prahacs, *et al.*, "Multi-domain monitoring of marine environments using a heterogeneous robot team," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 1747–1753, IEEE, 2012.

[2] L. Whitcomb, D. R. Yoerger, H. Singh, and J. Howland, "Advances in underwater robot vehicles for deep ocean exploration: Navigation, control, and survey operations," in *Robotics Research*, pp. 439–448, Springer, 2000.

[3] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *arXiv preprint arXiv:1703.10593*, 2017.

[4] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *European Conference on Computer Vision*, pp. 649–666, Springer, 2016.

[5] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be color!: joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 110, 2016.

[6] L. A. Torres-Méndez and G. Dudek, "Color correction of underwater images for aquatic robot inspection," in *EMMCVPR*, pp. 60–73, Springer, 2005.

[7] J. Li, K. A. Skinner, R. M. Eustice, and M. Johnson-Roberson, "Watergan: Unsupervised generative network to enable real-time color correction of monocular underwater images," *arXiv preprint arXiv:1702.07392*, 2017.

[8] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2536–2544, 2016.

[9] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[10] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint arXiv:1611.07004*, 2016.

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[12] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," *arXiv preprint ArXiv:1611.04076*, 2016.

[13] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[14] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein gans," *arXiv preprint arXiv:1704.00028*, 2017.

[15] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," *arXiv preprint arXiv:1609.03126*, 2016.

[16] C. Villani, *Optimal transport: old and new*, vol. 338. Springer Science & Business Media, 2008.

[17] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," *arXiv preprint arXiv:1511.05440*, 2015.

[18] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, Springer, 2015.

[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning* (F. Bach and D. Blei, eds.), vol. 37 of *Proceedings of Machine Learning Research*, (Lille, France), pp. 448–456, PMLR, 07–09 Jul 2015.

[20] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

[21] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[22] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[23] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[24] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 248–255, IEEE, 2009.

[26] M. J. Islam and J. Sattar, "Mixed-domain biological motion tracking for underwater human-robot interaction," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4457–4464, IEEE, 2017.

[27] F. Shkurti, W.-D. Chang, P. Henderson, M. J. Islam, J. C. G. Higuera, J. Li, T. Manderson, A. Xu, G. Dudek, and J. Sattar, "Underwater multi-robot convoying using visual tracking by detection," in *2017 IEEE International Conference on Intelligent Robots and Systems*, IEEE, 2017.
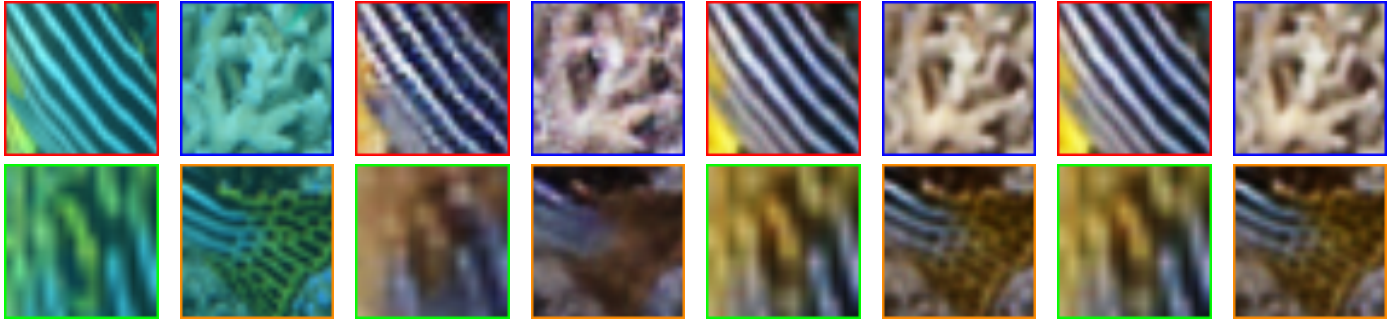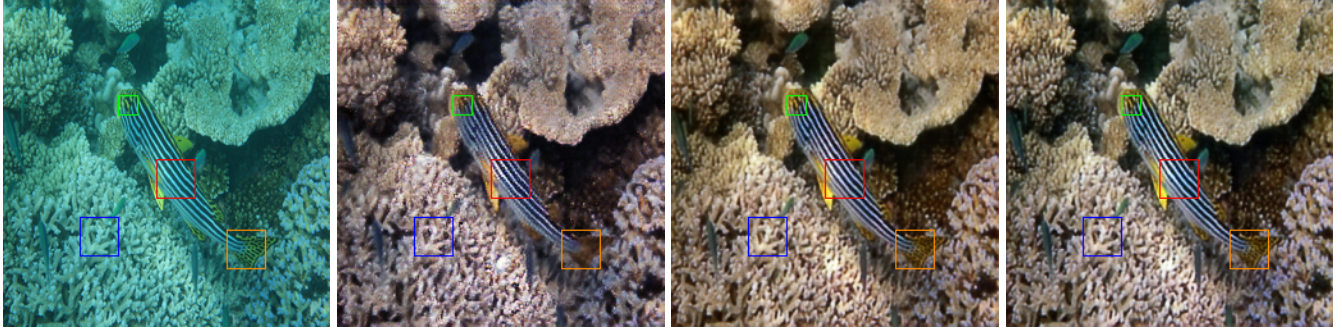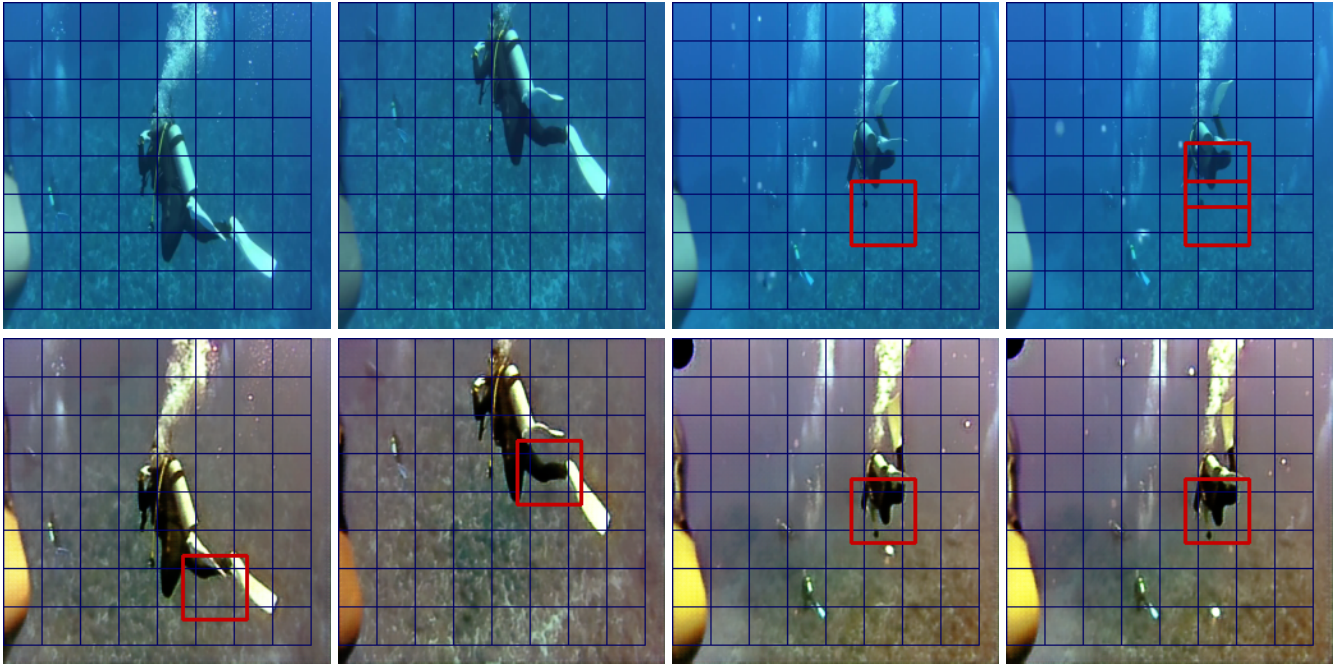
Fig. 3. Zooming in on comparisons.

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

Fig. 4. Performance of MDPM tracker [26] 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.

5 Original	CycleGAN	UGAN	UGAN-P
0	125.78	80.68	78.49
0	94.57	38.22	37.29
0	100.69	87.81	87.95

Fig. 5. 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.