



ShipGAN: Generative Adversarial Network based simulation-to-real image translation for ships

Yuxuan Dong^a, Peng Wu^c, Sen Wang^b, Yuanchang Liu^{c,*}

^a Computer Science, University College London, London WC1E 6BT, UK

^b Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2AZ, UK

^c Department of Mechanical Engineering, University College London, Torrington Place, London WC1E 7JE, UK

ARTICLE INFO

Keywords:

Generative networks
Sim-to-real translation
Autonomous navigation
Unmanned surface vessels

ABSTRACT

Recent advances in robotics and autonomous systems (RAS) have significantly improved the autonomy level of unmanned surface vehicles (USVs) and made them capable of undertaking demanding tasks in various environments. During the operation of USVs, apart from normal situations, it is those unexpected scenes, such as busy waterways or navigation in dust/nighttime, impose most dangers to USVs as these scenes are rarely seen during training. Such a rare occurrence also makes the manual collection and recording of these scenes into dataset difficult, expensive and inefficient, with the majority of existing public available datasets not able to fully cover them. One of many plausible solutions is to purposely generate these data using computer vision techniques with the assistance from high-fidelity simulations that can create various desirable motions/scenarios. However, the stylistic difference between the simulation images and the natural images would cause a domain shift problem. Hence, there is a need for designing a method that can transfer the data distribution and styles of the simulation images into the realistic domain. This paper proposes and evaluates a novel solution to fill this gap using a Generative Adversarial Network (GAN) based model, ShipGAN, to translate the simulation images into realistic images. Experiments were carried out to investigate the feasibility of generating realistic images using GAN-based image translation models. The synthetic realistic images from the simulation images were demonstrated to be reliable by the object detection and image segmentation algorithms trained with natural images.

1. Introduction

The research of Uncrewed Surface Vessels or Unmanned Surface Vehicles (USVs) is becoming increasingly important. Across various application areas from search and rescue to environment monitoring, USVs are proven to be reliable and flexible with the payload to carry several sensors or some special equipment. As shown in Fig. 1, a model of WAM-V 20 series USVs as well as a configurable USV platform sponsored by Society of Maritime Industries (SMI) are presented. Both vessels are equipped with GPS, Lidar, and cameras for perception and navigation. In fact, when undertaking missions, a real-time and robust environment perception system is the key to achieve autonomous navigation and completing tasks for USVs. This is particularly important when USVs are being increasingly deployed in extreme environments for exploration and prospecting. For the perception system, robust detection and recognition of any object on the water surfaces are

essential for autonomous navigation as any object is likely to be an obstacle causing a potential danger to the sailing ships.

Besides the common scenarios in the real world, USVs also need to deal with complex situations, such as busy waterways and traffic in dusk or at night. These situations hold great importance for operational safety but unfortunately have not been well recorded and included in the present maritime datasets with only limited amounts of videos or images available. For example, Fig. 2 shows frames extracted from the videos of the MODD2 dataset (Bovcon et al., 2018) and the Singapore Maritime Dataset (Prasad et al., 2017) and they are captured by cameras on a USV and onshore, respectively. These two datasets are widely used in training the visual perception systems of USVs. However, it can be seen that in most of the videos in these datasets, there is only one or even no ship sailing at a visible speed. These scenarios are not sufficient to train models which USVs can use to deal with emergency situations

* Corresponding author.

E-mail addresses: yuxuan.dong.21@ucl.ac.uk (Y. Dong), peng.wu.14@ucl.ac.uk (P. Wu), sen.wang@imperial.ac.uk (S. Wang), yuanchang.liu@ucl.ac.uk (Y. Liu).

<https://doi.org/10.1016/j.apor.2022.103456>

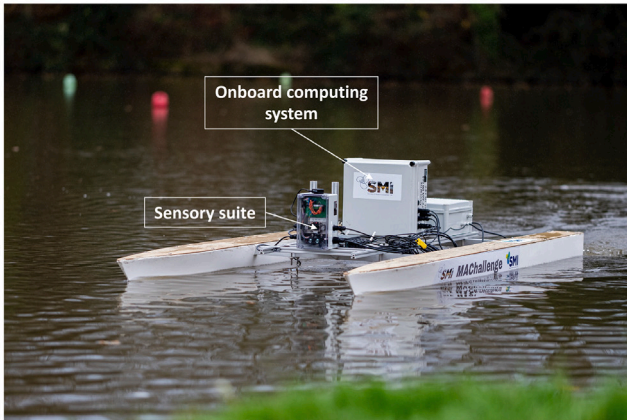
Received 9 November 2022; Received in revised form 15 December 2022; Accepted 31 December 2022

Available online 11 January 2023

0141-1187/© 2023 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).



(a)



(b)

Fig. 1. (a) WAM-V 20 series USV (Marketing and Communication, 2022) (b) Configurable USV platform sponsored by Society of Maritime Industries, UK.

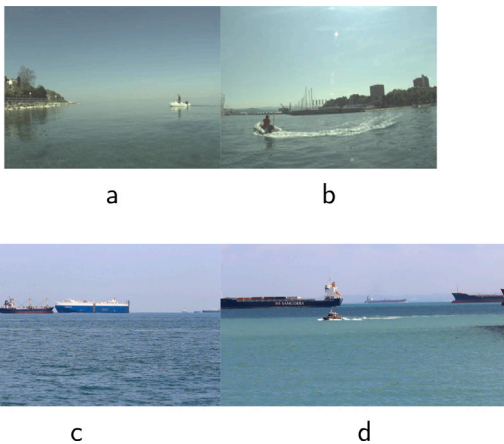


Fig. 2. Example images from existing maritime datasets. (a), (b) are from the MODD2 dataset (Bovcon et al., 2018) and (c), (d) are from the Singapore Maritime Dataset (Prasad et al., 2017).

in busy waters. The desired images are similar to Fig. 2 in which several boats are sailing concurrently (see Fig. 3).

To address such an issue, one possible solution is to collect the data manually, which is however expensive or impossible in practice, as hours or even days are possibly required to record one uncommon scene which may only last for seconds. Or if we suppose that it is possible to manoeuvre several boats to navigate in an area at the same time to



Fig. 3. An example image of busy waterways (News Isle Wight, 2013).

create different scenarios, it is still not practicable to deliberately and manually generate some dangerous or extreme scenarios.

Another approach is the use of simulation to generate high-fidelity scenarios which contain various ships' motions and environmental contexts. However, this is not a silver bullet with no compromises. The most prominent issue is that the texture within the simulation may not be visually realistic enough compared to real-world images. Although some simulation software or engines, such as Unity 3D (Haas, 2014) and Unreal Engine 5 (Epic Games, 2022), have robust rendering systems and realistic texture packs, they are computationally expensive and require high standard computer configurations to output images or videos with high graphic quality. This would still cause a problem in domain shift due to the data inefficiency limitation of deep learning networks (Motiian et al., 2017; Saito et al., 2019). In general, deep learning networks trained by millions of examples may provide good performances within the current training domain but cannot well generalize into new domains. In other words, when the training set and the testing set for a deep learning network are in different domains, the objects in the testing set could be incorrectly classified. The consequence of this is that well-rendered simulation images may visually look appealing, but they can still not fit the distribution of realistic images, causing domain shifts when they are used for training visual perception systems.

Hence, the core research challenge becomes how to efficiently and effectively obtain a dataset that captures complex scenarios sharing the same data distribution/domain with real images. In recent years, domain transfer has been actively used as a solution to address this challenge (Saito et al., 2019; Sun and Saenko, 2016; Peng et al., 2018; Hoffman et al., 2016). When images in a simulation environment can be transferred into a realistic domain, the generated images will be able to contain artificial scenarios with visually realistic textures. Generative Adversarial Networks (GANs) can be trained to achieve image translation between different domains, and more specifically the Cycle-Consistent Adversarial Networks, namely CycleGAN, is proposed to accomplish unpaired image translation between two domains (Zhu et al., 2017). For example, a horse in an image can be transferred into a zebra by CycleGAN, and an apple can be transferred into an orange. Therefore, we argue that the CycleGAN can also be used to translate an image in a simulation domain into a realistic domain and transfer a 3D model of a boat into a real boat. However, different to those attempts made within the CycleGAN examples, the application of CycleGAN into marine application requires considerations of maritime features such as ocean waves, ship dynamics and ship-wave interactions.

Therefore, the aim of this paper is to generate synthetic images of ships with enriched and realistic textures via a simulation-to-real pipeline, which can translate ship images simulated in a virtual environment into real scenarios. To achieve this goal, the paper has three key objectives:

- To construct desired scenarios containing ships with all natural physical effects. At this stage, the most important thing is to simulate the physically realistic ship motions including the interaction

between the ships and the sea in a simulation environment with no rendering or textures required. The scenario simulation is done in Unity 3D, a graphic engine, in which real-world physics can be simulated by its Scripting API.

- To generate visually realistic images that contain the constructed scenarios obtained from the previous step. This generation task can be accomplished by using generative models, such as Generative Adversarial Networks. In this paper, GAN-based approaches are chosen to synthesize new realistic images with one to translate the simulation images into natural images and the other one to generate nighttime images.
- To evaluate the reliability in visual perception systems that can be used for USVs. The synthetic images are expected to be visually realistic via inspection by our naked eyes and also, the fake ships in the synthetic images should be able to ‘fool’ the object detection models that are trained by real ship images. The object detection algorithms, Yolov4 (Bochkovskiy et al., 2020) and Detectron2 (Pham et al., 2020), are used to test the fidelity of the synthetic images. It is anticipated that if the generated ships can be detected as real ships with a high confidence score, they can be used to train the visual perception systems for autonomous systems.

The contributions of this paper can be summarized as follows.

- A GAN-based system, i.e., ShipGAN, has been proposed for synthesizing realistic images of ships from Unity simulation models. This is the very first attempt of converting simulation images into realistic images by using generative adversarial networks in maritime environments.
- Vivid ship movement scenarios have been created using Unity 3D by considering interactions between ships and waves. A seamless connection between Unity 3D and ShipCAN has been achieved to facilitate sim-to-real translation.
- Enriched validation studies on the synthetic ship images have been carried out using object detection and image segmentation techniques. This ensures the reliability of the synthetic images in practical applications.

2. Related works

Generative Adversarial Networks (GANs) have one or multiple generators and discriminators that are improving together but also against each other. The generators aim to generate images that are as close as possible to the images from the training sets and the discriminators try to distinguish the difference between the generated images and the training images. Therefore, an ideal generator’s output distribution would be able to match that of the training data. GANs can be roughly separated into unsupervised and supervised approaches and both approaches can translate images between domains. For supervised GANs, the training of the models requires paired training data from the source and target domains, while for unsupervised GANs the training data in two domains does not need to be related in any way. GANs are widely used in image style translation, data augmentation, and also medical images. For example, CycleGAN (Zhu et al., 2017), Pix2Pix (Isola et al., 2017) and UNIT (Liu et al., 2017) are three popular expanded GANs that are used in converting image styles. There have been numerous applications of GANs and their extensions, such as that Bargshady et al. (2022), Jin et al. (2021) and Karlsson and Welander (2018) illustrate the robustness and reliability of GANs in generating medical images. Besides, Zaher (2020), Arruda et al. (2019), Machiraju and Balasubramanian (2020) and Matsui and Ikehara (2021) show some examples of the applications of GANs in weather and day-and-night conversion for autonomous systems. However, these works focus on editing the existing scenarios using GANs rather than generating new scenarios, and they have very limited research on the complex situations for autonomous systems, especially for maritime applications.

As stated in the previous section, simulation can be used to generate any desired scenario and the problem left would be the domain shift when fitting the models trained by the simulation images to the real world. To address the domain shift, Peng and Saenko (2018) proposed a Deep Generative Correlation Alignment Network (DGCAN) aligning the images of 3D CAD models to some natural background images. However, as the features of the adapted 3D models in the result images are learnt from the background images, the distribution of the resultant images does not match the real world and the domain shift is not totally eliminated. Xu et al. (2021) proposed a data generation method which can be a substitute for real data collection using the conditional GAN, Pix2Pix, to translate the instance maps of the simulation scenarios into realistic images. This work sufficiently eliminated the domain shift problem when fitting the model training by the Pix2Pix-generated images to the real world, but this method required a large amount of paired training sets containing the real images and their corresponding instance maps and semantic labels. What is more, in Xu et al. (2021), obtaining the instance maps and semantic labels can be time-consuming and hence reduce the efficiency of the whole system. Therefore, there is a need for a direct converting method between the simulation images and the realistic images to replace the real data collection.

Compared to the relevant works discussed in this section, ShipGAN is a novel and reliable solution to generate visually realistic images according to the desired scenarios constructed in a simulation environment. With CycleGAN, the simulation images are directly translated into the realistic domain without the presence of instance maps and semantic labels, and the training data do not need to be paired, which reduces the complexity of data preparation. Most importantly, with ShipGAN, it is possible for us to get as many realistic images as possible by constructing the desired scenarios in simulation and then inputting the simulation images into the trained models to generate the corresponding images in the realistic domain.

3. ShipGAN methodology

3.1. System overview

To investigate the feasibility of converting Unity simulation images into visually realistic images to generate scenarios that we are interested in for autonomous systems of USVs, GAN is used to synthesize visually realistic images by referring to the models generated in Unity 3D. More specifically, a new network named as ShipGAN has been built, which is based on CycleGAN models and are trained and tested for image translation from simulation to real-world, and from daytime to night, respectively.

ShipGAN can be separated into two main parts including (1) simulating the motion of boats in Unity 3D and (2) training the image translation models. These two parts will be discussed with more details later in this section. Fig. 4 provides a graphical illustration about ShipGAN uses two separate CycleGAN models to achieve the translation between simulation and realistic domain, i.e., translating simulation images into the realistic daytime domain using Sim2Real CycleGAN and translating the daytime domain into the night domain using Day2Night CycleGAN.

Specific procedures to generate new ship images using ShipGAN are as follows:

1. Constructing various scenes containing boats in Unity 3D.
2. Preparing the training dataset which has two sets of data with one set containing the images of boats in the real domain and the other set having the images in the unity simulation domain. As the CycleGAN does not require the training images in two different domains to correspond to each other, the real images and the simulation images are not paired and their amounts do not need to be exactly the same.
3. Using the training set in the previous step to train the Sim2Real CycleGAN (highlighted in red in Fig. 4).

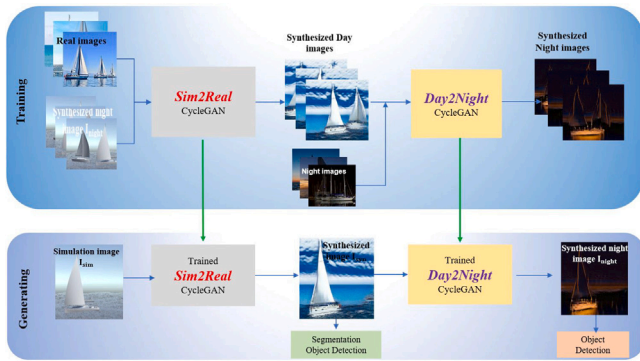


Fig. 4. Illustration of the procedures of ShipGAN.

4. Using the trained Sim2Real to transfer different unity simulation images into real domain.
5. Preparing the training dataset for Day2Night CycleGAN (highlighted in purple in Fig. 4).
6. Training and testing the Day2Night CycleGAN model.
7. Using the trained Day2Night model to translate daytime images into night images.
8. The final step is to prove the reliability of the synthetic images by applying image segmentation and object detection algorithms.

3.2. ShipGAN data generation

3.2.1. Simulation images of ships

Unity was used for creating ship models, and in this paper sailing boats are primarily modelled. The 3D model was used to construct the scenarios without any texture. The reason that the sailing boat was chosen is that a real sailing boat has relatively fewer complex features compared to other types of ships, which could effectively reduce the difficulty of training the GAN models, especially for the unpaired image translation CycleGAN.

There are five different types of motions for sailing boats used in this study, as shown in Fig. 5. Each scene lasts 10 s. In Scene1 and Scene2, the two sailing boats are sailing toward each other, while in Scene1, there is an angle of 45 degrees between the courses of the two sailing boats, and in Scene2, the two boats are sailing in the opposite direction. In Scene3, there is only one sailing boat sailing from the right to the left of the frame. In Scene4 and Scene5, Sailing Boat 2 is sailing away from Sailing Boat 1, while Sailing Boat 1 is sailing toward Sailing Boat 2, with angles of 45 degrees and 60 degrees between their courses respectively.

The reasons for these five scenes to be constructed are that there are different numbers of sailing boats and these five scenes contain various positional relationships between the two boats. What is more, these five scenarios cover situations where one ship is sailing alone, two ships are travelling opposite each other, two ships are moving away from each other, and two ships meet, etc. These situations can be used as the basis to construct other different scenarios.

To make the motions of the sailing boats more physically realistic, a wave object is generated by modelling the physical properties of the ocean in unity scripts which have options to adjust the scale, direction and speed of the wave. The mathematical model for generating the wave object is partially based on Ditzel (2022). Four floating points and a collision added around each sailing boat allow the sailing boats to interact with the wave object, which is presented by the four green spheres in Fig. 6. The displacements of the boats are achieved by the animation options in Unity.

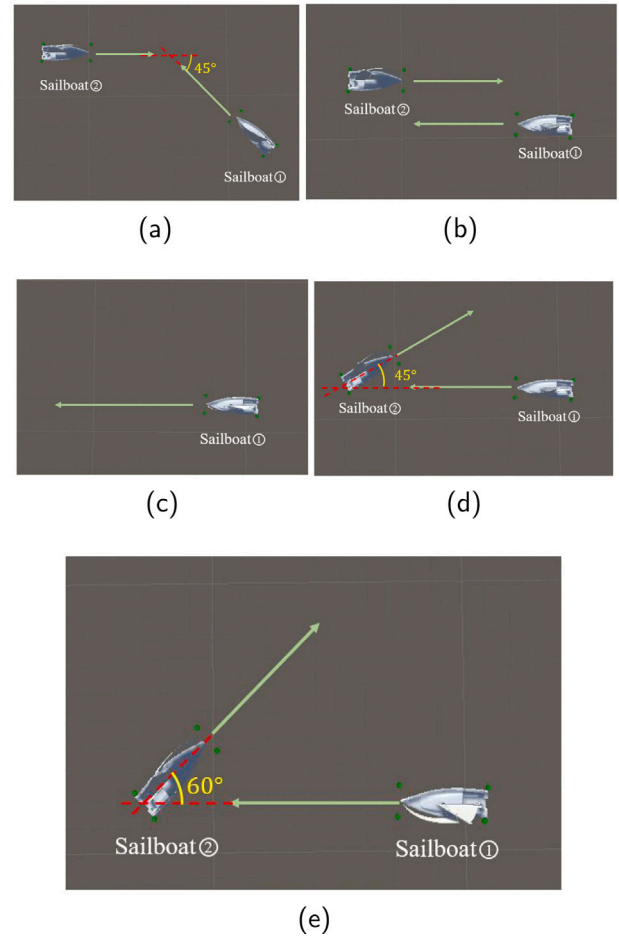


Fig. 5. Top views of the four scenes for training: (a) Top view of Scene 1; (b) Top view of Scene 2; (c) Top view of Scene 3; (d) Top view of Scene 4. (e) Top view of Scene 5 for testing.

The waves are assumed to be sinusoidal and different octaves with different heights and scales can be added together without superposition. Mesh was created on the surface of the wave object. The position vector of each vertex of the mesh is defined as (Ditzel, 2022):

$$P = \begin{bmatrix} x \\ a \sin k(x - ct) \end{bmatrix} \quad (1)$$

where P is the position vector, which gives a sine wave along the X dimension and a denotes the amplitude of the wave which can be manipulated. k is a factor that relates to the wavelength and is defined as

$$k = \frac{\pi}{\lambda} \quad (2)$$

where c and t represent the speed of the wave and a time offset respectively.

When simulating the floating effect of the sailing boats on an ocean surface, the air drag and water drag are defined and taken into account as an offset. The poses of the sailing boats depend on the positions of the four floating points, and the position vectors of the floating points are calculated by adding the offsets caused by drags to the position vectors of the connecting points of the wave to the floating points on the boats. To implement the simulation of waves, a collision object was inserted in Unity and meshes were constructed on that object. Then, the effect of waves was actually the movements of the vertex of the meshes which were modelled as sinusoid waves. The difference between this work and Ditzel (2022) is that in Ditzel (2022), the boat is driven by

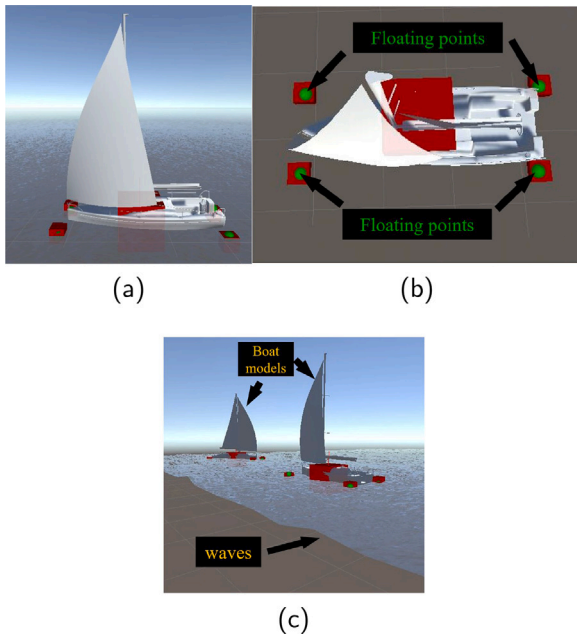


Fig. 6. Physical effects of sailing boats in Unity simulation. (a) Side view of the sailing boat with four floating points. (b) Top view of the sailing boat. (c) View of the sailing boat and the waves effect.

its own motor which is controlled by the script, but in this work, the motion of the sailing boat is determined by the position change between key-frames. Overall, when the boats move in the simulation, alongside the interaction between the boats and the wave, the sailing boat sways from side to side as it goes up and down.

3.2.2. Real images of ships

In parallel to the simulation dataset, real images are also extracted and used as inputs for ShipGAN. The real images of ships dataset contain 100 real daytime images of sailing boats which have a maximum of two sails to match the 3D model. Most images containing 1 sailing boat are cropped to have the sailing boat in the centre so that the training result will not be affected by features other than those of the sky, ocean and sailing boats. Then these images are resized into the size of 256×256 . There are also 50 real night images of sailing boats. However, due to the very limited number of images captured at the midnight, images shot at sunset are also included in the training set as long as there is an obvious change in the tune of the image compared to the daytime images. Examples of real images of ships are shown in Fig. 7.

3.3. Image translation

As described previously, the cornerstone of ShipGAN is the CycleGAN (Zhu et al., 2017), which is a successful approach for unpaired image-to-image translation using Generative Adversarial Network architecture. The goal of CycleGAN is to learn the mapping models between two domains, *Domain A* and *Domain B*, and the mapping between two domains is bi-directional. In this context, the underlying mechanism of the CycleGAN is an extension to the basic GAN architecture which can train two generator models and two discriminator models simultaneously. The main purpose of implementing CycleGAN is for domain transfer and an overview of the structure of CycleGAN is shown in Fig. 8.

Here we explain in details about how CycleGAN works for domain transfer using simulation images and real images as per examples in Fig. 8. Suppose there are two datasets, *A* and *B*, containing the simulation images and the real images respectively. The simulation

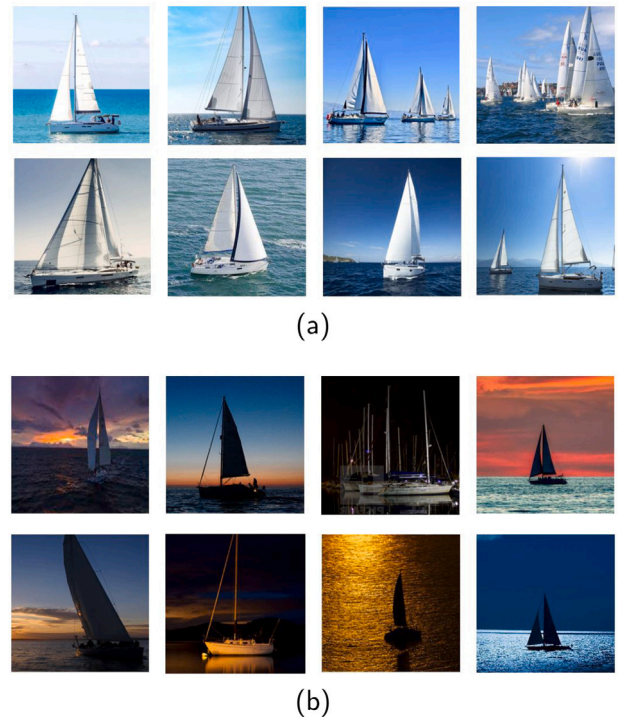


Fig. 7. Examples of real sailing boats images. (a) Daytime dataset. (b) Nighttime dataset.

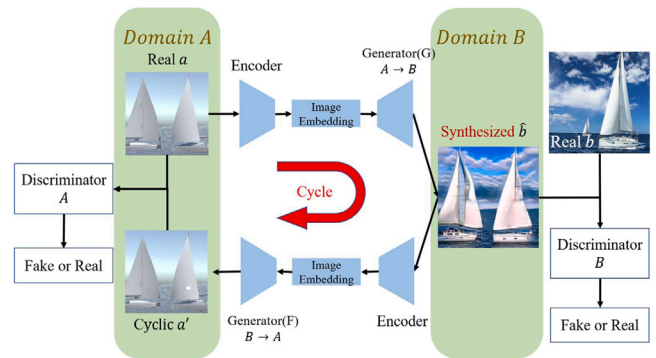


Fig. 8. Structure of CycleGAN with information flow from Domain A to Domain B and then back to Domain A.

image in Dataset *A* is denoted as *a*, and the real image in Dataset *B* is denoted as *b*. A generator that takes a simulation image as input and outputs a visually realistic image is desired, i.e. $G(a) = b'$, $a \in A$ and $(\cdot)'$ denotes a data sample generated from a generator; and another generator that can take a real image as input and translate that image into simulation style should also be constructed, i.e. $F(b) = a'$, $b \in B$. To achieve this, two discriminators are needed to determine if the generated images are close enough to the original style.

As shown in Fig. 8, Discriminator D_A receives simulation images sampled from Dataset *A* and the fake generated simulation images from the Generator *F* and determines if an image is generated or comes from Dataset *A*. Similarly, Discriminator D_B receives real images sampled from Dataset *B* and the fake generated real images from the Generator *G* and determines if an image is generated or comes from Dataset *B*. If the generator produces an image that is not similar enough to an image in the dataset, the discriminator will assign it with a low score (with a specified minimum of 0), and conversely, if the image looks like an image in the dataset, the discriminator should reward it with a high score (with a specified maximum of 1). Hence, the

discriminator should always give an image from the dataset a high score. Therefore, as with other GANs, the generators of a CycleGAN should try to maximize the discriminator scores of the generated image, which can be mathematically formulated as minimizing Adversarial Losses.

3.4. Loss functions

3.4.1. Adversarial loss

The adversarial losses are applied to the generators in both directions from A to B and from B to A . The total adversarial loss for CycleGAN is the sum of the losses in both directions and can be expressed as (Zhu et al., 2017):

$$\begin{aligned} \text{Loss}_{GAN} &= \mathcal{L}_{GAN}(G, D_B, A, B) + \mathcal{L}_{GAN}(F, D_A, A, B) \\ &= E_{b \sim B} [\log D_B(b)] + E_{a \sim A} [\log (1 - D_B(G(a)))] \\ &\quad + E_{a \sim A} [\log D_A(a)] + E_{b \sim B} [\log (1 - D_A(F(b)))] \end{aligned} \quad (3)$$

The parameters of the discriminators, D_A and D_B , are fixed when training the generator. The values of $\log D_B(b)$ and $\log D_A(a)$ are unchanged and as a and b are from the training sets, their discriminator scores should be close to 1. Hence, the objective of minimizing the adversarial losses becomes maximizing the discriminator scores for the generated images by both generators. The objective can be written as

$$G^* = \arg \min_G \max_{D_B} \mathcal{L}_{GAN}(G, D_B, A, B) \quad (4)$$

$$F^* = \arg \min_F \max_{D_A} \mathcal{L}_{GAN}(F, D_A, A, B) \quad (5)$$

Inversely, when training the discriminator, the total adversarial loss should be maximized. This is because the lower the score given to the fake images, the better the discriminator. Therefore, for the discriminators, the objective can be expressed as:

$$D_A^* = \arg \max_F \min_{D_A} \mathcal{L}_{GAN}(F, D_A, A, B) \quad (6)$$

$$D_B^* = \arg \max_G \min_{D_B} \mathcal{L}_{GAN}(G, D_B, A, B) \quad (7)$$

3.4.2. Cycle consistency loss

The cycle consistency loss (Zhu et al., 2017) is a $L1$ Loss to describe the difference between the original image and the reconstructed image after the original image been transferred into *DomainB* by the generator G and transferred back to *DomainA* by the generator F . This loss term encourages the generators to synthesize images with no change in their contents before and after the domain transfer, i.e. $F(G(a)) = a$. The mathematical expression of the cycle consistency loss is:

$$\mathcal{L}_{\text{cycle}} = E_{a \sim A} [\|F(G(a)) - a\|_1] + E_{b \sim B} [\|G(F(b)) - b\|_1] \quad (8)$$

If the cycle consistency loss is minimized to zero, this would generate an ideal result as:

$$F(G(a)) = a, G(F(b)) = b \quad (9)$$

Applying this loss would guarantee the contents of the images to be mostly unchanged during style translation. For example, in the cycle of translating an image in *DomainA* to *DomainB* and then back to *DomainA* as shown in Fig. 8, the only premise for $F(G(a)) = a$ to hold is that $G(a) \approx a$ which means the contents of $G(a)$ should be generally consistent with that of a , as the generator cannot reconstruct a from $G(a)$ if a and $G(a)$ are unrelated. Hence, the cycle consistency loss ensures that the content of the images generated by the generator remains largely the same.

3.4.3. Identity loss

CycleGAN also introduces a third loss term, i.e. identity loss. The idea about identity loss comes from that ideally, if an image in *DomainA* is transferred into the same domain by the generator F , the synthesized image should be exactly identical to the input image, which can be expressed as:

$$G(b) = b, F(a) = a \quad (10)$$

The identity loss is also $L1$ Loss just like the cycle consistency loss and is given as:

$$\mathcal{L}_{\text{identity}}(G, F) = E_{b \sim B} [\|G(b) - b\|_1] + E_{a \sim A} [\|F(a) - a\|_1] \quad (11)$$

3.4.4. Full objective

In total, there are four terms in the final loss function of CycleGAN, two adversarial losses, one cycle consistency loss and one identity loss. According to the functions of those losses, their contributions and importance are different. Factors λ_1 and λ_2 are introduced to change the contribution of each loss to the total loss. The full loss function becomes:

$$\begin{aligned} \mathcal{L}(G, F, D_A, D_B) &= \mathcal{L}_{GAN}(G, D_B, A, B) \\ &\quad + \mathcal{L}_{GAN}(F, D_A, B, A) \\ &\quad + \lambda_1 \mathcal{L}_{\text{cycle}}(G, F) \\ &\quad + \lambda_2 \mathcal{L}_{\text{identity}}(G, F) \end{aligned} \quad (12)$$

The actual values of λ_1 and λ_2 are 10 and 0.5 respectively. A larger λ_1 will give better reconstruction loss, but the style of the image might not be completely translated into another domain and there would only be small changes in the translated images. A smaller λ_1 will allow the model to make significant style changes to the translated images but will also cause unnecessary artefacts. The value of λ_2 has less impact on the performance of the model and it prevents artefacts.

4. Implementation details

All the training, testing, validation and evaluation work done in this paper were carried out on Google Colaboratory, using Tesla V100-SXM2-16 GB GPU.

4.1. Architecture of generator and discriminator

The architecture of ShipGAN is based on the CycleGAN (Zhu et al., 2017). Fig. 9(a) shows a typical architecture of a generator. In the first half of the structure is an auto-encoder containing two convolutional layers which are followed by a batch normalization layer and an activation function. The convolutional layer will downsample the input spatial dimension by a factor of 2 and the number of channels is multiplied by a factor of 2 by the second convolutional layer. In the centre of the architecture, there is a residual block that contains two 3×3 convolutional layers with 256 filters on each layer for a 256×256 input image. The final part of this architecture will upsample the image by a factor of 2 until the output image has the same size as the input image.

The architecture of a discriminator is illustrated by Fig. 9(b). The discriminator has a similar architecture as the convolutional neural network(CNN) for classification. This type of discriminator is named PatchGAN (Isola et al., 2017). After inputting an image to the PatchGAN, it will downsample the image by four or more convolutional layers followed by batch normalization layers and activation functions. The output of the PatchGAN would be the probability of the input image being real and being fake.

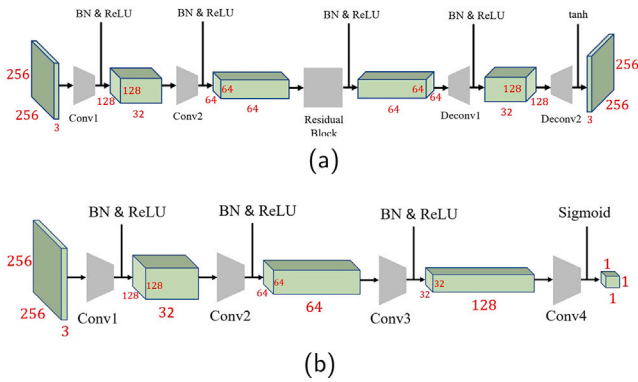


Fig. 9. Typical structure of CycleGAN's (a) generator and (b) discriminator.

Table 1

Training parameters for Sim2Real Model.

n_epoch	n_epoch_decay	ngf	ndf	lr
300	200	128	128	0.0003

Table 2

Training parameters for Day2Night Model.

n_epoch	n_epoch_decay	ngf	ndf	lr
1600	400	128	128	0.0003

4.2. ShipGAN training

4.2.1. Sim2Real CycleGAN

The simulation images for training and validation are extracted from the videos of Scene1 to Scene4. The ratio of the number of training images and that of the validation images is 1:4. In other words, extracting one frame every five frames as the training data and the rest images are for testing. Scene5 does not take part in the training set so it can be used to test the sim2real model trained. When training the network, the parameters chosen for training the Sim2Real model are given by Table 1.

n_epoch and n_epoch_decay are the number of epochs with the initial learning rate and the number of epochs to linearly decay learning rate to zero respectively, and lr is the initial learning rate (Junyanz, 2019). ngf is the number of generator filters in the last convolutional layer and ndf is the number of discriminator filters in the first convolutional layer. These two parameters determine the depth of the network. What is more, the frequency for saving checkpoints and training outputs is set to every epoch so that the training of each epoch can be monitored.

4.2.2. Day2Night CycleGAN

The training set of daytime images contains the synthesizing images generated by the sim2real CycleGAN from the images of Scene1 to Scene4 and also the real daytime sailing boat images, while night images are all from the real domain. The synthesizing images translated from simulation images of Scene5 do not take part in the training set so that it can validate the day2night model trained. When training the Day2Night model, the parameters are set as shown in Table 2.

5. Results and discussions

In this section, the results of image translation using ShipGAN are presented including: (1) from the simulation domain to the realistic domain and (2) from the daytime domain to the night domain. In order to demonstrate the reliability of the GAN-generated data for training and validating the perception systems for USVs, object detection, instance segmentation, and semantic segmentation algorithms are applied to the synthesized images, respectively.

5.1. Evaluation metrics

To test if the generated images are real enough to fool the object detection and image segmentation algorithms, the pre-trained models trained by the COCO dataset (Lin et al., 2014) were used to detect the sailing boats in the generated images.

5.1.1. Object detection, and instance and semantic segmentation

Semantic segmentation is a form of image segmentation that combines object detection, object classification, localization and segmentation tasks. Essentially, the task of semantic segmentation can be referred to as image classification at the pixel level. The goal of semantic segmentation is to take an image and generate an output such that it contains a class label map where the pixel value of the input image is transferred into a class label value. Instance Segmentation is also a form of image segmentation and unlike semantic segmentation, instance segmentation deals with both detecting objects and demarcating their boundaries.

These two methods for image segmentation are widely used in real-world scenarios such as autonomous driving systems and medical imaging. Therefore, they are suitable for evaluating the GAN-generated images of ships. If the fake ships can be detected and segmented from the background and other objects by instance and semantic segmentation model trained by images containing real ships, it means that the synthetic images can be used in the training of the visual perception systems of autonomous ships. The performance of the generated images on object detection will be tested by applying Yolov4 Object Detection (Bochkovskiy et al., 2020) and Detectron2 (Pham et al., 2020).

5.2. Simulation images to realistic images

5.2.1. Training

The upper rows of Fig. 10 are the unity simulation images in the training set and the lower rows are the corresponding translated images in the realistic domain. The figures on the left-hand side illustrate the motions of the two sailing boats in the four scenes. The model used to generate the results was trained for 1953 epochs and for about 361,000 steps.

The results indicate that after 1953 epochs of training, the network has learned the features of the sky with clouds and the ocean. For the moving sailing boats over frames, the features of the hulls and the sails were extracted both in the simulation and the real images, so that the corresponding areas can be transferred into the other domain. More importantly, when translating the frame with two boats meeting and one being partially blocked by the other, the two boats could still be distinguished from each other. Besides, some more detailed features in the real domain had also been extracted. Firstly, in addition to the hulls and the sails, the sailing boats in the synthesized images also have decks, windows and even sailors onboard. Secondly, when the two boats overlap, the shadow of the boat in front falls on the boat being blocked. Another detail to note is the reflections of the sailing boats on the sea. All these features were extracted from the sailing boats in the real images, as in the simulation images there are only objects without texture and the reflection and refraction option of the wave was turned off.

In Fig. 11, the loss curves of training the CycleGAN to translate simulation images into realistic images indicate that the training process has converged and the identity loss and consistency loss have been minimized. The adversarial loss does not converge as the generator and the discriminator are improving simultaneously.

5.2.2. Validation

Fig. 12 are the testing results and their corresponding input simulation images. From the results, it can be seen that the synthesized images contain all the key features of the realistic training images as expected, as the input images for testing belong to the same scenes of the training set, and the only difference is the positions of the sailing boats.

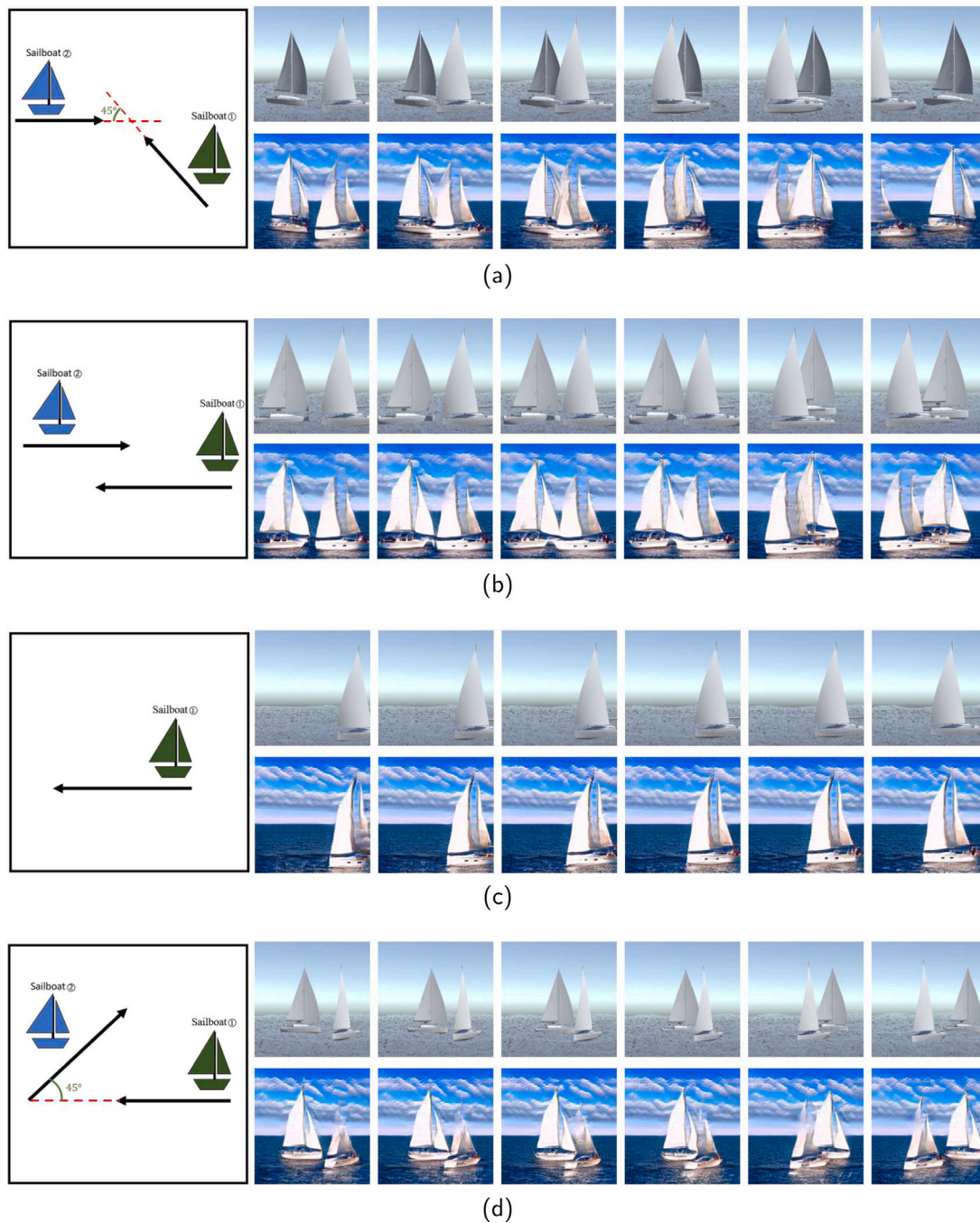


Fig. 10. Training results of Scenes 1–4 images. (a) Scene 1. (b) Scene 2. (c) Scene 3. (d) Scene 4.

5.2.3. Generating real images

As the simulation images of Scene5 were totally held back from the training process, it can be used to validate the trained model to see if it is a good fit for the Unity simulation environment containing sailing boat models. Fig. 13 shows the result of the validation set of data. In Scene5, sailing boat 1 is undertaking the same motion as in Scene4, while the angle between the courses of sailing boat 1 and sailing boat 2 is 60 degrees rather than 45 degrees.

Sailing boat 2 was translated into the real sailing boat as expected, and sailing boat 2 was also translated into the realistic domain with all the key features of a real sailing boat. This testing result indicates that the trained CycleGAN model fits well to translate the sailing boat models in the Unity simulation environment to the realistic domain.

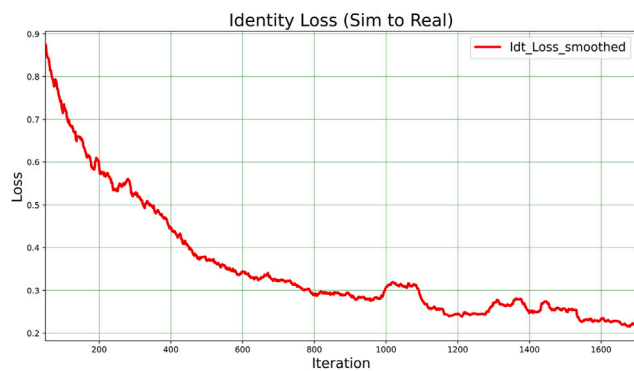
5.2.4. Object detection, instance and semantic segmentation

To further validate the reliability of the GAN-generated images in the visual perception algorithm, the object detection and image segmentation algorithms are applied to the synthesized images, as in most autonomous systems, the input images are pre-processed by objection detection and image segmentation techniques. The Yolov4 Objective Detection algorithm (Bochkovskiy et al., 2020) is used to detect the fake ships in the generated images and Detectron2 (Pham et al., 2020) is used for instance and semantic segmentation. In Fig. 14, each of the four rows of images presents the result of detection and segmentation for one frame from the translated Scene5, Scene4, Scene1 and Scene3 respectively.

The instance maps were produced according to the results of instance segmentation, in which one colour represents one instance by



a



b







c

Fig. 11. CycleGAN training losses (from simulation to real). (a) Adversarial loss. (b) Identity loss. (c) Cycle-consistency loss.

the value of its ID. The instance segmentation results of the four synthesized image examples in Fig. 14 indicate that the fake sailing boats in the synthesized images can be detected by the instance segmentation algorithm trained by the COCO Dataset. In other words, the key features of the translated sailing boats are close to those features of real boats, so the translated boats can be detected as real boats. In the same way, the semantic segmentation results in Fig. 14 show that translated sailing boats can be correctly classified as boats, as well as the sky and the sea in the synthesized images were also classified accurately. Similarly, the

Table 3

Maximum confidence scores for detecting sailing boats in the synthesized images.

Synthesized image	Instance Segmentation		Semantic Segmentation		Yolov4	
	Ship1	Ship2	Ship1	Ship2	Ship1	Ship2
	0.92	0.98	0.91	0.98	0.92	0.76
	0.67	0.98	0.98	0.98	0.93	0.52
	0.99	0.57	0.99	0.88	0.97	0.96
	0.85		0.97		0.91	

fake boats in the synthesized images can also be detected as boats by the Yolov4 object detector trained by the COCO Dataset.

To further investigate how confident the detection algorithms were about the detected object being a real boat, the confidence scores were recorded. Table 3 states the maximum confidence scores when detecting the two sailing boats in the four synthesized image examples. In this confidence score table, all the maximum confidence scores for both sailing boats are greater than 50% after applying the non-max suppression algorithm.

After synthesized images were tested for the visual perception algorithms, they were then brought into the real application of object detection and tracking. Fig. 15 presents the result of applying object tracking using the SORT tracking algorithm on the generated real images (Bewley et al., 2016). In Fig. 15, the green bounding box indicates the Yolov4 detection result and the blue bounding box indicates the prediction of the motion of the sailing boat in this frame based on the detection result in the previous frames.

5.2.5. Analysis and discussion

Overall, the trained Sim2Real cycleGAN model can accurately translate a Unity simulation image which contains sailing boats moving in any direction into an image in the realistic domain. Also, the sky, sea and sailing boats in the synthesized images can be detected by the visual perception systems trained by the real image dataset in the vast majority of cases. However, over-classification might happen in the instance segmentation, i.e one synthesized sailing boat will sometimes be recognized as two separate instances as shown in Fig. 14. One possible factor that could cause this over-classification is the gap between the sail and the hull of the sailing boat. The lack of connection between the hull and the sail may cause the two parts to be classified into two instances. What is more, it is noticed that there is a gap between the two sails of the synthesized sailing boat, while there is not such a gap in the Unity sailing boat model. The cause of this gap might be the gap between the sails of the real sailing boats in the training images, as in the real world, the sailing boats are driven by the wind which can cause the sails to flutter and this will result in a gap between the sails of a sailing boat.

5.3. Daytime images to night images

5.3.1. Training

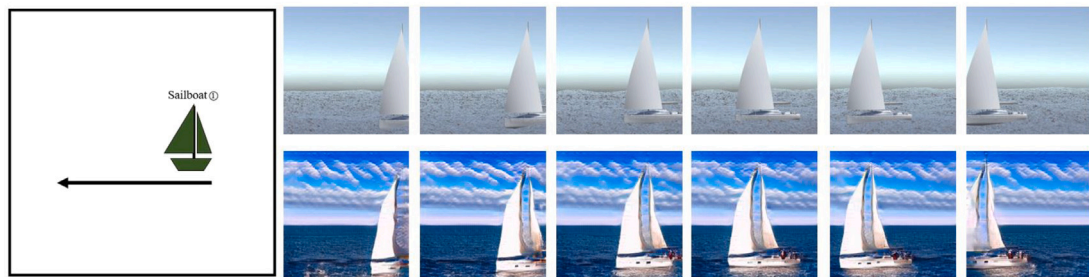
Fig. 16 presents the daytime images, which are synthesized by the trained Sim2Real CycleGAN presented in the previous section, and



(a)



(b)



(c)



(d)

Fig. 12. Validation results of Scenes 1–4 images. (a) Scene 1. (b) Scene 2. (c) Scene 3. (d) Scene 4.

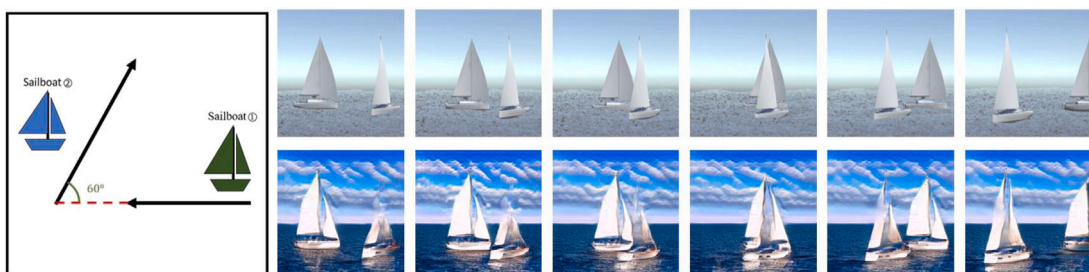
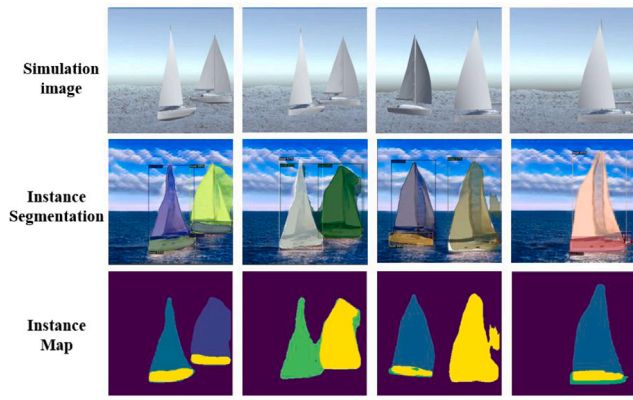
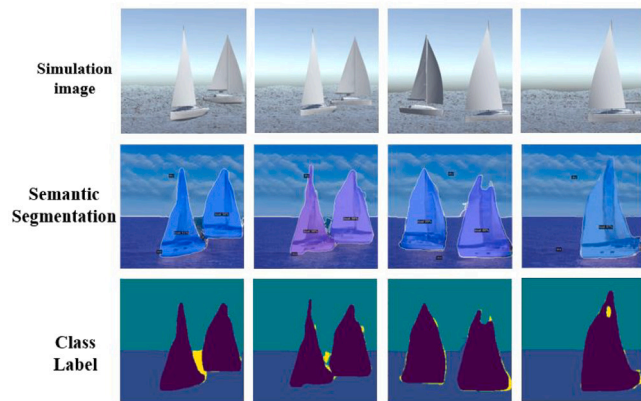


Fig. 13. Synthesized real domain images by translating Scene5 simulation images.



(a)



(b)



(c)

Fig. 14. Results of applying (a) Instance segmentation, (b) Semantic segmentation, and (c) Yolov4 object detection on the synthesized images.

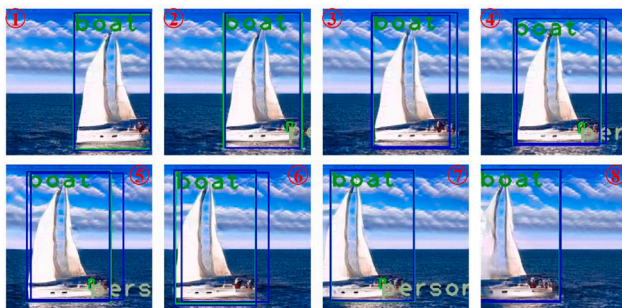


Fig. 15. Bounding box of Sort object tracking on the sequenced synthesized images.

the corresponding night images generated by the trained Day2Night CycleGAN. By inspection, the sailing boats in the synthesized night images have hulls, sails and also their reflections on the sea. What is more, the sea level in the night images is clearly visible, although the intensity of the synthesized images is low and the background is dark.

Fig. 17 gives the real daytime images for training and the corresponding translated night images. The images on the upper row are the input daytime images and the images below are the corresponding synthetic night images. For the images in the realistic domain, the contents of the images were kept unchanged and there was only a change in tone from day to night.

The results are visually realistic and Fig. 18, the losses of the training process of Day2Night CycleGAN, indicates that the training process has converged.

5.3.2. Validation

In Fig. 19, the upper rows of images are the synthesized images by the Sim2Real cycleGAN from Scene1 to Scene4 but are held back from the training process, and the lower rows of images are the corresponding synthesized night images. As expected, the daytime images are translated into night images with the key features that make sailing boats visually realistic.

Fig. 19 shows the validation results of converting the images in the real daytime domain into the night domain. Also as expected, the contents are kept unchanged and only the tone was changed from daytime style to night style, which is the same as the results of the training process. This result indicates that the trained Day2Night fits well with the synthesized Sim2Real images of Scene1 to Scene4 (see Fig. 20).

5.3.3. Generated night images

After validation, the Sim2Real synthesized images of Scene5, which were totally held back from training, were input to the trained Day2Night CycleGAN and Fig. 21 presents those Sim2Real images and the corresponding synthesized images in the night domain. From the results, it can be seen that the same as the training and validation results, the features of the sailing boats in the Day2Night synthesized images match those corresponding features of the sailing boats in the Sim2Real synthesized images in the daytime domain, such as the hull, sails and the reflections on the sea. These results indicate that the trained Day2Night CycleGAN model fits well with the Sim2Real synthesized images even with those which have been held back from training.

5.3.4. Yolov4 object detection

To investigate the reliability of the synthesized Night Domain images for the visual perception systems, Yolov4 object detection algorithm was applied to the synthesized images. Fig. 22 shows the object detection outcomes by applying Yolov4 object detector to the translated night images in the realistic domain. Most fake sailing boats can be detected as real boats by the Yolov4 detector, and this indicates that the trained Day2Night model keeps most contents unchanged. The maximum confidence scores of the sailing boat detection in the synthesized night images are above 70%. This result once again confirms that the key features of the translated sailing boats in the synthesized images can fool the Yolov4 object detector trained by the COCO dataset into classifying the fake sailing boats as real ones.

Fig. 23 shows that in the synthesized night images translated from the Sim2Real daytime images, some generated sailing are not recognized as real boats.

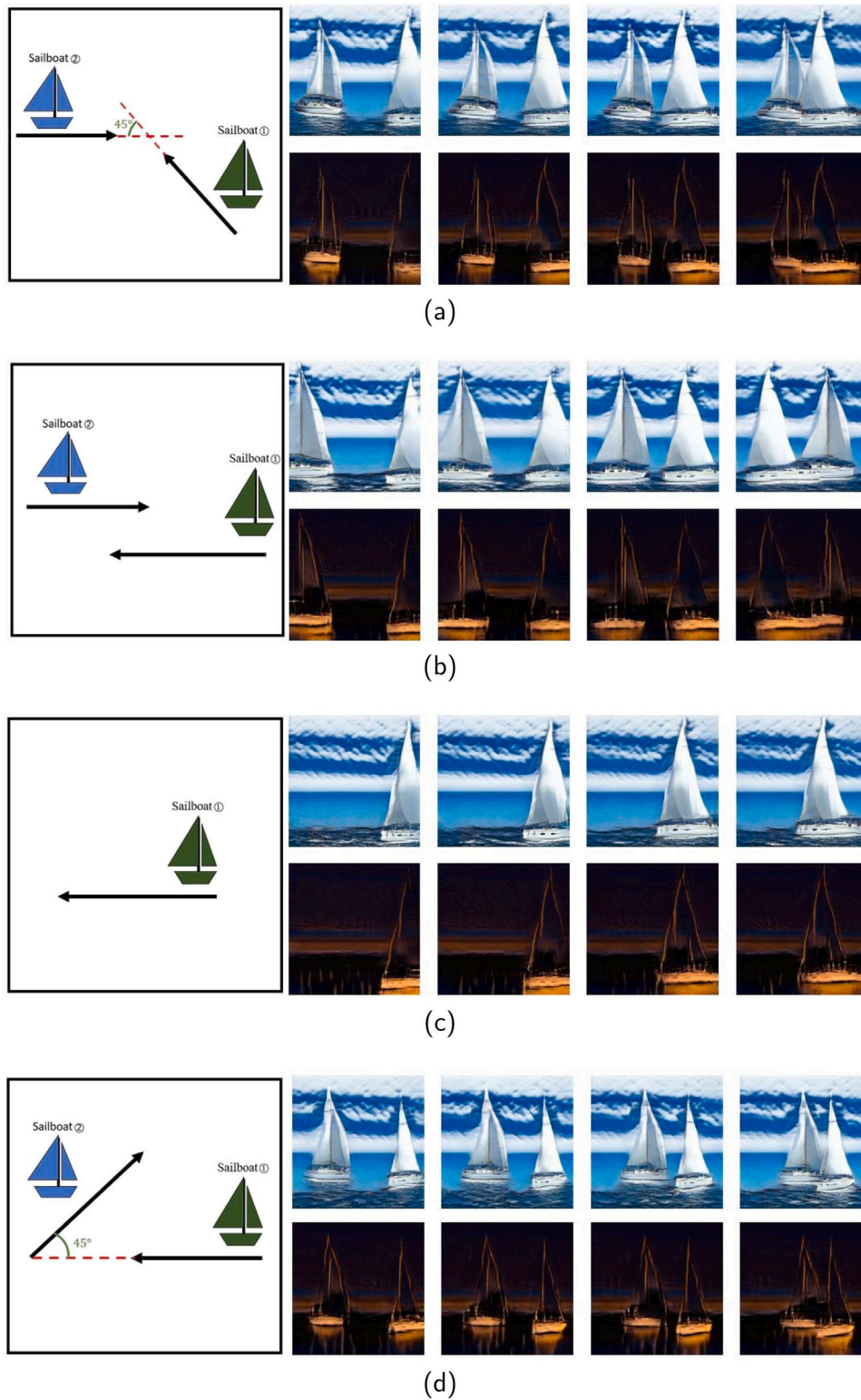


Fig. 16. Training results of Scenes 1–4 night images. (a) Scene 1. (b) Scene 2. (c) Scene 3. (d) Scene 4.

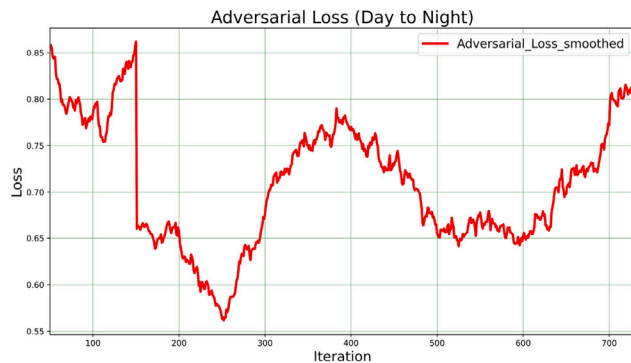
5.3.5. Analysis and discussion

In general, the daytime images in both the realistic domain and the Sim2Real synthesized daytime domain can be translated into the night domain by the trained Day2Night CycleGAN with all the key features of the sailing boats. The reliability of synthesized night images was proved, as the fake sailing boats in the synthesized images can be classified as real boats by the Yolov4 object detector. However, there

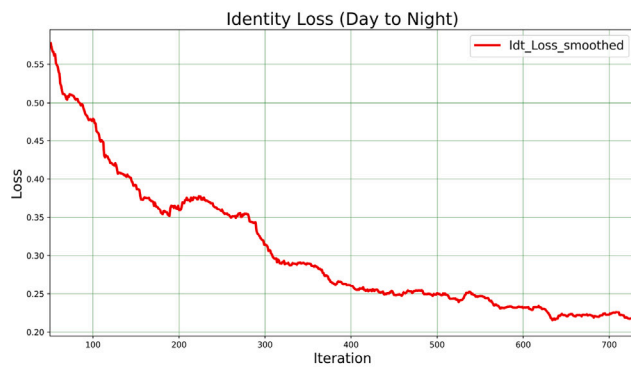
are still some failure cases in which the generated sailing boats cannot be detected, although visually there is no large difference between the sailing boats that can be detected and those that cannot. This might be caused by the lack of effective features of a real sailing boat in the synthesized sailing boats, and some of those features are imperceptible to the human eye. What is more, in the night images translated from the Sim2Real daytime images, the white hulls are orange at night, while



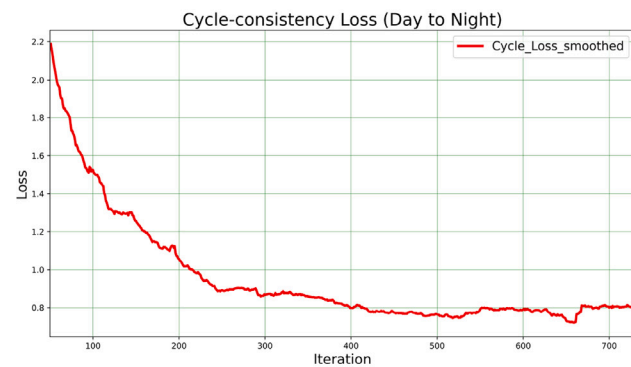
Fig. 17. Training results of real daytime images.



a



b



c

Fig. 18. CycleGAN training losses (from day to night).

the white sails are in the same colour as the night sky but still retain their outlines. The reason could be that in the training set, some sailing boats at night with the same tone will have their sails lowered and there are only hulls and masts left. Therefore, when translating the daytime images into the night domain, the trained model tried to fit the translated sailing boats to the real boats at night as similarly as possible by giving the sails the colour of the night sky to make them look like being lowered.

6. Conclusions and future works

This paper proposed a deep learning model, ShipGAN, based on the cycle-consistent adversarial network to translate simulation images into realistic images. Five scenarios containing sailing boats with all the physical effects modelled were created in the Unity simulation engine. Then, the frames extracted from the simulation videos and real images of ships were used to train the ShipGAN model. The trained models were validated and tested to generate the synthetic realistic images from the simulation images. Finally, the synthetic images were evaluated by visual perception algorithms for their fidelity and reliability. The results of the evaluation of the ShipGAN-generated realistic images show that the fake ships are able to be detected as real ships with high confidence scores (approximately 95%) and the sky, the ocean and the ships can be successfully segmented from each other by applying image segmentation algorithms trained by natural images. As the navigation and perception systems of the autonomous ships contain pre-processing using object detection and image segmentation, the evaluation result also means that the ShipGAN-generated images of ships can be used and are reliable for the training of autonomous vessels. What is more, the generated dataset containing busy water and complex scenarios is able to improve the ability of autonomous vessels in dealing with such unusual cases when they are operating in the real world.

In conclusion, by the proposed ShipGAN, the Unity simulation images of ships can be translated into realistic images that are not only visually realistic but also are tested to be reliable with the object detection and image segmentation algorithms. The ShipGAN can be seen as a bridge between the simulation domain and the realistic domain for images. By ShipGAN, the ships in images can be converted between these domains.

This paper mostly focused on investigating the feasibility of converting the simulation images into realistic images by using Generative Adversarial Networks, and only one type of ship is included in this paper. The proposed ShipGAN just opens the gateway leading to a new method to create maritime datasets. Therefore, there are still some works that can be done to improve the reliability of the synthesized images for autonomous USV systems training. Some future works can be suggested as:

- *Translating images with complex textures*: The range of boats could be more varied, for example, cruises and yachts, which contain more features to be extracted. However, the existing cycleGAN structure might not be sufficient for those types of ships with more complex textures. Therefore, inspired by the multidimensional structure of the discriminator of pix2pixHD, the generator and discriminator could be improved to be multidimensional as well to handle features on different scales, so that more ships with complex features can be generated to enrich the variety of ships in the generated dataset.
- *Generating images in a video sequence*: The current model treats every training image as an individual even if they belong to one

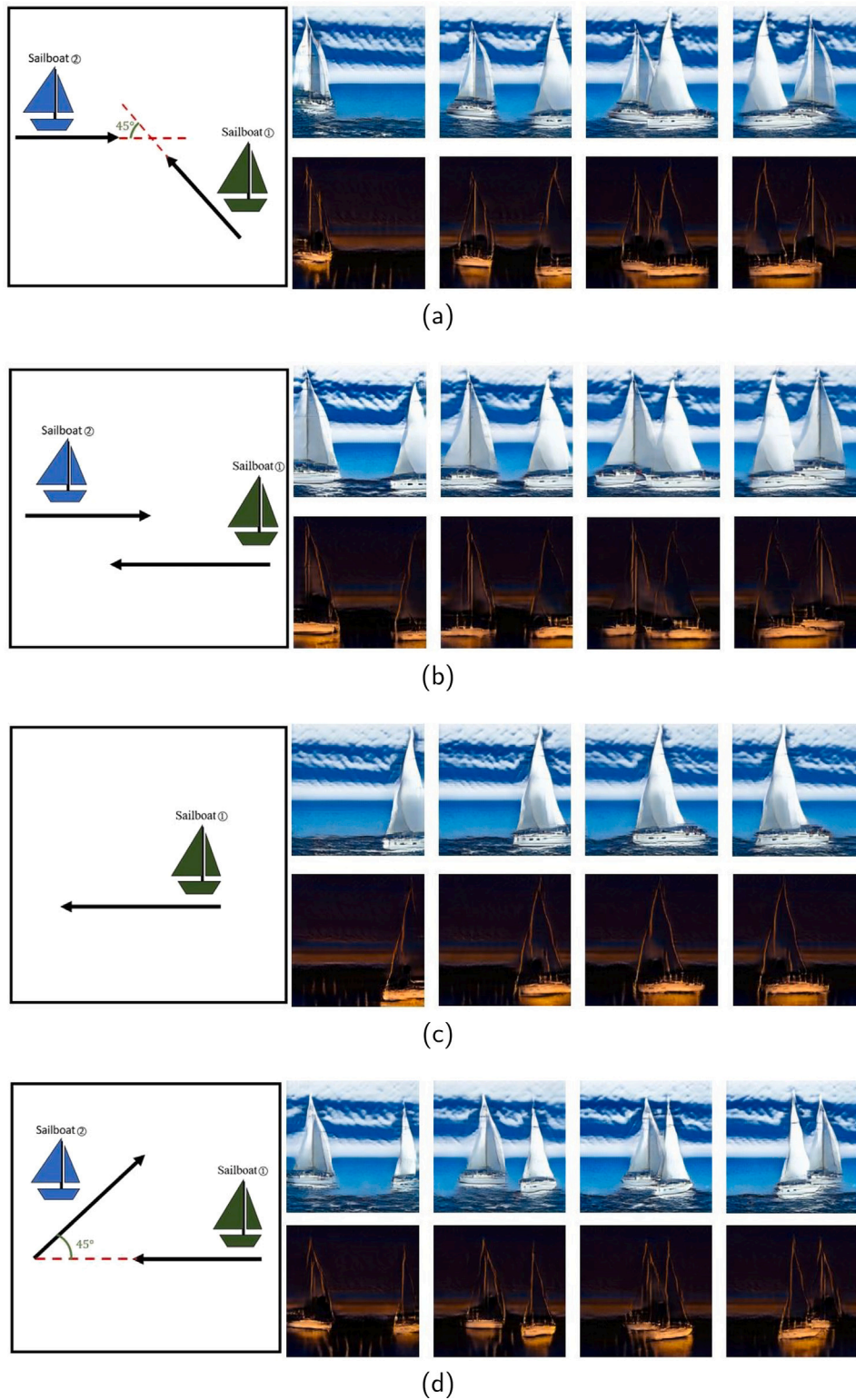


Fig. 19. Validation results of Scenes 1–4 night images. (a) Scene 1. (b) Scene 2. (c) Scene 3. (d) Scene 4.

video. This results in that when using synthetic images to create videos, some detailed features are not stable through the videos. Therefore, some improvements can be done to feed the image tensors to the network so that images in one video can be treated as a whole.

- *Learning the poses of the ships:* Another improvement that can make the motions of the ships more realistic in the synthetic images

is to combine another deep learning network that can learn the poses of the ships when moving. As mentioned before, the current approaches for image translation are mostly treating the images as individuals, and no pose estimation is included in the training process. This makes the motions of the ships in the synthetic videos not so physically realistic. By taking the pose information of the objects in the images into account as an additional feature



Fig. 20. Validation results of real daytime images.



Fig. 21. Synthesized night domain images by translating Scene5 images.



Fig. 22. Synthesized night domain images by translating real daytime images.

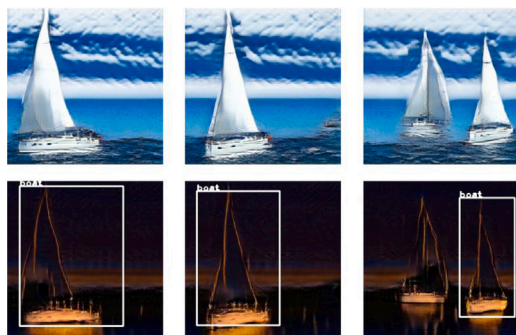


Fig. 23. Results of applying Yolov4 object detection on the generated night images.

to learn, the model would be able to generate images containing objects doing more physically realistic motions.

CRediT authorship contribution statement

Yuxuan Dong: Conceptualization, Methodology, Software, Investigation, Writing – original draft, Writing – review & editing. **Peng Wu:** Conceptualization, Supervision, Writing – original draft. **Sen Wang:** Visualization, Methodology, Writing – original draft. **Yuanchang Liu:** Methodology, Investigation, Supervision, Writing – original draft, Writing – review & editing, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

This research is funded by The Royal Society Research, United Kingdom Grant (RGS-R2-212343).

References

- Arruda, V.F., Paixão, T.M., Berriel, R.F., De Souza, A.F., Badue, C., Sebe, N., Oliveira-Santos, T., 2019. Cross-domain car detection using unsupervised image-to-image translation: From day to night. In: 2019 International Joint Conference on Neural Networks (IJCNN). IEEE, pp. 1–8.
- Bargshady, G., Zhou, X., Barua, P.D., Gururajan, R., Li, Y., Acharya, U.R., 2022. Application of CycleGAN and transfer learning techniques for automated detection of COVID-19 using X-ray images. *Pattern Recognit. Lett.* 153, 67–74.
- Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B., 2016. Simple online and realtime tracking. In: 2016 IEEE International Conference on Image Processing (ICIP). pp. 3464–3468. <http://dx.doi.org/10.1109/ICIP.2016.7533003>.
- Bochkovskiy, A., Wang, C.-Y., Liao, H.-Y.M., 2020. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.
- Bovcon, B., Muhovič, J., Perš, J., Kristan, M., 2018. Stereo obstacle detection for unmanned surface vehicles by IMU-assisted semantic segmentation. *Robot. Auton. Syst.* 104, 1–13.
- Ditzel, 2022. ditzel/UnityOceanWavesAndShip. URL: <https://github.com/ditzel/UnityOceanWavesAndShip>.
- Epic Games, 2022. Unreal Engine, URL: <https://www.unrealengine.com>.
- Haas, J.K., 2014. A history of the unity game engine.
- Hoffman, J., Wang, D., Yu, F., Darrell, T., 2016. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv preprint arXiv:1612.02649*.
- Isola, P., Zhu, J.-Y., Zhou, T., Efros, A.A., 2017. Image-to-image translation with conditional adversarial networks. In: *Computer Vision and Pattern Recognition (CVPR)*, 2017 IEEE Conference on.
- Jin, D., Zheng, H., Zhao, Q., Wang, C., Zhang, M., Yuan, H., 2021. Generation of vertebra micro-CT-like image from MDCT: A deep-learning-based image enhancement approach. *Tomography* 7 (4), 767–782.
- Junyanz, 2019. junyanz/pytorch-CycleGAN-and-pix2pix. URL: <https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix>.
- Karlsson, S., Welander, P., 2018. Networks for Image-to-Image Translation on Street View and MR Images (Ph.D. thesis). URL: <https://www.diva-portal.org/smash/get/diva2:1216606/FULLTEXT01.pdf>.
- Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context. In: *European Conference on Computer Vision*. Springer, pp. 740–755.
- Liu, M.-Y., Breuel, T., Kautz, J., 2017. Unsupervised image-to-image translation networks. *Adv. Neural Inf. Process. Syst.* 30.
- Machiraju, H., Balasubramanian, V.N., 2020. A little fog for a large turn. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. pp. 2902–2911.
- Marketing, F., Communication, 2022. WAM-V 16 ASV, URL: <https://geo-matching.com/usvs-unmanned-surface-vehicles/wam-v-16-asv>.
- Matsui, T., Ikehara, M., 2021. GAN-based rain noise removal from single-image considering rain composite models. In: 2020 28th European Signal Processing Conference (EUSIPCO). IEEE, pp. 665–669.
- Motian, S., Piccirilli, M., Adjeroh, D.A., Doretto, G., 2017. Unified deep supervised domain adaptation and generalization. In: *Proceedings of the IEEE International Conference on Computer Vision*. pp. 5715–5725.
- News Isle Wight, 2013. New navigation rules for Cowes Harbour. URL: <https://onthewight.com/new-navigation-rules-for-cowes-harbour/>.
- Peng, X., Saenko, K., 2018. Synthetic to real adaptation with generative correlation alignment networks. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE, pp. 1982–1991.
- Peng, X., Usman, B., Saito, K., Kaushik, N., Hoffman, J., Saenko, K., 2018. Syn2real: A new benchmark for synthetic-to-real visual domain adaptation. *arXiv preprint arXiv:1806.09755*.
- Pham, V., Pham, C., Dang, T., 2020. Road damage detection and classification with detectron2 and faster r-cnn. In: 2020 IEEE International Conference on Big Data (Big Data). IEEE, pp. 5592–5601.
- Prasad, D.K., Rajan, D., Rachmawati, L., Rajabally, E., Quek, C., 2017. Video processing from electro-optical sensors for object detection and tracking in a maritime environment: A survey. *IEEE Trans. Intell. Transp. Syst.* 18 (8), 1993–2016.
- Saito, K., Kim, D., Sclaroff, S., Darrell, T., Saenko, K., 2019. Semi-supervised domain adaptation via minimax entropy. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 8050–8058.
- Sun, B., Saenko, K., 2016. Deep coral: Correlation alignment for deep domain adaptation. In: *European Conference on Computer Vision*. Springer, pp. 443–450.

Xu, W., Souly, N., Brahma, P.P., 2021. Reliability of gan generated data to train and validate perception systems for autonomous vehicles. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 171–180.

Zaher, G., 2020. Simulating Weather Conditions on Digital Images (Ph.D. thesis).

Zhu, J.-Y., Park, T., Isola, P., Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Computer Vision (ICCV), 2017 IEEE International Conference on.