Underwater Image Enhancement

Surya Boddu
Boston University
Boston, Massachusetts
bsurya27@bu.edu

Yashwanth Samudrala
Boston University
Boston, Massachusetts
yash0512@bu.edu

Yukkta Seelam
Boston University
Boston, Massachusetts
yukktas@bu.edu

Abstract

The visual quality of underwater imagery suffers from multiple visual artifacts caused by scattering, absorption and turbidity, leading to color alteration and visibility reduction. The standard image enhancement techniques show limited success in maintaining both fine details alongside balanced colors. Under this work we developed a ViT-CycleGAN architecture that merges Vision Transformer (ViT) encoder operations within CycleGAN to optimize underwater image enhancement. The ViT encoder identifies extensive contextual information yet the UNet-style decoder maintains both spatial organization and textural characteristics. The network training executes adversarial and cycle-consistency and identity loss optimization to maintain color accuracy while minimizing artifacts and preserving shape structure. The proposed ViT-CycleGAN model surpasses baseline CycleGAN through experimental testing, which shows superior performance in PSNR along with SSIM and FID metrics for underwater image enhancement.

1. Introduction

Underwater images suffer from a wide range of distortions caused by water properties such as light scattering, color absorption and turbidity. These significantly reduce the visibility and quality of images captured in underwater environments. Traditional image enhancement techniques struggle to restore the true colors and details of underwater images, which often leads to unnatural color shifts. We have seen CycleGANs [4], a type of Generative Adversarial Network for unpaired image-to-image translations , that were good at image enhancement. But their ability to handle complex visual distortions underwater is limited. They struggle to capture global image context and maintain fine details, which are crucial for accurate underwater image restoration.

In this work, we propose an enhanced CycleGAN architecture that replaces the standard convolutional encoder in the generator with a Vision Transformer (ViT). The ViT encoder is capable of modeling long-range dependencies and capturing global contextual features, which we hypothesize to be beneficial for resolving color distortions and structural inconsistencies in underwater images. The ViT features are decoded using a UNet-style upsampling decoder, with optional skip connections that help preserve low-frequency content. We retain the original PatchGAN-based discriminators and train the network using standard adversarial, cycle-consistency, and identity losses.

Our objective is to investigate whether incorporating ViT-based representations into the CycleGAN framework improves the visual quality of underwater image enhancement, and to evaluate its performance both qualitatively and quantitatively.

2. Related Work

Underwater image enhancement has garnered significant attention due to the critical challenges posed by underwater environments, such as color distortion, scattering, and light attenuation, which limit visibility and degrade image quality. Generative Adversarial Networks (GANs) have emerged as powerful tools to address these challenges, enabling complex image transformations that restore clarity and detail.

CycleGAN and Related GAN-Based Models

CycleGAN [4], a pioneering framework for unpaired image-to-image translation, introduced cycle-consistency loss to ensure a one-to-one mapping between domains. This loss has proven essential in scenarios like underwater image enhancement, where paired data is scarce. Enhancements of CycleGAN, such as UVCGAN [3], incorporate advanced architectures like Vision Transformers to capture non-local dependencies and improve the fidelity of translated images. These advancements demonstrate superior results compared to traditional CycleGAN approaches, offering improved realism and domain adaptation.

GANs in Underwater Image Enhancement
GANs specifically tailored for underwater contexts, such as WaterGAN, focus on reconstructing high-quality underwater visuals by simulating the physical degradation of underwater images[1]. Models like DenseGAN and MEvo-GAN further leverage this architecture to tackle specific underwater challenges, including backscatter reduction and depth-based color restoration. These methods often integrate GANs with traditional enhancement techniques for improved robustness

3. Proposed Method

This project has been built on top of an existing Cycle-GAN [4] work by Zhu et al. .The base model, CycleGAN uses a pair of GANs. Each generator translates images between two domains (raw underwater images and enhanced underwater images in our case), while the discriminators attempt to distinguish real images from generated ones.

The CycleGAN is trained using a combination of three losses:

- **Adversarial Loss:** This loss encourages each generator to produce outputs indistinguishable from real images in the target domain. It is defined as:

$$\mathcal{L}_{GAN}(G_{A\to B}, D_B) = \mathbb{E}_{y\sim B}\left[\log D_B(y)\right] + \mathbb{E}_{x\sim A}\left[\log\left(1 - D_B(G_{A\to B}(x))\right)\right] \quad (1)$$

- **Cycle Consistency Loss:** This loss ensures that translating an image to the other domain and back again reconstructs the original input:

$$\mathcal{L}_{cyc}(G_{A\to B}, G_{B\to A}) = \mathbb{E}_{x\sim A}\left[\|G_{B\to A}(G_{A\to B}(x)) - x\|_1\right] + \mathbb{E}_{y\sim B}\left[\|G_{A\to B}(G_{B\to A}(y)) - y\|_1\right] \quad (2)$$

- **Identity Loss:** This loss helps preserve domain-specific features by encouraging generators to behave like identity mappings when given inputs from their own domain:

$$\mathcal{L}_{idt}(G_{A\to B}) = \mathbb{E}_{y\sim B}\left[\|G_{A\to B}(y) - y\|_1\right]$$

$$\mathcal{L}_{idt}(G_{B\to A}) = \mathbb{E}_{x\sim A}\left[\|G_{B\to A}(x) - x\|_1\right]$$

The total loss function for the generator is then expressed as:

$$\mathcal{L}_G = \lambda_{GAN}\mathcal{L}_{GAN} + \lambda_{cyc}\mathcal{L}_{cyc} + \lambda_{idt}\mathcal{L}_{idt}$$

We propose a modification to this CycleGAN architecture where the generator leverages a Vision Transformer (ViT) encoder instead of traditional convolutional layers. The goal is to enable better global context modeling and enhance image realism.

3.1. Architecture of ViT-CycleGAN

In our proposed method, we replace the ResNet-based generator in the original CycleGAN with a hybrid ViT-UNet generator. This architecture combines the global feature modeling capabilities of a Vision Transformer (ViT) with the local structure-preserving advantages of a UNet-style decoder. The objective is to improve the generative quality of unpaired underwater image enhancement by capturing both semantic and spatial information effectively.

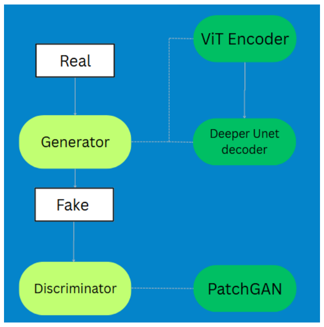

Figure 1. ViT architecture.

3.1.1 Generator Architecture: ViTUNetGeneratorV2

Each generator ($G_{A\to B}$ and $G_{B\to A}$) follows the structure outlined below:

Input Image: An RGB image $x \in \mathbb{R}^{3\times224\times224}$ from either domain A or B is passed to the generator. A 1×1 convolutional layer ensures the input has 3 channels if not already in that format.

ViT Encoder: The input image is tokenized into non-overlapping 16×16 patches using the patch embedding module of ViT. This results in:

$$\text{tokens} \in \mathbb{R}^{B\times N\times D}, \quad \text{where} \quad N = 196, \quad D = 768$$

A class token is prepended, and positional embeddings are added. The token sequence is processed through 12 transformer blocks. The class token is discarded before reshaping.

Feature Map Reconstruction: The remaining 196 patch tokens are reshaped into a 2D spatial feature map:

$$\text{ViT features} \in \mathbb{R}^{B\times768\times14\times14}$$

This serves as the input to the decoder.

UNet-style Decoder (ConvTranspose2d): The decoder consists of 4 sequential upsampling blocks:

- $14 \times 14 \rightarrow 28 \times 28$: $768 \rightarrow 512$ channels
- $28 \times 28 \rightarrow 56 \times 56$: $512 \rightarrow 256$ channels
- $56 \times 56 \rightarrow 112 \times 112$: $256 \rightarrow 128$ channels
- $112 \times 112 \rightarrow 224 \times 224$: $128 \rightarrow 64$ channels

Each block consists of a transposed convolution, batch normalization, and ReLU activation.

Skip Connection (Optional): If enabled, a skip connection from the encoder's projected input $x_{\text{proj}} \in \mathbb{R}^{B \times 3 \times 224 \times 224}$ is concatenated to the decoder output before the final convolution.

Output Layer: A 3×3 convolution reduces the channel count to 3, and a Tanh activation bounds the output image values to $[-1, 1]$, matching CycleGAN conventions.

3.1.2 Discriminator Architecture

Each discriminator (D_A and D_B) follows the PatchGAN architecture, a fully convolutional network designed to classify overlapping image patches as real or fake. The input image size is $3 \times 224 \times 224$, and the output is a matrix of real/fake scores for each spatial patch (e.g., 30×30 or 70×70).

Input Layer:

- Input: $3 \times 224 \times 224$ RGB image
- Conv2d: in_channels = 3, out_channels = 64, kernel_size = 4, stride = 2, padding = 1
- Activation: LeakyReLU (negative_slope = 0.2)
- Output: $64 \times 112 \times 112$

Second Layer:

- Conv2d: $64 \rightarrow 128$, kernel_size = 4, stride = 2, padding = 1
- InstanceNorm2d(128)
- LeakyReLU (0.2)
- Output: $128 \times 56 \times 56$

Third Layer:

- Conv2d: $128 \rightarrow 256$, kernel_size = 4, stride = 2, padding = 1
- InstanceNorm2d(256)
- LeakyReLU (0.2)
- Output: $256 \times 28 \times 28$

Fourth Layer:

- Conv2d: $256 \rightarrow 512$, kernel_size = 4, stride = 1, padding = 1
- InstanceNorm2d(512)
- LeakyReLU (0.2)
- Output: $512 \times 27 \times 27$

Output Layer:

- Conv2d: $512 \rightarrow 1$, kernel_size = 4, stride = 1, padding = 1
- No activation (raw logits used for adversarial loss)
- Output: $1 \times 26 \times 26$

Each output value represents a patch-level real/fake score, effectively acting as a localized classifier over the input image. This approach encourages the generator to focus on high-frequency details such as textures and edges.

The same architecture is reused for both discriminators (D_A and D_B), ensuring symmetry between both domains.

3.2. Forward Pass

During training, each iteration processes a batch of real images from both domains A and B. The generator and discriminator pairs work collaboratively to minimize adversarial and reconstruction losses while learning domain mappings.

Domain Translation (Forward Mapping): Each generator transforms images from one domain to the style of the other:

$$\hat{x}_B = G_{A \rightarrow B}(x_A), \quad \hat{x}_A = G_{B \rightarrow A}(x_B)$$

These outputs are intended to be indistinguishable from real images in the target domain.

Cycle Reconstruction (Backward Mapping): Each generated (fake) image is passed through the inverse generator to reconstruct the original input:

$$\tilde{x}_A = G_{B\to A}(\hat{x}_B) \approx x_A, \quad \tilde{x}_B = G_{A\to B}(\hat{x}_A) \approx x_B$$

This enforces cycle consistency, ensuring that content is preserved despite domain translation.

Identity Mapping (Optional Regularization): Real images are passed through their own domain's generator to encourage identity mapping:

$$\text{id}_A = G_{B\to A}(x_A) \approx x_A, \quad \text{id}_B = G_{A\to B}(x_B) \approx x_B$$

This discourages unnecessary changes and helps maintain domain-specific traits such as color tones and textures.

Discriminator Evaluation: Each discriminator receives both real and fake images:

$$D_B(x_B), \quad D_B(\hat{x}_B)$$
$$D_A(x_A), \quad D_A(\hat{x}_A)$$

Each discriminator attempts to distinguish between real and generated samples using adversarial loss.

The following losses are computed during each training iteration:

- **Adversarial Loss:** This loss pushes the generator to produce outputs that are indistinguishable from real images.
- **Cycle Consistency Loss:** Ensures that translating an image to the other domain and back again reconstructs the original input.
- **Identity Loss (Optional):** Penalizes the generator when it alters input images that already belong to the target domain.

The total loss function for the generator is expressed as:

$$\mathcal{L}_G = \mathcal{L}_{GAN} + \lambda_{cyc}\mathcal{L}_{cyc} + \lambda_{idt}\mathcal{L}_{idt}$$

where λ_{cyc} and λ_{idt} are hyperparameters that control the weight of the cycle consistency and identity losses, respectively.

3.3. Backward Pass and Parameter Update

After computing the losses, the backward pass is initiated to update the network parameters.

1. Discriminator Update: The discriminators D_A and D_B are updated independently by minimizing the adversarial loss. The discriminator loss is calculated as:

$$\mathcal{L}_D = \frac{1}{2}\left(\mathbb{E}\left[(D_A(x_A)-1)^2\right] + \mathbb{E}\left[(D_A(\hat{x}_A))^2\right]\right)$$

$$+\frac{1}{2}\left(\mathbb{E}\left[(D_B(x_B)-1)^2\right] + \mathbb{E}\left[(D_B(\hat{x}_B))^2\right]\right)$$

The gradients are backpropagated, and the discriminator parameters are updated using the Adam optimizer.

2. Generator Update: The generators $G_{A\to B}$ and $G_{B\to A}$ are updated next. The generator loss is a weighted combination of adversarial loss, cycle consistency loss, and identity loss:

$$\mathcal{L}_G = \mathbb{E}\left[(1 - D_B(\hat{x}_B))^2\right] + \lambda_{cyc}\mathbb{E}\left[\|\tilde{x}_A - x_A\|_1 \right. \\ \left. + \|\tilde{x}_B - x_B\|_1 \quad (3)$$

$$+\lambda_{idt}\mathbb{E}\left[\|\text{id}_A - x_A\|_1 + \|\text{id}_B - x_B\|_1\right]$$

Gradients are calculated, and the generator parameters are updated using the Adam optimizer.

3. Optimization Strategy: The generator and discriminator updates are alternated to maintain adversarial balance. We used the Adam Optimizer with learning rates α_G and α_D, typically set to 0.0002, with $\beta_1 = 0.5$ and $\beta_2 = 0.999$. Learning rates are kept constant for a fixed number of epochs and then linearly decayed (100 and 50 in our case).

4. Experiment Details

In our experiments for underwater image enhancement, we developed and compared several architectures that leverage Vision Transformers (ViT) for feature extraction and U-Net-like decoders for image reconstruction. Our final model, ViTUNetGeneratorV2, utilizes a pre-trained ViT encoder (vit_base_24) with a patch size of 16 and an input size of 224x224. To accommodate our 256x256 underwater images, we employed a 1x1 convolutional layer to adjust the input channels, followed by the ViT encoder, which outputs features on a 14x14 grid. These features are then upsampled by our custom decoder, DecoderUNetDeep, which uses transposed convolutions with kernel size 3, stride 2, and output padding 1 to precisely reconstruct the image to 224x224, ensuring minimal artifacts. We also explored two alternative decoders: one with transposed convolutions of kernel size 4 (ViTUNetGenerator) and another with bilinear upsampling (ViTUNetGeneratorV3). The final model's decoder was selected for its ability to maintain fine details and reduce scattering effects, crucial for enhancing underwater images. Additionally, the use of skip connections in the decoder helps preserve low-level features, further improving the quality of the enhanced images. The encoder's weights were frozen to leverage pre-trained knowledge and reduce computational overhead, allowing the model to focus on learning the enhancement-specific transformations in the decoder. This approach proved effective in addressing the unique challenges of underwater image enhancement, such as color distortion and low contrast.

4.1. Dataset

For training the GAN model, we used a mixed dataset containing images from various sources, including the EUVP, SUID, and UIEB datasets. The dataset was taken from the work of Srivastava et al., [2]. It includes annotations for 15 classes of objects commonly found in aquatic environments, such as masks, cans, cellphones, electronics, glass bottles, gloves, metals, nets, plastic bags, plastic bottles, plastics, rods, sunglasses, and tires.

We used the train subset of it and split into train and test splits. The train split consisted of 5515 images which we used for training in our experiments. The remaining were used for testing and calculating the metrics which will be discussed in the further sections.

4.2. Hyperparameters

The training of the proposed ViT-CycleGAN model was conducted using the following key hyperparameters:

Training Configuration:

- **Batch Size:** 8
- **Number of Epochs:** 100 epochs with an additional 50 epochs for linear decay
- **Image Size:** Input images were resized to 224×224 pixels
- **Dataset Mode:** Unaligned, allowing unpaired image translation
- **Dataset Size:** 5515 training images

Optimization Settings:

- **Learning Rate:** 0.0001 for both generator and discriminator
- **Learning Rate Decay:** Linear decay after the initial 100 epochs
- **Optimizer:** Adam with $\beta_1 = 0.5$ and $\beta_2 = 0.999$
- **Weight Initialization:** Normal distribution with $\mu = 0$ and $\sigma = 0.02$

Loss Weights:

- **Cycle Consistency Loss Weight (λ_{cyc}):** 10.0
- **Identity Loss Weight (λ_{idt}):** 0.5

Other Relevant Settings:

- **GAN Mode:** Least Squares GAN (LSGAN)
- **Data Augmentation:** Random cropping and flipping enabled
- **Checkpoint Frequency:** Model checkpoints saved every 5 epochs

5. Results and Observatios

Our modified CycleGAN model with a Vision Transformer (ViT) encoder demonstrates several key improvements over the baseline CycleGAN in enhancing underwater images. To qualitatively assess performance, we visualize six representative outputs from both models. real_A, fake_B, rec_A, real_B, fake_A, and rec_B which allow us to analyze the forward translation (distorted to enhanced), reverse translation (enhanced to distorted), and cycle consistency (reconstruction fidelity).

We observe notable improvements across the following dimensions:

- **1. Color Fidelity:** The ViT-CycleGAN shows marked improvement in recovering natural and visually pleasing colors. In the sample shown, the baseline model's enhanced output (fake_B) appears dull and overcorrected, with muted orange-yellow tones on the fish. In contrast, our ViT-augmented model restores a vibrant and balanced color distribution that more closely matches the ground truth (real_B). This demonstrates the model's ability to better correct underwater color distortions introduced by absorption and scattering
- **2. Artifact Reduction:** Another significant improvement is the reduction of artifacts in the output images. The baseline CycleGAN often introduces checkerboard patterns and blotchy regions in dark or high-texture areas. Our ViT-CycleGAN minimizes these artifacts, producing cleaner, more natural-looking outputs. This suggests that the global attention mechanism in the ViT encoder helps avoid local inconsistencies and better generalizes over spatially complex areas.

- **3. Texture Preservation:** Our model preserves fine-grained textures more effectively than the baseline. In particular, details in the fish's outline and the surrounding coral are visibly sharper and more defined in the ViT-CycleGAN outputs. The base model tends to produce smoother, less detailed outputs, likely due to its limited receptive field and lack of global spatial awareness. The ViT component helps the model retain high-frequency detail crucial for underwater scenes

- **4. Cycle Consistency:** The reconstructions (rec_A, rec_B) from the ViT-CycleGAN closely resemble their original inputs, more so than those generated by the baseline model. This indicates stronger cycle-consistency performance, implying that the model is learning more robust mappings in both directions. Maintaining semantic and structural information through both translations shows the value of incorporating ViT in the encoding stage.

Table 1. Comparison of image quality metrics between ViT-CycleGAN and CycleGAN. Higher PSNR and SSIM, and lower FID indicate better performance.

Model	**PSNR** ↑	**SSIM** ↑	**FID** ↓
ViT-CycleGAN	23.54	0.8142	30.28
CycleGAN	17.96	0.6048	74.47

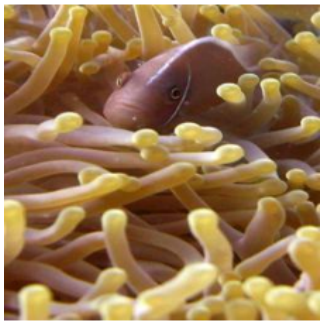
Figure 2. real image.

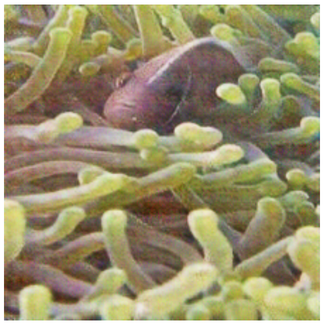
Figure 3. this is the image produced by the base model

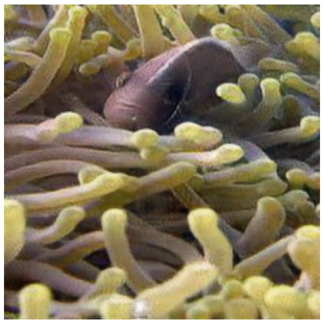
Figure 4. the image produced by our model (ViT).

6. Future Directions

- **1. Bilinear Decoder Variants:** Replace ConvTranspose2d layers in the decoder with bilinear upsampling followed by convolution to reduce checkerboard artifacts and produce smoother textures.

- **2. Perceptual Loss Integration:** Incorporate perceptual losses (e.g., using pretrained VGG features) to guide the generator toward more realistic color and texture reconstruction, beyond simple pixel-level differences.

- **3. Task-Oriented Evaluations:** Assess enhancement quality using downstream tasks like object detection or segmentation to verify whether the enhancements improve practical underwater vision performance.

- **4. User-Controlled Enhancement:** Introduce tunable parameters for saturation, haze removal, or contrast that allow users to adjust the enhancement output interactively for different underwater scenarios.

References

[1] K. K. Babu, A. Tabassum, B. Navaneeth, T. Jahnavi, and Y. Akshaya. Underwater image enhancement using generative adversarial networks: A survey, 2025. 2

[2] A. K. Srivastava, K. Singh, and S. Gupta. Underwater image enhancement and trash detection using deep learning. In *2024 IEEE Students Conference on Engineering and Systems (SCES)*, pages 1–6, 2024. 5

[3] D. Torbunov, Y. Huang, H. Yu, J. Huang, S. Yoo, M. Lin, B. Viren, and Y. Ren. Uvcgan: Unet vision transformer cycle-consistent gan for unpaired image-to-image translation, 2022. 1

[4] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020. 1, 2

7. Contributions

- **Yashwanth Samudrala:** Optimized training parameters through systematic adjustments of learning rate together with batch size and cycle consistency weights to achieve the best training stability. The task involved the set up of training scripts and maintenance of experimental reproducibility alongside the configuration process.
- **Yukkta Seelam:** Managed the implementation of data loaders and applied data augmentation techniques to enhance model generalization. Also monitored the training process, identifying potential training instabilities and adjusting configurations as needed.
- **Surya Boddu:** Is responsible for both qualitative and quantitative analysis, using relevant evaluation metrics to assess the model's performance against baseline models. Handled documentation, including detailed reporting of experimental setups, architecture modifications, and final outcomes.