



โครงการ

การเรียนการสอนเพื่อเสริมประสบการณ์

ชื่อโครงการ	การลบเงาของเครื่องบินโดยอัตโนมัติจากภาพถ่ายระยะไกลโดยใช้ Mask-ShadowGAN Automatic Aircraft Shadow Removal from Remote Sensing Images using Mask-ShadowGAN		
ชื่อนิสิต	นายศิรปวีณ์	กันยาพรกุล	603 36626 23
ภาควิชา	คณิตศาสตร์และวิทยาการคอมพิวเตอร์ สาขาวิชาวิทยาการคอมพิวเตอร์		
ปีการศึกษา	2563		

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

การลบเงาของเครื่องบินโดยอัตโนมัติจากรูปภาพพระยะไกลโดยใช้ Mask-ShadowGAN

นายศิริปวีณ์ กัญยาพรกุล

โครงการนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต
สาขาวิชา วิทยาการคอมพิวเตอร์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2563
ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

Automatic Aircraft Shadow Removal from Remote Sensing Images using
Mask-ShadowGAN

Sirapavee Ganyaporngul

A Project Submitted in Partial Fulfillment of the Requirements
for the Degree of Bachelor of Science Program in Computer Science
Department of Mathematics and Computer Science
Faculty of Science
Chulalongkorn University
Academic Year 2020
Copyright of Chulalongkorn University

นายศิริปวีณ์ กัญญาพรกุล: การลบเงาของเครื่องบินโดยอัตโนมัติจากรูปภาพระยะไกลโดยใช้ Mask-ShadowGAN. (Automatic Aircraft Shadow Removal from Remote Sensing Images using Mask-ShadowGAN) อ.ที่ปรึกษาโครงการหลัก: รองศาสตราจารย์ ดร.นกุล คุณะโรจนานนท์, 46 หน้า.

โครงการวิจัยในชั้นเรียน เรื่อง “การลบเงาของเครื่องบินโดยอัตโนมัติจากรูปภาพระยะไกลโดยใช้ Mask-ShadowGAN” มีวัตถุประสงค์ คือ นำเสนอวิธีการใหม่ในการลบเงาของเครื่องบิน ซึ่งเราได้ใช้รูปภาพถ่ายทางไกลในโครงการนี้ โดยวิธีการนี้จะเรียนรู้ที่จะลบเงาโดยอัตโนมัติโดยการใช้รูปภาพที่มีเงาและปราศจากเงา Mask-ShadowGAN แตกต่างจากอัลกอริทึมอื่น ๆ คือ framework นี้ไม่จำเป็นต้องใช้รูปภาพที่เหมือนกันในการ train โมเดล (รูปที่มีเงาสามารถแตกต่างจากรูปที่ปราศจากเงาได้ เช่น แตกต่างกันในโมเดลของเครื่องบินหรือพื้นหลัง) คุณสมบัตินี้ทำให้ผู้ใช้ทำการรวบรวมข้อมูลได้ง่ายมากขึ้น framework นี้ใช้ Pytorch ในการพัฒนาโดยใช้ภาษา Python ในการเขียน จากรูปผลลัพธ์ ผู้เขียนสังเกตได้ว่ารูปภาพปราศจากเงาที่โมเดลสร้างขึ้นนั้นมีเงาที่เบาบางลง แต่ยังคงรูปทรงของเครื่องบินไว้ได้ ผู้เขียนทำการประเมินผลลัพธ์โดยใช้ Root-Mean-Square-Error กับรูปภาพปราศจากเงาที่โมเดลสร้างขึ้น และ Jaccard Index กับรูปภาพรูปทรงของเครื่องบินแบบ Binary ที่สร้างจากรูปภาพปราศจากเงาที่โมเดลสร้างขึ้นโดยใช้เทคนิคทางการประมวลผลภาพในการสกัดรูปทรงของเครื่องบินออกมา หลังจากนั้นทำการเปรียบเทียบผลลัพธ์กับ framework อื่นโดยใช้ Jaccard Index คะแนนของ Mask-ShadowGAN คือ 0.7799 ในขณะที่ Modified DSS ซึ่งเป็นโมเดลแบบ CNN อีกรูปแบบหนึ่งที่ใช้เทคนิค post-processing กับ L2-Normalization ทำคะแนนได้ 0.7755 ซึ่ง Mask-ShadowGAN ทำคะแนนได้ดีกว่า Modified DSS

ภาควิชา.....คณิตศาสตร์และวิทยาการคอมพิวเตอร์.....ลายมือชื่อนิสิต.....
 สาขาวิชา.....วิทยาการคอมพิวเตอร์.....ลายมือชื่อ อ.ที่ปรึกษาโครงการหลัก.....
 ปีการศึกษา.....2563.....

6033662623: MAJOR COMPUTER SCIENCE

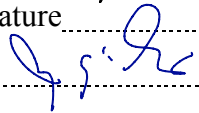
KEYWORDS: DEEP LEARNING / REMOTE SENSING IMAGE /

MASK-SHADOWGAN / SHADOW REMOVAL

SIRAPAVEE GANYAPORNGUL: AUTOMATIC AIRCRAFT SHADOW
REMOVAL FROM REMOTE SENSING IMAGES USING MASK-
SHADOWGAN. ADVISOR: ASSOC. PROF. NAGUL COOHAROJANANONE,
Ph.D. 46 pp.

The topic of the project is Automatic Aircraft Shadow Removal from Remote Sensing Images using Mask-ShadowGAN. The objective of this research is to introduce a new method to remove the shadow in airplane images. We use remote sensing images in this work. The new procedure learns to remove the shadow automatically by using shadow and shadow-free images. Unlike other algorithms, Mask-ShadowGAN requires no identical training image set. This aspect eases the data collection process for the user. The framework develops in a Python environment using PyTorch. We can see that the generated shadow-free images have shadows thinned or faded out, but the airplane shape is still intact. We then evaluate the results using Root-Mean-Square-Error with generated shadow-free images and Jaccard Index with their binary images. The binary images are obtained through a custom image processing technique. Our score is 0.7799, while the Modified DSS, which is traditional CNN with post-processing and L2-Normalization, the score is 0.7775, which the Mask-ShadowGAN performs better than the Modified DSS.

Department: Mathematics and Computer Science Student's Signature 

Field of Study: Computer Science Advisor's Signature 

Academic Year: 2020

ACKNOWLEDGEMENTS

I wish to express my deepest gratitude to the people who give advice and help complete this project.

First and foremost, I would like to thank whole-heartedly my advisor, which is Assoc. Prof. Nagul Cooharajanone, who gives me invaluable advice and helps me check and complete this project. This work cannot be published on ICIEA 2021 without the guidance and motivation he gives to me.

I wish to thank Mr. Pravee Kruachottikul, who gives me access to his computational resource. This work cannot finish without his support.

I would also like to thank Mr. Donnaphat Trakulwaranont, who give me advice on choosing the project topic.

Finally, I would like to thank my parents for hearten me when I'm feeling discouraged while giving me wise counsel and hear me for everything.

CONTENTS

	Page
ABSTRACT IN THAI	i
ABSTRACT IN ENGLISH.....	ii
ACKNOWLEDGEMENTS	iii
CONTENTS	iv
LIST OF TABLES	vi
LIST OF FIGURES.....	vii
CHAPTER I INTRODUCTION	1
1.1 Background and Rationale	1
1.2 Objectives.....	2
1.3 Scope.....	2
1.4 Project Activities	3
1.5 Benefits.....	3
1.6 Report Outlines	4
CHAPTER II LITERATURE REVIEW	5
2.1 Deeply Supervised Salient Object Detection with Short Connections	5
2.2 Aircraft Segmentation from Remote Sensing Images using Modified Deeply Supervised Salient Object Detection with Short Connection	5
2.3 Mask-ShadowGAN: Learning to Remove Shadows from Unpaired Data	6
CHAPTER III METHODOLOGY	7
3.1 Learning to Remove Shadow from Unpaired Data.....	7
3.2 Code Explanations.....	9
3.3 Evaluation Method	14
3.3.1 LAB color space.....	15

3.3.2 Binary Image Acquisition	15
3.3.3 Root Mean Square Error	15
3.3.4 Jaccard Index.....	17
CHAPTER IV RESULTS	19
4.1 Dataset.....	19
4.2 Evaluation Result	19
CHAPTER V CONCLUSION	26
5.1 Conclusion.....	26
REFERENCES.....	27
APPENDIX A The Project Proposal of Course 2301399 Project Proposal Academic Year 2020	30
APPENDIX B Automatic Aircraft Shadow Removal from Remote Sensing Images Using Mask-ShadowGAN.....	35
BIOGRAPHY	40

LIST OF TABLES

	Page
Table 1.4.1 Project Activities	3
Table 4.2.1 Evaluation Score for Five Highest Score Results	22
Table 4.2.2 Evaluation Score for Five Lowest Score Results	23

LIST OF FIGURES

	Page
Figure 2.3.1 Mask-ShadowGAN architecture.....	6
Figure 3.1.1 Mask-ShadowGAN for aircraft shadow removal architecture:	7
(a) Learning from actual shadow images and.....	7
(b) Learning from actual shadow-free images.....	7
Figure 3.2.1 The necessary imported libraries	9
Figure 3.2.2 The arguments code	10
Figure 3.2.3 The directory specification and GPU checking code.....	10
Figure 3.2.4 The models initiation code.....	10
Figure 3.2.5 The loss functions, optimizer and LR scheduler initiation code.....	11
Figure 3.2.6 The code for continue training.....	11
Figure 3.2.7 The input image definition and memory allocation code	11
Figure 3.2.8 The code for hold loss values	12
Figure 3.2.9 The training process for the generators.....	12
Figure 3.2.10 The training process for the shadow generator.....	13
Figure 3.2.11 The code for logging the loss values	13
Figure 3.2.12 The code for update learning rate and save the model state	14
Figure 3.2.12 The code for save model state and loss graph images by epoch.....	14
Figure 3.3.3.1 The RMSE code.....	16
Figure 3.3.3.2 The predicted and true value lists with RMSE result	16
Figure 3.3.4.1 The Jaccard Index code.....	17
Figure 3.3.4.2 The predicted and true value lists with Jaccard Index result	18
Figure 4.2.1 The five highest score results: (a) Original image,.....	20
(b) Manually shadow removal image (ground truth),	20
(c) GAN generated of (a), (d) Aircraft shape of (b),	20
(e) Aircraft shape of (c), (f) Aircraft shape from Meeboonmak [4].....	20
Figure 4.2.2 The five lowest score results: (a) Original image,.....	21
(b) Manually shadow removal image (ground truth),	21
(c) GAN generated of (a), (d) Aircraft shape of (b),	21

(e) Aircraft shape of (c)	21
Figure 4.2.3 Sample of the original image (a) and the corrupted version (b)	22
Figure 4.2.4 Overall score graph and average score	23
Figure 4.2.5 Score comparison between Mask-ShadowGAN and DSS by image order	24
Figure 4.2.6 An image that Mask-ShadowGAN performs better than DSS	24
Figure 4.2.7 An image that DSS performs better than Mask-ShadowGAN	25
Figure 4.2.8 Overall score graph and average score between using and not using the augmentation	25

CHAPTER I

INTRODUCTION

1.1 Background and Rationale

Image classification models might poorly perform when working with objects that have the shadow since it can make their shape different from it should be. Removing the shadow without disturbing other detail on it can ease the image classification model in many ways. For instance, shadow-free images can help the model correctly classified, while all preserved detail can be used later to identify the object model or type. Also, we can use it to train the machine learning model if you have only shadow images but intend to use shadow-free images.

There are some proposed methods for shadow detection and removal. For instance, A. Gatter [1] proposed an algorithm to identify a self-cast shadow of aircraft while in airborne, which can be removed later by other procedures. Rahman et al. [2] proposed a method to detect and remove shadows in each orthophoto component of UAV-based orthomosaics. Then use the feature-matching technic with help from a commercial software package to create the final orthomosaic.

In the aircraft images, the aircraft from remote sensing images consist of issues for making classification correctly, such as background complexity, blurring, light scattering, shadows on the body, or chromatic distortion. As we mentioned above, these problems can cause the classification model to defectively classify the aircraft. Aircraft shape might look unusual if the background is too complicated due to many other objects such as stair trucks. In this work, we mainly focus on the shadow because the shadow can make the aircraft's shape being torn apart. The challenging issue is how can we remove object shadow without losing the shape of the aircraft. Wei et al. [3] proposed shadow detection and removal as part of image preprocessing by scanning an original aircraft image to create a binary shadow mask. The mask will use to indicate a shadow pixel location, which applies in the removal process. In the present day, Machine Learning techniques can do many things better and faster than any others. Many researchers use Machine Learning to remove object shadow. For instance, Meeboonmak [4] employed Deeply Supervised Salient Object Detection with Short

Connections (DSS) proposed by Hou et al. [5] and doing some post-processing to remove shadows from the aircraft while still maintain aircraft shape integrity.

Generative Adversarial Network (GAN) is a popular method to generate a new image based on its individual properties, which involved shadow removal on the image. Many researchers proposed the shadow removal method by using GAN. For example, Wang et al. [6] implemented Stacked Conditional Generative Adversarial Network (ST-CGAN) to detect the shadow and then remove it. Zhang et al. [7] used GAN to inspect residual and illumination of the image to remove the shadow.

Hu et al. [8] proposed a new way to remove the shadow by learning from unpaired shadow and shadow-free data called Mask-ShadowGAN. This model learned to create shadow masks from real shadow and generated shadow-free images to use as guidance for shadow generation, which used to improve shadow-free generator to produce precise shadow-free images.

To that aim, we adopt the Mask-ShadowGAN to our aircraft images dataset. Then we use generated shadow-free aircraft images to find aircraft shape in a binary manner to compare with the ground truth shape to see if a generated shadow-free image still preserves intact aircraft shape. There are three reasons why we use this model. First, it uses unpaired images, which suitable for data gathered from Google Earth. Second, the model does not require the same amount of data between shadow and shadow-free images, which is good because shadow-free images are hard to find. Finally, there is no application of this model for aircraft images yet.

1.2 Objectives

To apply Mask-ShadowGAN to our remote sensing airplane image dataset, which contains both shadow and shadow-free images to inspect if its capability of aircraft shadow removal.

1.3 Scope

- An image dataset contains only an aircraft from remote sensing images from Google Earth, which will resize to 400 by 400 pixels.
- Aircraft shadows range from a little to long shade, which is an effect from Sun at different times.

- There is a limitation, which is ground truth images are hard to obtain. Hence, we use photoshop to edit shadow images to remove the shadow manually, which makes them the ground truth images.

1.4 Project Activities

1. Studying all aspect of related research paper
2. Collect images from Google Earth to create the dataset
3. Implementing and debugging the system
4. Training and Testing
5. Summary and making the project report

Table 1.4.1 Project Activities

Project Activities	2020				2021		
	Sept.	Oct.	Nov.	Dec.	Jan.	Feb.	Mar.
1. Studying all aspect of related research paper							
2. Collect images from Google Earth to create the dataset							
3. Implementing and debugging the system							
4. Training and Testing							
5. Summary and making the project report							

1.5 Benefits

The benefits for student

1. To develop deep learning skills
2. To learn more deep learning framework
3. To gain the data preparation skill
4. To develop thinking and analyzing skill

The benefits for users

1. To ease their classification model

2. To generate more artificial data for them to feed into the training process
For research purpose

1. This new technique can be improved to produce more realistic images for the images contain the object and its shadow within.

1.6 Report Outlines

In chapter 2, we will discuss research papers that are involved in the shadow removal task and this work.

In chapter 3, the detailed methods for removing the shadow are reviewed, along with the code explanation.

In chapter 4, we will examine the dataset and preprocessing technique. After that, the result evaluation will be discussed, as well as the comparison between models.

In chapter 5, there is a conclusion for what we have done in this work. We will consider what the framework did for the good and drawback. Afterward, we will discuss issues we faced and the solutions we used, as well as improvements.

CHAPTER II

LITERATURE REVIEW

The research papers that are involved in shadow removal and this work will be reviewed in this chapter.

2.1 Deeply Supervised Salient Object Detection with Short Connections

Hou *et al.* [5] proposed a new framework for salient object detection, which produces the series of short connections for side output from deeper to shallower under the Holistically-nested edge detector (HED) network proposed by Xie *et al.* [9]. Deeper side output has high-level features, which contain more detail out of low-level features. Shallower has low-level features, which can locate the real salient area. A better saliency map can be achieved by combine these two side outputs.

DSS architecture consisted of a base convolution, side-output, short connection, and fusion layer. The base convolution layer depends on VGG16 proposed by Simonyan *et al.* [10]. To extract the saliency maps, they add the side-output layers after the last convolutional layer of each stage (conv1_2, conv2_2, conv3_3, conv4_3, and conv5_3) and the final max-pooling layer (pool5) from VGG16. After that, they use the short connection to combine deeper side output with shallower side output. Lastly, the DSS model then optimized by applying fusion loss, which combined from cross-entropy function in each side output.

2.2 Aircraft Segmentation from Remote Sensing Images using Modified Deeply Supervised Salient Object Detection with Short Connection

Meeboonmak [4] adapted DSS to the aircraft dataset by adding the L2-Normalization layer between the connection of conv1_1 from VGG16 and the shallowest side-output layer, which reduces noise in the background and produces more salient results. Subsequently, post-process then used after obtaining the result from DSS by using Bitwise AND operation between side outputs 5 and 6. Then create the binary mask by thresholding. Next, make the object within the binary mask lager by using the

Dilation procedure. Finally, the fused saliency map then combined with the dilated mask.

2.3 Mask-ShadowGAN: Learning to Remove Shadows from Unpaired Data

Mask-ShadowGAN [8] consisted of two parts: the first part is to learn from real shadow images to produce shadow masks, which used as a guide for shadow generation. The second part is to learn from shadow-free images, which use a random shadow mask to generate shadow on shadow-free images to help training shadow-free generator.

Mask-ShadowGAN has adopted network architecture from Johnson et al. [11] as its generator network, which contains three convolution procedures, nine residual blocks with the stride-two convolutions, and two deconvolutions. Each convolution and deconvolution procedures followed by instance normalization. Then used PatchGAN, which proposed by Isola et al. [12] as the discriminator. There are two sets of each generator and discriminator.

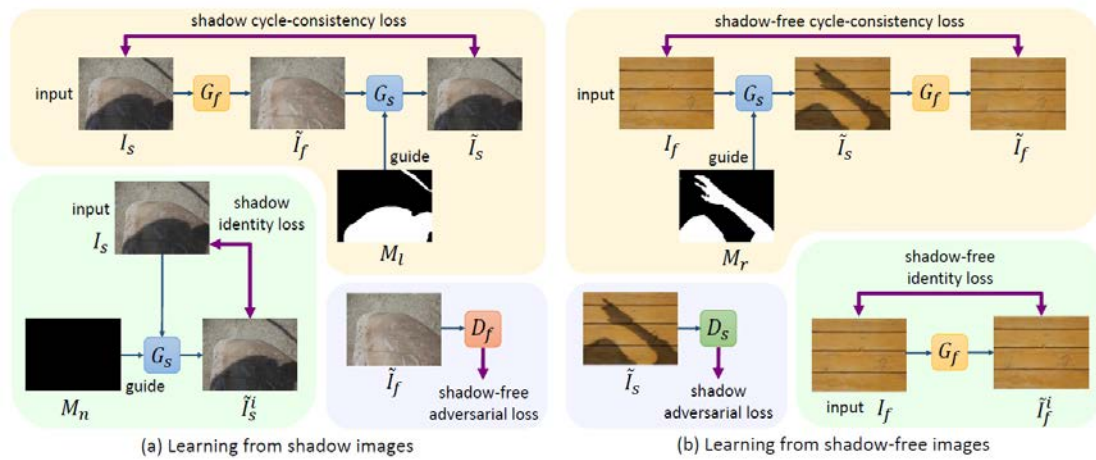


Figure 2.3.1 Mask-ShadowGAN architecture

CHAPTER III

METHODOLOGY

The detailed procedures and the code of this framework for learning and removing shadow are discussed in this chapter.

3.1 Learning to Remove Shadow from Unpaired Data

As we mentioned above, Mask-ShadowGAN uses two sets of each generator and discriminator and contains two parts of the operation. There are three losses in each part: cycle-consistency loss (blue), identity loss (green), and adversarial loss (pink). G_s and G_f are generators that create shadow and shadow-free images. Whereas D_s and D_f denote discriminators, which distinguish generated shadow or shadow-free images to be real or not. I_s and I_f denote the actual shadow and shadow-free images. \tilde{I}_s and \tilde{I}_s^i are the generated shadow imgs, \tilde{I}_f and \tilde{I}_f^i denote the generated shadow-free images, and M_n , M_l and M_r denote the shadow masks as shown in Figure 3.1.1.

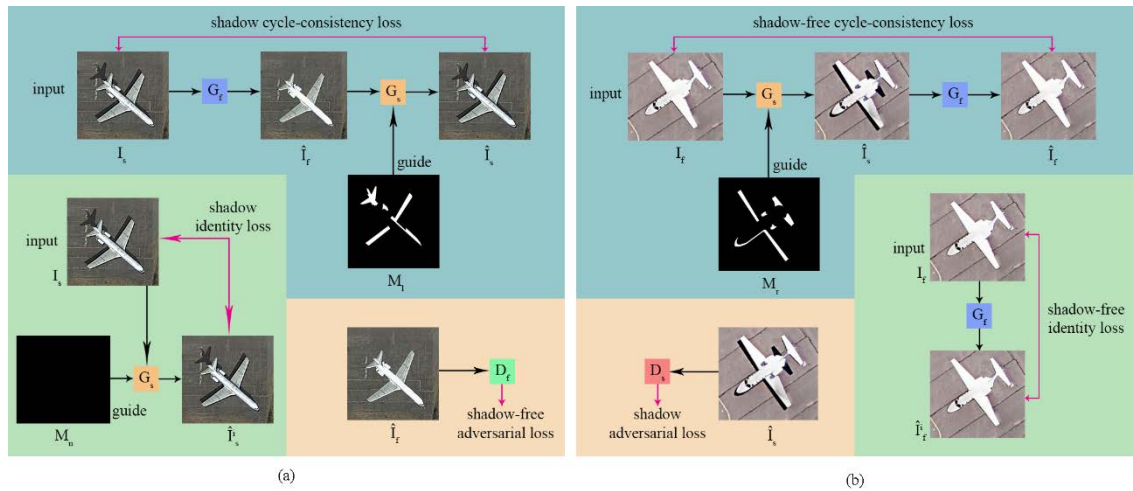


Figure 3.1.1 Mask-ShadowGAN for aircraft shadow removal architecture:

(a) Learning from actual shadow images and

(b) Learning from actual shadow-free images.

First, learning from real shadow images by using a shadow-free generator to generate a shadow-free image. Then use a shadow-free discriminator to examine that

newly generated image is real shadow-free or not. Next, use the adversarial loss to optimize the generator and the discriminator. After that, Hu *et al.* use a shadow generator to convert the generated shadow-free image back to its original and expect to be the same. To do that, they calculate the difference between the generated shadow-free and actual shadow image and binarizing the result to create a mask, which uses as guidance later to preserve the consistency between the generated shadow image, and its original. They formulate shadow cycle consistency loss, which stimulates shadow generated image to be the same as its original.

For the last process of the first part, they create an image with no new shadows added by using an all-zero values mask with a real shadow image to be input for the shadow generator. For this reason, they can regularize the generated image to be identical to its original by using the shadow identity loss.

So they can expect that the shadow generated image will have no new shadow added and conserving color composition.

Second, learning from actual shadow-free images to remove shadows by using a shadow generator to create a shadow image with guidance from the randomly selected previously produced mask. They generate many shadow image by using a different mask to generalize the deep model. Those images later are distinguished by the discriminator, whether it is a real shadow image or not. Then optimize the generator and the discriminator using an adversarial loss.

After that, they take a shadow-free generator to use a shadow generated image to create a shadow-free image. Then optimize the network by using shadow-free cycle consistency loss.

For the last process of this part, they put the actual shadow-free image as input into a shadow-free generator to generate another shadow-free image and later encourage the input to be as identical as an output image. To that aim, an identity loss will be used as a constraint so that the shadow-free generator will learn to remove shadows without disturbing the region that is not the shadow.

For the final loss function, they combine all loss from two parts above as a weighted sum of the adversarial loss, cycle consistency loss, and identical loss and then use minimax to optimize the whole of their framework.

Mask-ShadowGAN uses two sets of each generator and discriminator. For generators, they consisted of three convolution procedures, then there are nine residual

blocks plus the stride-two convolution and do feature map sampling by including two deconvolution operations. After each convolution and deconvolution procedure, they include the instance normalization operation. As mentioned above, there are two generators in this network. The first one is a shadow generator responsible for creating the shadow on images and requires a shadow-free image and a shadow mask to be put in. The second is a shadow-free generator that is accountable for generating shadow-free images by feeding three channels of shadow images. Both of these generators create three channels of residual images that will use as the final shadow or shadow-free image with the input image.

For the two discriminators, they use PatchGAN to determine which image patches are valid or not for shadow-free and shadow images.

3.2 Code Explanations

Figure 3.2.1 shows the necessary imported libraries and utilities for prepare the data and models and save progress. The `__future__` use for compatibility with Python 2. The `argparse` use to define arguments for the model. The `torch` library uses to do the job related to the model and data, such as load or transform images, define loss functions, define optimizers, load or save training states, and check GPU availability. The `PIL` stands for Python Image Library, will do the image job. The `Matplotlib` use to visualize the loss functions. The `models_guided` and `utils` are custom-made utilities by Hu *et al.*: the Mask-ShadowGAN authors for creating models, image datasets, and masks.

```

1 from __future__ import print_function
2 import os
3 import datetime
4 import argparse
5 import itertools
6
7 import torchvision.transforms as transforms
8 from torch.utils.data import DataLoader
9 from torch.autograd import Variable
10 from PIL import Image
11 import torch
12
13 from models_guided import Generator_F2S, Generator_S2F
14 from models_guided import Discriminator
15 from utils import ReplayBuffer
16 from utils import LambdaLR
17 from utils import weights_init_normal
18
19 # training set:
20 from datasets_planes import ImageDataset
21
22 import matplotlib.pyplot as plt
23 from utils import mask_generator
24 from utils import QueueMask

```

Figure 3.2.1 The necessary imported libraries

The following figure, fig. 3.2.2, showing the defined arguments for use to initiate models, dataset, and training process. This time we also establish the argument parser named opt.

```

26 if __name__ == '__main__':
27
28     parser = argparse.ArgumentParser()
29     parser.add_argument('--epoch', type=int, default=0, help='starting epoch')
30     parser.add_argument('--n_epochs', type=int, default=200, help='number of epochs of training')
31     parser.add_argument('--batchSize', type=int, default=1, help='size of the batches')
32     parser.add_argument('--dataroot', type=str, default='datasets/horse2zebra/', help='root directory of the dataset')
33     parser.add_argument('--lr', type=float, default=0.0002, help='initial learning rate')
34     parser.add_argument('--decay_epoch', type=int, default=100,
35                         help='epoch to start linearly decaying the learning rate to 0')
36     parser.add_argument('--size', type=int, default=400, help='size of the data crop (squared assumed)')
37     parser.add_argument('--input_nc', type=int, default=3, help='number of channels of input data')
38     parser.add_argument('--output_nc', type=int, default=3, help='number of channels of output data')
39     parser.add_argument('--cuda', action='store_true', help='use GPU computation')
40     parser.add_argument('--n_cpu', type=int, default=8, help='number of cpu threads to use during batch generation')
41     parser.add_argument('--snapshot_epochs', type=int, default=50, help='number of epochs of training')
42     parser.add_argument('--resume', action='store_true', help='resume')
43     parser.add_argument('--iter_loss', type=int, default=500, help='average loss for n iterations')
44     opt = parser.parse_args()

```

Figure 3.2.2 The arguments code

In fig. 3.2.3, we specify the dataset and log path using opt variable, which is argparse, while checking for available GPU. If we wish to continue learning, we can set the resume argument to true.

```

46 # dataset
47 opt.dataroot = 'S:\Download\Document\CSCU y4\Senior Project\shadow_dataset'
48
49 opt.log_path = os.path.join('output', str(datetime.datetime.now().strftime("%Y%m%d_%H%M%S"))) + '.txt')
50
51 if torch.cuda.is_available():
52     opt.cuda = True
53
54 opt.resume = False

```

Figure 3.2.3 The directory specification and GPU checking code

We define the models in figure 3.2.4, which are generators and discriminators. If the GPU is available, we will assign the GPU to the models. Then we will apply the initial weight for each model.

```

58 ##### Definition of variables #####
59 # Networks
60 netG_A2B = Generator_S2F(opt.input_nc, opt.output_nc) # shadow to shadow_free
61 netG_B2A = Generator_F2S(opt.output_nc, opt.input_nc) # shadow_free to shadow
62 netD_A = Discriminator(opt.input_nc)
63 netD_B = Discriminator(opt.output_nc)
64
65 if opt.cuda:
66     netG_A2B.cuda()
67     netG_B2A.cuda()
68     netD_A.cuda()
69     netD_B.cuda()
70
71 netG_A2B.apply(weights_init_normal)
72 netG_B2A.apply(weights_init_normal)
73 netD_A.apply(weights_init_normal)
74 netD_B.apply(weights_init_normal)

```

Figure 3.2.4 The models initiation code

Figure 3.2.5 illustrates the defined loss functions, optimizer, and LR scheduler (for dynamic learning rate adjustment).

```

76 # Lossess
77 criterion_GAN = torch.nn.MSELoss() # Lsgan
78 # criterion_GAN = torch.nn.BCEWithLogitsLoss() #vanilla
79 criterion_cycle = torch.nn.L1Loss()
80 criterion_identity = torch.nn.L1Loss()
81
82 # Optimizers & LR schedulers
83 optimizer_G = torch.optim.Adam(itertools.chain(netG_A2B.parameters(), netG_B2A.parameters()),
84                                     lr=opt.lr, betas=(0.5, 0.999))
85 optimizer_D_A = torch.optim.Adam(netD_A.parameters(), lr=opt.lr, betas=(0.5, 0.999))
86 optimizer_D_B = torch.optim.Adam(netD_B.parameters(), lr=opt.lr, betas=(0.5, 0.999))
87
88 lr_scheduler_G = torch.optim.lr_scheduler.LambdaLR(optimizer_G,
89                                                     lr_lambda=LambdaLR(opt.n_epochs, opt.epoch, opt.decay_epoch).step)
90 lr_scheduler_D_A = torch.optim.lr_scheduler.LambdaLR(optimizer_D_A,
91                                                     lr_lambda=LambdaLR(opt.n_epochs, opt.epoch, opt.decay_epoch).step)
92 lr_scheduler_D_B = torch.optim.lr_scheduler.LambdaLR(optimizer_D_B,
93                                                     lr_lambda=LambdaLR(opt.n_epochs, opt.epoch, opt.decay_epoch).step)

```

Figure 3.2.5 The loss functions, optimizer and LR scheduler initiation code

The program will load the saved state of the model, optimizer, and LR scheduler. If the resume argument is set to true, as showing in fig. 3.2.6.

```

95 ##### resume the training process
96 if opt.resume:
97     print
98     'resume training:'
99     netG_A2B.load_state_dict(torch.load('output/netG_A2B.pth'))
100    netG_B2A.load_state_dict(torch.load('output/netG_B2A.pth'))
101    netD_A.load_state_dict(torch.load('output/netD_A.pth'))
102    netD_B.load_state_dict(torch.load('output/netD_B.pth'))
103
104    optimizer_G.load_state_dict(torch.load('output/optimizer_G.pth'))
105    optimizer_D_A.load_state_dict(torch.load('output/optimizer_D_A.pth'))
106    optimizer_D_B.load_state_dict(torch.load('output/optimizer_D_B.pth'))
107
108    lr_scheduler_G.load_state_dict(torch.load('output/lr_scheduler_G.pth'))
109    lr_scheduler_D_A.load_state_dict(torch.load('output/lr_scheduler_D_A.pth'))
110    lr_scheduler_D_B.load_state_dict(torch.load('output/lr_scheduler_D_B.pth'))

```

Figure 3.2.6 The code for continue training

In fig. 3.2.7, the input images are initiated. We also do the memory allocation.

```

112 # Inputs & targets memory allocation
113 Tensor = torch.cuda.FloatTensor if opt.cuda else torch.Tensor
114 input_A = Tensor(opt.batchSize, opt.input_nc, opt.size, opt.size)
115 input_B = Tensor(opt.batchSize, opt.output_nc, opt.size, opt.size)
116 target_real = Variable(Tensor(opt.batchSize).fill_(1.0), requires_grad=False)
117 target_fake = Variable(Tensor(opt.batchSize).fill_(0.0), requires_grad=False)
118 mask_non_shadow = Variable(Tensor(opt.batchSize, 1, opt.size, opt.size).fill_(-1.0), requires_grad=False) #-1.0 non-shadow
119
120 fake_A_buffer = ReplayBuffer()
121 fake_B_buffer = ReplayBuffer()

```

Figure 3.2.7 The input image definition and memory allocation code

We define the necessary variables and lists to hold the loss value and the transformation routine for PIL image conversion in figure 3.2.8. Moreover, we create

the mask_queue to contain masks for use in the shadow-free to shadow generation procedure.

```

135     plt.ioff()
136     curr_iter = 0
137     G_losses_temp = 0
138     D_A_losses_temp = 0
139     D_B_losses_temp = 0
140     G_losses = []
141     D_A_losses = []
142     D_B_losses = []
143     to_pil = transforms.ToPILImage()
144
145     mask_queue = QueueMask(dataloader.__len__()//4)
146     open(opt.log_path, 'w').write(str(opt) + '\n\n')

```

Figure 3.2.8 The code for hold loss values

Figure 3.2.9 represents the training process of the two generators. The process will gather the three losses then add up into the total loss while stepping up the optimizer. The training process is described from this figure and onward.

```

148     ##### Training #####
149     for epoch in range(opt.epoch, opt.n_epochs):
150         for i, batch in enumerate(dataloader):
151             # Set model input
152             real_A = Variable(input_A.copy_(batch['A']))
153             real_B = Variable(input_B.copy_(batch['B']))
154
155             ##### Generators A2B and B2A #####
156             optimizer_G.zero_grad()
157
158             # Identity Loss
159             # G_A2B(B) should equal B if real B is fed
160             same_B = netG_A2B(real_B)
161             loss_identity_B = criterion_identity(same_B, real_B) * 5.0 # ||Gb(b)-b||1
162             # G_B2A(A) should equal A if real A is fed, so the mask should be all zeros
163             same_A = netG_B2A(real_A, mask_non_shadow)
164             loss_identity_A = criterion_identity(same_A, real_A) * 5.0 # ||Ga(a)-a||1
165
166             # GAN Loss
167             fake_B = netG_A2B(real_A)
168             pred_fake = netD_B(fake_B)
169             loss_GAN_A2B = criterion_GAN(pred_fake, target_real) # Log(Db(Gb(a)))
170
171             mask_queue.insert(mask_generator(real_A, fake_B))
172
173             fake_A = netG_B2A(real_B, mask_queue.rand_item())
174             pred_fake = netD_A(fake_A)
175             loss_GAN_B2A = criterion_GAN(pred_fake, target_real) # Log(Da(Ga(b)))
176
177             # Cycle Loss
178             recovered_A = netG_B2A(fake_B, mask_queue.last_item()) # real shadow, false shadow free
179             loss_cycle_ABA = criterion_cycle(recovered_A, real_A) * 10.0 # ||Ga(Gb(a))-a||1
180
181             recovered_B = netG_A2B(fake_A)
182             loss_cycle_BAB = criterion_cycle(recovered_B, real_B) * 10.0 # ||Gb(Ga(b))-b||1
183
184             # Total Loss
185             loss_G = loss_identity_A + loss_identity_B + loss_GAN_A2B + loss_GAN_B2A + loss_cycle_ABA + loss_cycle_BAB
186             loss_G.backward()
187
188             #G_losses.append(loss_G.item())
189             G_losses_temp += loss_G.item()
190
191             optimizer_G.step()

```

Figure 3.2.9 The training process for the generators

Unlike the previous figure, fig. 3.2.10 contain the training process of discriminator A, which is the shadow discriminator. The same as discriminator B, which is the shadow-free discriminator. The process collects the real and fake loss then adds up into the total loss while stepping up the optimizer.

```

194     ##### Discriminator A #####
195     optimizer_D_A.zero_grad()
196
197     # Real Loss
198     pred_real = netD_A(real_A)
199     loss_D_real = criterion_GAN(pred_real, target_real) # Log(Da(a))
200
201     # Fake Loss
202     fake_A = fake_A_buffer.push_and_pop(fake_A)
203     pred_fake = netD_A(fake_A.detach())
204     loss_D_fake = criterion_GAN(pred_fake, target_fake) # Log(1-Da(G(b)))
205
206     # Total Loss
207     loss_D_A = (loss_D_real + loss_D_fake) * 0.5
208     loss_D_A.backward()
209
210     #D_A_losses.append(Loss_D_A.item())
211     D_A_losses_temp += loss_D_A.item()
212
213     optimizer_D_A.step()

```

Figure 3.2.10 The training process for the shadow generator

If the current iteration is at 500, we will write all losses and their average to the log file while appending losses to their lists. Besides, the fake shadow and shadow-free images are saved to the output folder, as shown in figure 3.2.11.

```

238     curr_iter += 1
239
240     if (i+1) % opt.iter_loss == 0:
241         log = '[iter %d], [loss_G %.5f], [loss_G_identity %.5f], [loss_G_GAN %.5f], ' \
242             '[loss_G_cycle %.5f], [loss_D %.5f]' % \
243             (curr_iter, loss_G, (loss_identity_A + loss_identity_B), (loss_GAN_A2B + loss_GAN_B2A),
244             (loss_cycle_ABA + loss_cycle_BAB), (loss_D_A + loss_D_B))
245         print(log)
246         open(opt.log_path, 'a').write(log + '\n')
247
248         G_losses.append(G_losses_temp / opt.iter_loss)
249         D_A_losses.append(D_A_losses_temp / opt.iter_loss)
250         D_B_losses.append(D_B_losses_temp / opt.iter_loss)
251         G_losses_temp = 0
252         D_A_losses_temp = 0
253         D_B_losses_temp = 0
254
255         avg_log = '[the last %d iters], [loss_G %.5f], [D_A_losses %.5f], [D_B_losses %.5f], ' \
256                 '% (opt.iter_loss, G_losses[G_losses.__len__()-1], D_A_losses[D_A_losses.__len__()-1], \
257                 D_B_losses[D_B_losses.__len__()-1])'
258         print(avg_log)
259         open(opt.log_path, 'a').write(avg_log + '\n')
260
261         img_fake_A = 0.5 * (fake_A.detach().data + 1.0)
262         img_fake_A = (to_pil(img_fake_A.data.squeeze(0).cpu()))
263         img_fake_A.save('output/fake_A.png')
264
265         img_fake_B = 0.5 * (fake_B.detach().data + 1.0)
266         img_fake_B = (to_pil(img_fake_B.data.squeeze(0).cpu()))
267         img_fake_B.save('output/fake_B.png')

```

Figure 3.2.11 The code for logging the loss values

In figure 3.2.12, the program will update the learning rate and save the model, optimizer, and LR schedule state

```

269     # Update Learning rates
270     lr_scheduler_G.step()
271     lr_scheduler_D_A.step()
272     lr_scheduler_D_B.step()
273
274
275     # Save models checkpoints
276     torch.save(netG_A2B.state_dict(), 'output/netG_A2B.pth')
277     torch.save(netG_B2A.state_dict(), 'output/netG_B2A.pth')
278     torch.save(netD_A.state_dict(), 'output/netD_A.pth')
279     torch.save(netD_B.state_dict(), 'output/netD_B.pth')
280
281     torch.save(optimizer_G.state_dict(), 'output/optimizer_G.pth')
282     torch.save(optimizer_D_A.state_dict(), 'output/optimizer_D_A.pth')
283     torch.save(optimizer_D_B.state_dict(), 'output/optimizer_D_B.pth')
284
285     torch.save(lr_scheduler_G.state_dict(), 'output/lr_scheduler_G.pth')
286     torch.save(lr_scheduler_D_A.state_dict(), 'output/lr_scheduler_D_A.pth')
287     torch.save(lr_scheduler_D_B.state_dict(), 'output/lr_scheduler_D_B.pth')

```

Figure 3.2.12 The code for update learning rate and save the model state

For every 50 epoch, the program will save each model state denote by epoch number. Additionally, the loss of generators and discriminators images is saved, as shown in fig. 3.2.13.

```

289     if (epoch + 1) % opt.snapshot_epochs == 0:
290         torch.save(netG_A2B.state_dict(), ('output/netG_A2B_%d.pth' % (epoch + 1)))
291         torch.save(netG_B2A.state_dict(), ('output/netG_B2A_%d.pth' % (epoch + 1)))
292         torch.save(netD_A.state_dict(), ('output/netD_A_%d.pth' % (epoch+1)))
293         torch.save(netD_B.state_dict(), ('output/netD_B_%d.pth' % (epoch+1)))
294
295     print('Epoch:{}'.format(epoch))
296
297     if (epoch + 1) % opt.snapshot_epochs == 0:
298         plt.figure(figsize=(10, 5))
299         plt.title("Generator Loss During Training")
300         plt.plot(G_losses, label="G loss")
301         plt.xlabel("iterations / %d" % (opt.iter_loss))
302         plt.ylabel("loss")
303         plt.legend()
304         plt.savefig('./output/generator.png')
305         # plt.show(block=False)
306
307         plt.figure(figsize=(10, 5))
308         plt.title("Discriminator Loss During Training")
309         plt.plot(D_A_losses, label="D_A loss")
310         plt.plot(D_B_losses, label="D_B loss")
311         plt.xlabel("iterations / %d" % (opt.iter_loss))
312         plt.ylabel("loss")
313         plt.legend()
314         plt.savefig('./output/discriminator.png')
315         # plt.show(block=False)
316         plt.close('all')

```

Figure 3.2.12 The code for save model state and loss graph images by epoch

3.3 Evaluation Method

We evaluate the model performance by using aircraft shape to compute Root Mean Square Error and Jaccard Similarity Index. The generated shadow-free airplane

images are processed using the image processing technique to extract their shape in a binary manner for evaluation using those criteria. We evaluate the generated shadow-free airplane images in LAB color space using RMSE and evaluate its binary using Jaccard Index.

3.3.1 LAB color space

This color space has three values according to the color. One is L^* (perceptual lightness), two for the position of the color, which are a^* (between red and green), and b^* (between yellow and blue). These values are the coordinate of three-dimensional space.

3.3.2 Binary Image Acquisition

To demonstrate the performance of Mask-ShadowGAN, we do custom image processing steps to extract most of an aircraft shape in testing images. We will eliminate unwanted objects and fill the hole in the aircraft shape. Here are the steps.

1. We do the thresholding using a thresholding value between 180, 190, 200, 210, and 220 since the images differ in light level. We look at the image and determine which values should be used. We will obtain binary images in this step.
2. After that, we will filter out the unwanted object in the binary image. To do that, we are looping through the image to get the size of the white objects. By default, the airplane shape is the largest object in the image. So we will filter other things small than the airplane out.
3. In this step, we will erode or dilate the airplane to exactly match its shape from its original image. We use a 3x3, 5x5, or 7x7 mask depends on its shape size. We also look for the hole in the aircraft shape to fill it later.

3.3.3 Root Mean Square Error

Root Mean Square Error (RMSE) [13] is a measure of the difference between real value and predicted value. The RMSE value is non-negative, and the lower value indicates better performance than the higher value. The RMSE is the square root of Mean Square Error (MSE), which is defined below.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} (\hat{y}_i - y_i)^2} \quad (1)$$

where \hat{y}_i denoted predicted value (generated image), y_i indicated real value (ground truth image), and n is the total value.

```

1 import numpy as np
2 import math as m
3
4 def RMSE(true_val, pred_val):
5
6     sum_square_dif = sum(list(map(lambda x: (x[0]-x[1])**2, zip(*(pred_val, true_val)))))
7     rmse = m.sqrt(sum_square_dif/len(true_val))
8
9     return rmse
10
11 np.random.seed(0)
12
13 true_val = np.random.randint(1, 100, size=5)
14 pred_val = np.random.randint(1, 100, size=5)
15 print(f'Predicted value list: {pred_val}\nTrue value list:      {true_val}')
16
17 rmse = RMSE(true_val, pred_val)
18 print(f'RMSE: {rmse}')

```

Figure 3.3.3.1 The RMSE code

- In figure 3.3.3.1, the RMSE function, which takes the list of the real and predicted value to compute RMSE, has two variables: the sum_square_dif and rmse.
- The sum_square_dif uses the zip function to pack each pair of the real and predicted value into a tuples list. For instance, [1, 2] and [3, 4] are packed into [(1, 3), (2, 4)]. Then use map function with lambda to map each tuple to square difference formula, which converted into a list such that [(1, 3), (2, 4)] is turned to [4, 4].
- The sum function is used to sum all elements in that list, so [4, 4] is now 8.
- The rmse takes the sum divided by total value, which is the length of either real or predicted value list, then uses the square root to calculate the RMSE.

```

Predicted value list: [10 84 22 37 88]
True value list:      [45 48 65 68 68]
RMSE: 33.855575611706854

```

Figure 3.3.3.2 The predicted and true value lists with RMSE result

- If we packed these two lists from figure 3.3.3.2, then compute the square difference, we will get [1225, 1296, 1849, 961, 400]. After that, we will sum it into 5731.
- Eventually, we divided by the total number of values, which is 5, we get 1146.2. Then use square root to calculate RMSE, which is approximately 33.856.

3.3.4 Jaccard Index

Jaccard Index [14] is a measure of similarity between two sets. Its value is non-negative. Unlike the RMSE, the higher value indicates better performance than the lower value. Jaccard Index is defined by

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \quad (2)$$

where A and B denoted true value (ground truth image) and predicted value set (generated image), respectively.

```

1 import numpy as np
2 from sklearn.metrics import jaccard_score
3
4 np.random.seed(1)
5
6 true_val = np.random.randint(0, 2, size=(3, 3))
7 pred_val = np.random.randint(0, 2, size=(3, 3))
8
9 jaccard_score = jaccard_score(true_val, pred_val, average='weighted')
10
11 print(f'Predicted:\n {pred_val}')
12 print(f'True:\n {true_val}')
13 print(f'Jaccard Similarity Score: {jaccard_score}')

```

Figure 3.3.4.1 The Jaccard Index code

- We use the Scikit-learn to compute the Jaccard Index [15]. It calculates the intersection by column and average by total elements in that column. After that, it computes the weighted average using the number of true values from each column as the weight.

```

Predicted:
[[0 0 1]
 [0 1 1]
 [0 0 1]]
True:
[[1 1 0]
 [0 1 1]
 [1 1 1]]
Jaccard Similarity Score: 0.3333333333333333

```

Figure 3.3.4.2 The predicted and true value lists with Jaccard Index result

- We can see that the intersection of two sets has three items. We get the list of the score of each column: $[0, \frac{1}{3}, \frac{2}{3}]$.
- For each element in that list, we multiply it by weight. The weight is the number of the true instance of each column from the true set. We get
$$\frac{((0 \times 2) + (\frac{1}{3} \times 3) + (\frac{2}{3} \times 2))}{7} = 0.333.$$

CHAPTER IV

RESULTS

The dataset and preprocessing technique are reviewed, as well as result evaluation. Besides, the comparison of the results between models will be shown in this chapter.

4.1 Dataset

We gathered both shadow and shadow-free aircraft images from Google Earth, which include 1400 and 233 aircraft for shadow and shadow-free images, respectively.

Due to outnumber for shadow-free images, we use the image augmentation process to increase their quantity. We rotate them for 90, 180, and 270 degrees then add them up to the shadow-free collection, which adds up to 932 images. After that, we divide the shadow image set into 84% for training and 16% for testing. We leave no touch for shadow-free collection because it is for training only. We then resize images to 400x400 pixels for use in the training and testing process.

To evaluate, when we get the results from the fed testing images, there is a need for their identical ground truth to be used to test to see if the shadow is gone or not, which is difficult to obtain, especially when images collected from Google Earth. Therefore, we manually remove the shadow from shadow images using Photoshop to create ground truth images instead.

4.2 Evaluation Result

To evaluate the model's performance, we use root mean square error (RMSE) defined by

$$RMSE = \sqrt{\frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i, j) - K(i, j)]^2} \quad (3)$$

where I and K denote the generated shadow-free images and the ground truth images. These images are RGB images and will be converted to LAB color space later. And we use the Jaccard index as defined in equation 2, in which A and B denote aircraft shape

of generated shadow-free images and aircraft shape of ground truth images. Normally, the lower RMSE, the better result produced, as opposed to the Jaccard index.

We followed the Mask-ShadowGAN shadow removal evaluation by calculating RMSE between the ground truth and the generated shadow-free image in LAB color space [16, 17, 18]. After that, we compute the similarity score between the shape of aircraft in the ground truth and the generated shadow-free image in a binary manner. We extract the aircraft shape to be a binary image by following the steps from 3.3.2 Binary Image Acquisition. We compared our results with the results from the Meeboonmak [4] algorithm. We did not compare with other methods because Hu et al. [8] already compared and concluded that's Mask-ShadowGAN provided better results among them.

We collect five of each highest and lowest score by Jaccard similarity from the testing set. The results are shown in Figure 4.2.1 and Figure 4.2.2 respectively.



Figure 4.2.1 The five highest score results: (a) Original image, (b) Manually shadow removal image (ground truth), (c) GAN generated of (a), (d) Aircraft shape of (b), (e) Aircraft shape of (c), (f) Aircraft shape from Meeboonmak [4]

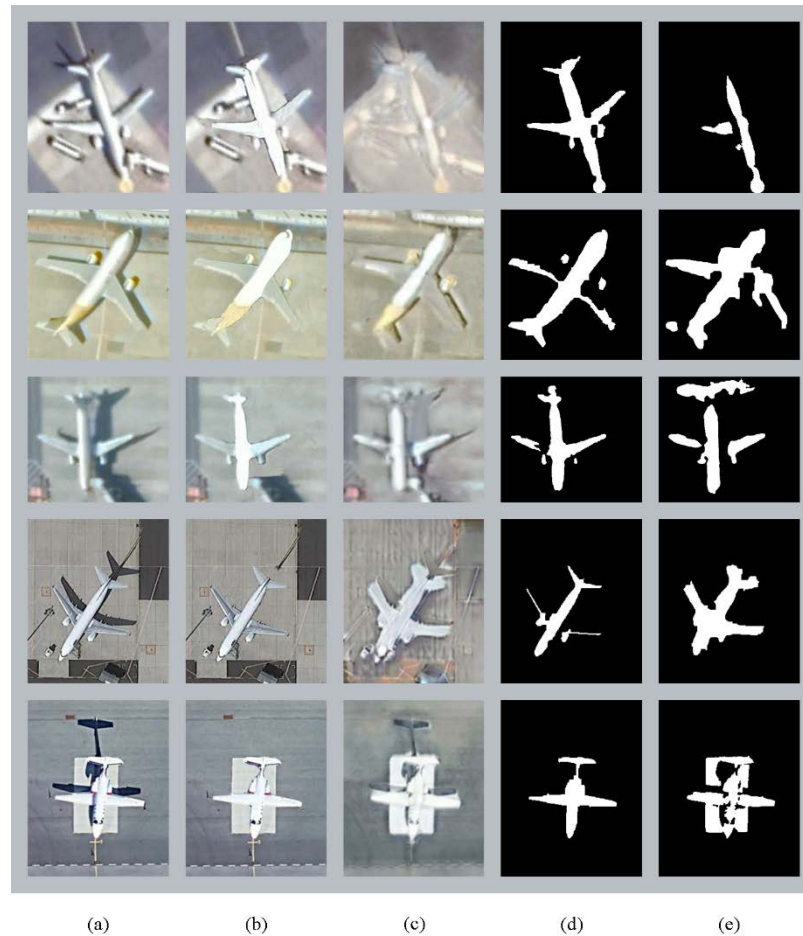





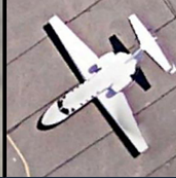

Figure 4.2.2 The five lowest score results: (a) Original image, (b) Manually shadow removal image (ground truth), (c) GAN generated of (a), (d) Aircraft shape of (b), (e) Aircraft shape of (c)

Then we plot all Jaccard index according to the individual images into a line graph, which represents overall performance, as shown in Figure 4.2.4. Also, we indicate the average score of this framework as the graph title.

As we see, the highest score images from Figure 4.2.1, the generated shadow-free may not preserve enough detail to be the same as its original, and the RMSE score from Table 4.2.1 is not low as expected. Despite high RMSE, we can see that the Jaccard similarity score is quite high as the model still preserves the aircraft's shape the same as its original, which is good. Compare to ground truth shape, most of the generated aircraft's shape is still similar to the ground truth. When compare to Meeboonmak [4], most of the aircraft's shape is still better intact, while some of our images have a hole or

have a torn part like a tail. If we take a look at generated images one more time, we will see that there are some shadow left but thinned, which caused less impact than the full shadow.

Table 4.2.1 Evaluation Score for Five Highest Score Results

Score Type					
RMSE	14.16	19.30	4.66	11.29	12.30
Jaccard Similarity Index	0.9359	0.9349	0.9277	0.9027	0.9001

For the lowest score images from Figure 4.2.2, we can see that the generated aircraft's shape has various issues such as torn or missing parts, bloated or thinned parts, or merge with the background object. If we look into generated shadow-free images, we will see that many images have the artifact, and the shadow is still present but more faded. Despite corrupted things in fig. 4.2.3, there exists appropriate stuff. For instance, some airplane images can regain their part back when converted into a binary image. However, those images still do not match their binary ground truth image. The problem mentioned above, such as the artifact will make aircraft parts look bloated. From the score in Table 4.2.2, most images perform better when evaluate using RMSE than images in Fig. 4.2.2, but the Jaccard similarity score is low due to their ground truth cannot be extracted to complete the shape in some images.

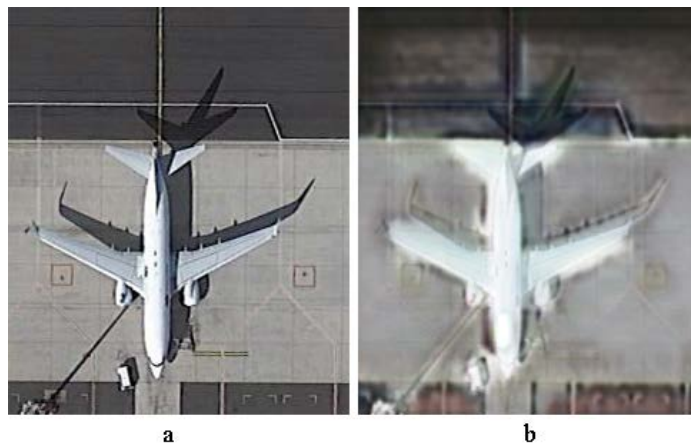







Figure 4.2.3 Sample of the original image (a) and the corrupted version (b)

Table 4.2.2 Evaluation Score for Five Lowest Score Results

Score Type					
RMSE	12.52	10.61	11.08	16.35	9.73
Jaccard Similarity Index	0.5460	0.5430	0.5059	0.4553	0.4426

The line graph in Fig. 4.2.4 illustrates the overall Jaccard index similarity score of the individual image generated from 2 frameworks: Mask-ShadowGAN (blue) and Meeboonmak [4] (orange), which indicates average scores: 0.7799 and 0.7775, respectively. We divided a graph into three portions to show the area of the lower or higher score of the Mask-ShadowGAN and Meeboonmak method. In the middle portion, the Meeboonmak algorithm performs quite well. Though, in the first and last part, Mask-ShadowGAN reaches a higher score. We can see that Mask-ShadowGAN achieves better in the average (located above graph), nethermost, and uppermost scores. In this case, we are only focusing on the sorted score between both frameworks.

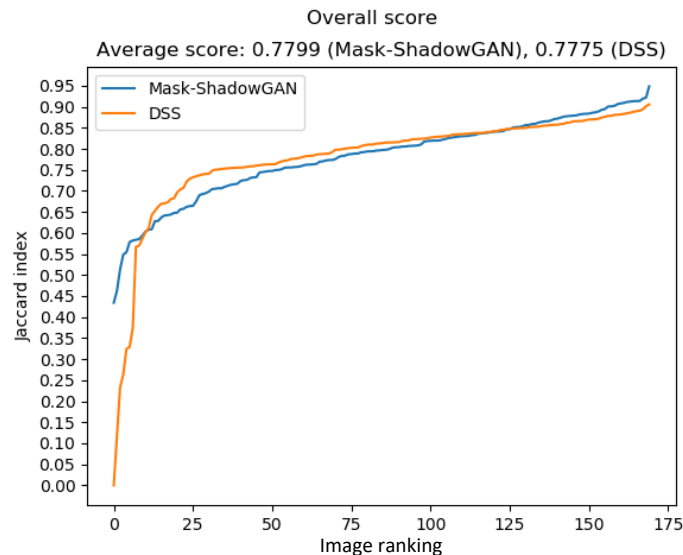
**Figure 4.2.4 Overall score graph and average score**

Figure 4.2.5 is the alternate version of figure 4.2.4, which indicates the score by image order sorted by Mask-ShadowGAN score. We can see that the Mask-

ShadowGAN and DSS perform better on some images. Figure 4.2.6 illustrates the blurred image. Mask-ShadowGAN score is 0.732, and DSS score is 0, in which Mask-ShadowGAN performs much better than DSS. Figure 4.2.7 indicates the detailed image with light illumination on top of the plane. Mask-ShadowGAN score is 0.462, and DSS score is 0.7572, in which DSS performs quite better than Mask-ShadowGAN.

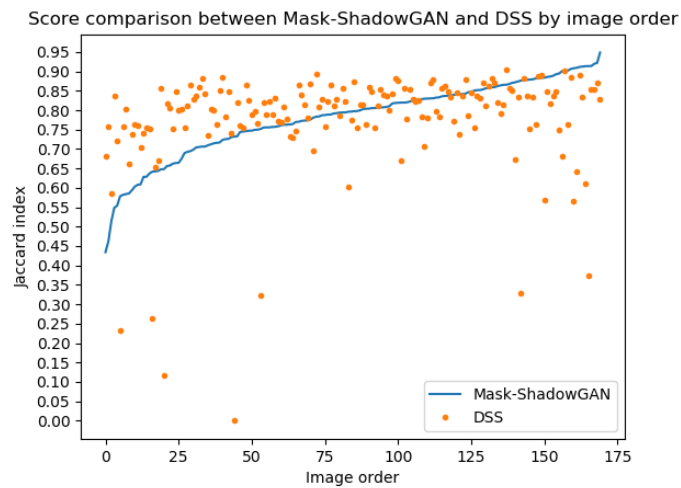


Figure 4.2.5 Score comparison between Mask-ShadowGAN and DSS by image order



Figure 4.2.6 An image that Mask-ShadowGAN performs better than DSS



Figure 4.2.7 An image that DSS performs better than Mask-ShadowGAN

Also, we are training the algorithms without data augmentation to inspect if the framework still steady performs the task. After that, we compare the result between the models that using and not using the augmentation. Figure 4.2.8 illustrates that the augmentation helps improve the model score.

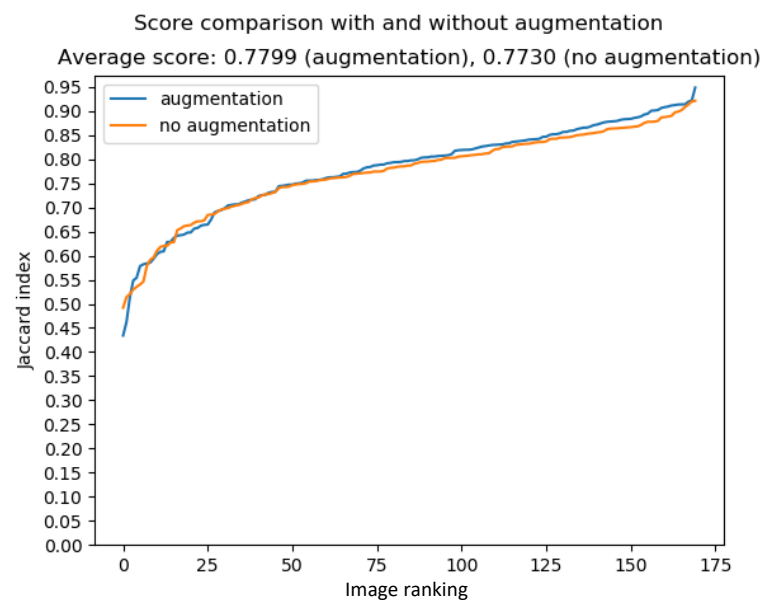


Figure 4.2.8 Overall score graph and average score between using and not using the augmentation

CHAPTER V

CONCLUSION

We will conclude what we did along with the advantage and the disadvantage of this framework. Also, the problems we encountered and the solutions we used are discussed in this chapter.

5.1 Conclusion

In this paper, we adopted the Mask-ShadowGAN and applied to our aircraft images dataset. From the experimental result, we can see that GAN may not entirely remove the object shadow but still makes it faded out or thinned. Although, the experimental result shows us that generated shadow-free aircraft's detail may be blurred out or blended in with the aircraft body but, its shape is still intact.

There are some issues when working with remote sensing images, especially airplane images, from the internet. For instance, we cannot get enough shadow-free images to feed into the system, so we do the augmentation instead, as we mentioned earlier. Additionally, there are no ground truth images for evaluating shadow images. Since the ground truth image is the shadow image itself but has no shadow. We tackle this problem by using Photoshop to create the ground truth images manually from shadow images.

In future work, we will improve the model that will thoroughly remove shadows and preserve the object detail. So that the removed shadow aircraft image can be further used for other purposes such as aircraft type classification.

REFERENCES

- [1] A. Gatter, Classifying self-cast shadow regions in aerial camera images, 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Auckland, New Zealand, pp. 1-6, 2018.
- [2] M. M. Rahman, G. Mcdermid, T. Mckeeman, J. Lovitt, A workflow to minimize shadows in UAV-based orthomosaics. *Journal of Unmanned Vehicle Systems* 7.2, pp. 107-117, 2019.
- [3] W. Wei and J. Zhang, Remote Sensing Image Aircraft Detection Technology Based on Deep Learning, 2019 11th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, pp. 173-177, 2019.
- [4] N. Meeboonmak and N. Cooharojananone, Aircraft Segmentation from Remote Sensing Images using Modified Deeply Supervised Salient Object Detection with Short Connection. *International Conference on Mathematics and Computers in Science and Engineering (MACISE)*, pp. 184-187, 2020.
- [5] Q. Hou, M. Cheng, X. Hu, A. Borji, Z. Tu, P. Torr, Deeply Supervised Salient Object Detection with Short Connections. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 815-828, 2018.
- [6] J. Wang, X. Li, L. Hui, J. Yang, Stacked Conditional Generative Adversarial Networks for Jointly Learning Shadow Detection and Shadow Removal. In *CVPR*, pp. 1788-1797, 2018.
- [7] L. Zhang, C. Long, X. Zhang, C. Xiao, RIS-GAN: Explore Residual and Illumination with Generative Adversarial Networks for Shadow Removal. *AAAI Conference on Artificial Intelligence (AAAI)*, pp.12829-12836, 2020.
- [8] X. Hu, Y. Jiang, C. Fu, P. Heng, Mask-ShadowGAN: Learning to Remove Shadows from Unpaired Data. *IEEE International Conference on Computer Vision (ICCV)*, pp. 2472-2481, 2019.
- [9] S. Xie, Z. Tu, Holistically-Nested Edge Detection. 2015 *IEEE International Conference on Computer Vision (ICCV)*, Santiago, pp. 1395-1403, 2015.

- [10] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition. International Conference on Learning Representations, arXiv: 1409.1556v6, pp. 1-14, 2015.
- [11] J. Johnson, A. Alahi, L. Fei-Fei, Perceptual Losses for Real-Time Style Transfer and Super-Resolution. European Conference on Computer Vision, pp. 694-711, 2016.
- [12] P. Isola, J. Zhu, T. Zhou, A. A. Efros, Image-to-Image Translation with Conditional Adversarial Networks. In CVPR, pp. 1125-1134, 2017.
- [13] Wikipedia, Root-mean-square deviation[Online]. Available from:
https://en.wikipedia.org/wiki/Root-mean-square_deviation [1 Mar. 2021]
- [14] Wikipedia, Jaccard index [Online]. Available from:
https://en.wikipedia.org/wiki/Jaccard_index [1 Mar. 2021]
- [15] Wikipedia, sklearn.metrics.jaccard_score[Online]. Available from:
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.jaccard_score.html [1 Mar. 2021]
- [16] Xiaowei Hu, Chi-Wing Fu, Lei Zhu, Jing Qin, and Pheng-Ann Heng. Direction-aware spatial context features for shadow detection and removal. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.
- [17] Liangqiong Qu, Jiandong Tian, Shengfeng He, Yandong Tang, and Rynson W.H. Lau. DshadowNet: A multi-context embedding deep network for shadow removal. In CVPR, pp. 4067–4075, 2017.
- [18] Jifeng Wang, Xiang Li, and Jian Yang. Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal. In *CVPR*, pp. 1788–1797, 2018

APPENDICES

APPENDIX A

The Project Proposal of Course 2301399 Project Proposal Academic Year 2020

Project Tittle (Thai)	การลบเงาของเครื่องบินโดยอัตโนมัติจากรูปภาพระยะไกลโดยใช้ Mask-ShadowGAN
Project Tittle (English)	Automatic Aircraft Shadow Removal from Remote Sensing Images using Mask-ShadowGAN
Project Advisor	Assoc. Prof. Dr. Nagul Cooharajanone
By	1. Sirapavee Ganyaporngul ID 6033662623 Computer Science Program, Department of Mathematics and Computer Science Faculty of Science, Chulalongkorn University

Background and Rationale

Image classification models might poorly perform when working with objects that have the shadow since it can make their shape different from it should be. Removing the shadow without disturbing other detail on it can ease the image classification model in many ways. For instance, shadow-free images can help the model correctly classified, while all preserved detail can be used later to identify the object model or type. Also, we can use it to train the machine learning model if you have only shadow images but intend to use shadow-free images.

There are some proposed methods for shadow detection and removal. For instance, A. Gatter [1] proposed an algorithm to identify a self-cast shadow of aircraft while in airborne, which can be removed later by other procedures. Rahman et al. [2] proposed a method to detect and remove shadows in each orthophoto component of UAV-based orthomosaics. Then use the feature-matching technic with help from a commercial software package to create the final orthomosaic.

In the aircraft images, the aircraft from remote sensing images consist of issues for making classification correctly, such as background complexity, blurring, light scattering, shadows on the body, or chromatic distortion. As we mentioned above, these

problems can cause the classification model to defectively classify the aircraft. Aircraft shape might look unusual if the background is too complicated due to many other objects such as stair trucks. In this work, we mainly focus on the shadow because the shadow can make the aircraft's shape being torn apart. The challenging issue is how can we remove object shadow without losing the shape of the aircraft. Wei et al. [3] proposed shadow detection and removal as part of image preprocessing by scanning an original aircraft image to create a binary shadow mask. The mask will use to indicate a shadow pixel location, which applies in the removal process. In the present day, Machine Learning techniques can do many things better and faster than any others. Many researchers use Machine Learning to remove object shadow. For instance, Meeboonmak [4] employed Deeply Supervised Salient Object Detection with Short Connections (DSS) proposed by Hou et al. [5] and doing some post-processing to remove shadows from the aircraft while still maintain aircraft shape integrity.

Generative Adversarial Network (GAN) is a popular method to generate a new image based on its individual properties, which involved shadow removal on the image. Many researchers proposed the shadow removal method by using GAN. For example, Wang et al. [6] implemented Stacked Conditional Generative Adversarial Network (ST-CGAN) to detect the shadow and then remove it. Zhang et al. [7] used GAN to inspect residual and illumination of the image to remove the shadow.

Hu et al. [8] proposed a new way to remove the shadow by learning from unpaired shadow and shadow-free data called Mask-ShadowGAN. This model learned to create shadow masks from real shadow and generated shadow-free images to use as guidance for shadow generation, which used to improve shadow-free generator to produce precise shadow-free images.

To that aim, we adopt the Mask-ShadowGAN to our aircraft images dataset. Then we use generated shadow-free aircraft images to find aircraft shape in a binary manner to compare with the ground truth shape to see if a generated shadow-free image still preserves intact aircraft shape. There are three reasons why we use this model. First, it uses unpaired images, which suitable for data gathered from Google Earth. Second, the model does not require the same amount of data between shadow and shadow-free images, which is good because shadow-free images are hard to find. Finally, there is no application of this model for aircraft images yet.

Objectives

To apply Mask-ShadowGAN to our remote sensing airplane image dataset, which contains both shadow and shadow-free images to inspect if its capability of aircraft shadow removal.

Scope

- An image dataset contains only an aircraft from remote sensing images from Google Earth, which will resize to 400 by 400 pixels.
- Aircraft shadows range from a little to long shade, which is an effect from Sun at different times.
- There is a limitation, which is ground truth images are hard to obtain. Hence, we use photoshop to edit shadow images to remove the shadow manually, which makes them the ground truth images.

Project Activities

Study Plan

1. Studying all aspect of related research paper
2. Collect images from Google Earth to create the dataset
3. Implementing and debugging the system
4. Training and Testing
5. Summary and making the project report

Duration of study

Study Plan	2020				2021		
	Sept.	Oct.	Nov.	Dec.	Jan.	Feb.	Mar.
1. Studying all aspect of related research paper							
2. Collect images from Google Earth to create the dataset							
3. Implementing and debugging the system							
4. Training and Testing							

5. Summary and making the project report							
--	--	--	--	--	--	--	--

Benefits

The benefits for student

1. To develop deep learning skills
2. To learn more deep learning framework
3. To gain the data preparation skill
4. To develop thinking and analyzing skill

The benefits for users

1. To ease their classification model
2. To generate more artificial data for them to feed into the training process

For research purpose

1. This new technique can be improved to produce more realistic images for the images contain the object and its shadow within.

Equipment

1. Hardware
 - 1.1. Personal laptop
 - Intel Core i7-7700HQ 2.80 GHz
 - 16 GB RAM
 - Nvidia GTX 1050M
2. Software
 - 2.1. Visual Studio Code
 - 2.2. Google Colab
 - 2.3. Chrome Remote Desktop
3. Miscellenous
 - 3.1. GitHub

Budget

This project requires no budget.

References

- [1] A. Gatter, Classifying self-cast shadow regions in aerial camera images, 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Auckland, New Zealand, pp. 1-6, 2018.
- [2] M. M. Rahman, G. Mcdermid, T. Mckeeman, J. Lovitt, A workflow to minimize shadows in UAV-based orthomosaics. *Journal of Unmanned Vehicle Systems* 7.2, pp. 107-117, 2019.
- [3] W. Wei and J. Zhang, Remote Sensing Image Aircraft Detection Technology Based on Deep Learning, 2019 11th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, pp. 173-177, 2019.
- [4] N. Meeboonmak and N. Cooharajanane, Aircraft Segmentation from Remote Sensing Images using Modified Deeply Supervised Salient Object Detection with Short Connection. *International Conference on Mathematics and Computers in Science and Engineering (MACISE)*, pp. 184-187, 2020.
- [5] Q. Hou, M. Cheng, X. Hu, A. Borji, Z. Tu, P. Torr, Deeply Supervised Salient Object Detection with Short Connections. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 815-828, 2018.
- [6] J. Wang, X. Li, L. Hui, J. Yang, Stacked Conditional Generative Adversarial Networks for Jointly Learning Shadow Detection and Shadow Removal. In *CVPR*, pp. 1788-1797, 2018.
- [7] L. Zhang, C. Long, X. Zhang, C. Xiao, RIS-GAN: Explore Residual and Illumination with Generative Adversarial Networks for Shadow Removal. *AAAI Conference on Artificial Intelligence (AAAI)*, pp.12829-12836, 2020.
- [8] X. Hu, Y. Jiang, C. Fu, P. Heng, Mask-ShadowGAN: Learning to Remove Shadows from Unpaired Data. *IEEE International Conference on Computer Vision (ICCV)*, pp. 2472-2481, 2019.

APPENDIX B

Automatic Aircraft Shadow Removal from Remote Sensing Images Using Mask-ShadowGAN

2021 IEEE 8th International Conference on Industrial Engineering and Applications

Automatic Aircraft Shadow Removal from Remote Sensing Images Using Mask-ShadowGAN

Sirapavee Ganyaporngul,
Nagul Cooharajanone
Department of Mathematics and
Computer Science
Faculty of Science,
Chulalongkorn University
Bangkok, Thailand
6033662623@student.chula.ac.th,
Nagul.C@chula.ac.th

Pravee Kruachottikul
Technopreneurship and
Innovation Management
Program
Chulalongkorn University
Bangkok, Thailand
Pravee.k@gmail.com

Donnaphat
Trakulwaranont*†
*University of Tokyo
Tokyo, Japan

Shin'ichi Satoh†*
†National Institute of
Informatics
Tokyo, Japan

Abstract—Objects with shadow may cause a problem for image classification. For example, it can separate one object into many objects. It can also alter the size or shape of the object resulting in misclassification. In this paper, we focus on removing aircraft shadow from remote sensing images where the shadows occur on wings, bodies, and tails. Since it is very difficult to get shadow-free aircraft images and a shadow aircraft image of the same type for the training part, we adopted Mask-ShadowGAN for solving this issue. The benefit of the Mask-ShadowGAN algorithm is that, in the training part, the technique does not require the same images that have both shadow and shadow-free. In the experiment, we evaluated our proposed technique using RMSE and Jaccard similarity index for measurement. The experimental result shows that our technique shows promising results. We present both best and worst result based on sorted similarity index.

Keywords—Deep Learning, Remote Sensing Image, Mask-ShadowGAN, Shadow Removal

I. INTRODUCTION

Image classification models might poorly perform when working with objects that have the shadow since it can make their shape different from it should be. Removing the shadow without disturbing other detail on it can ease the image classification model in many ways. For instance, shadow-free images can help the model correctly classified, while all preserved detail can be used later to identify the object model or type. Also, we can use it to train the machine learning model if you have only shadow images but intend to use shadow-free images.

There are some proposed methods for shadow detection and removal. For instance, A. Gatter [1] proposed an algorithm to identify a self-cast shadow of aircraft while in airborne, which can be removed later by other procedures. Rahman et al. [2] proposed a method to detect and remove shadows in each orthophoto component of UAV-based orthomosaics. Then use the feature-matching technic with help from a commercial software package to create the final orthomosaic.

The aircraft from remote sensing images consist of issues for making classification correctly, such as background complexity, blurring, light scattering, shadows on the body, or chromatic distortion. As we mentioned above, these problems can cause the classification model to defectively classify the aircraft. Aircraft shape might look unusual if the background is too complicated due to many other objects such as stair trucks. In this work, we mainly focus on the shadow because the shadow can make the aircraft's shape being torn apart. The challenging issue is how to remove object shadow without losing the shape of the aircraft. Wei et al. [3] proposed shadow detection and removal as part of image preprocessing by scanning an original aircraft image to create a binary shadow mask. The mask will use to indicate a shadow pixel location, which applies in the removal process. In the present day, Machine Learning techniques can do many things better and faster than any others. Many researchers use Machine Learning to remove object shadow. For instance, Meeboonmak [4] employed Deeply Supervised Salient Object Detection with Short Connections (DSS) proposed by Hou et al. [5] and doing some post-processing to remove shadows from the aircraft while still maintain aircraft shape integrity.

Generative Adversarial Network (GAN) is a popular method to generate a new image based on its individual properties, which involved shadow removal on the image. Many researchers proposed the shadow removal method by using GAN. For example, Wang et al. [6] implemented Stacked Conditional Generative Adversarial Network (ST-CGAN) to detect the shadow and then remove it. Zhang et al. [7] used GAN to inspect residual and illumination of the image to remove the shadow.

Hu et al. [8] proposed a new way to remove the shadow by learning from unpaired shadow and shadow-free data called Mask-ShadowGAN. This model learned to create shadow masks from real shadow and generated shadow-free images to use as guidance for shadow generation, which used to improve shadow-free generator to produce precise shadow-free images.

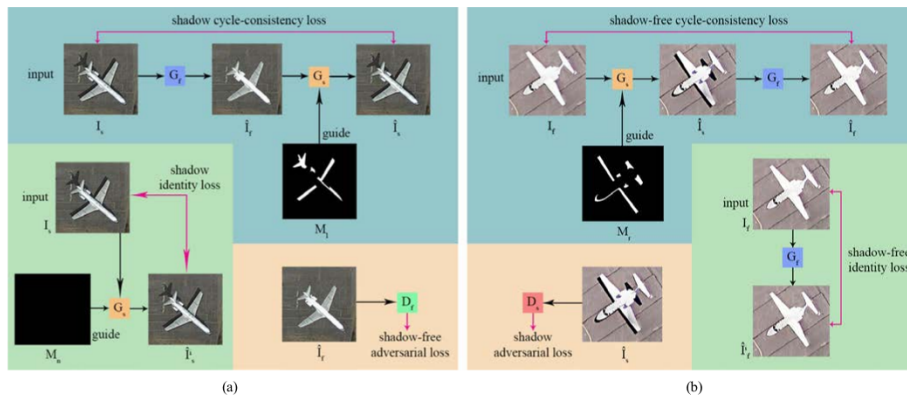


Figure 1. Mask-ShadowGAN for aircraft shadow removal architecture:
(a) Learning from actual shadow images and (b) Learning from actual shadow-free images.

In this paper, we adopt the Mask-ShadowGAN to our aircraft images dataset. Then we use generated shadow-free aircraft images to find aircraft shape in a binary manner to compare with the ground truth shape to see if a generated shadow-free image still preserves intact aircraft shape. There are three reasons why we use this model. First, it uses unpaired images, which suitable for data gathered from Google Earth. Second, the model does not require the same amount of data between shadow and shadow-free images, which is good because shadow-free images are hard to find. Finally, there is no application of this model for aircraft images yet.

II. RELATED WORK

A. Deeply Supervised Salient Object Detection with Short Connections

Hou *et al.* [5] proposed a new framework for salient object detection, which produces the series of short connections for side output from deeper to shallower under the Holistically-nested edge detector (HED) network proposed by Xie *et al.* [9]. Deeper side output has high-level features, which contain more detail out of low-level features. Shallower has low-level features, which can locate the real salient area. A better saliency map can be achieved by combine these two side outputs.

DSS architecture consisted of a base convolution, side-output, short connection, and fusion layer. The base convolution layer dependent on VGG16 proposed by Simonyan *et al.* [10]. To extract the saliency maps, they add

the side-output layers after the last convolutional layer of each stage (conv1_2, conv2_2, conv3_3, conv4_3, and conv5_3) and the final max-pooling layer (pool5) from VGG16. After that, they use the short connection to combine deeper side output with shallower side output. Lastly, the DSS model then optimized by applying fusion loss, which combined from cross-entropy function in each side output.

B. Aircraft Segmentation from Remote Sensing Images using Modified Deeply Supervised Salient Object Detection with Short Connection

Meeboonmak [4] adapted DSS to the aircraft dataset by adding the L2-Normalization layer between the connection of conv1_1 from VGG16 and the shallowest side-output layer, which reduces noise in the background and produces more salient results. Subsequently, post-process then used after obtaining the result from DSS by using Bitwise AND operation between side outputs 5 and 6. Then create the binary mask by thresholding. Next, make the object within the binary mask lager by using the Dilation procedure. Finally, the fused saliency map then combined with the dilated mask.

C. Mask-ShadowGAN: Learning to Remove Shadows from Unpaired Data

Mask-ShadowGAN consisted of two parts: the first part is to learn from real shadow images to produce shadow masks, which used as a guide for shadow generation. The second part is to learn from shadow-free images, which use a random shadow mask to generate shadow on shadow-free images to help training shadow-free generator.

TABLE I. EVALUATION SCORE FOR FIVE HIGHEST RESULTS

Score Type	Image*				
	14.16	19.30	4.66	11.29	12.30
Jaccard Similarity Index	0.9359	0.9349	0.9277	0.9027	0.9001

*. Scores from left to right side are arranged by top to bottom from Figure 2.

TABLE II. EVALUATION SCORE FOR FIVE LOWEST RESULTS

Score Type	Image*				
	12.52	10.61	11.08	16.35	9.73
Jaccard Similarity Index	0.5460	0.5430	0.5059	0.4553	0.4426

*. Scores from left to right side are arranged by top to bottom from Figure 3.

Mask-ShadowGAN has adopted network architecture from Johnson *et al.* [11] as its generator network, which contains three convolution procedures, nine residual blocks with the stride-two convolutions, and two deconvolutions. Each convolution and deconvolution procedures followed by instance normalization. Then used PatchGAN, which proposed by Isola *et al.* [12] as the discriminator. There are two sets of each generator and discriminator.

III. METHODOLOGY

As we mentioned above, Mask-ShadowGAN uses two sets of each generator and discriminator and contains two parts of the operation. There are three losses in each part: cycle-consistency loss (blue), identity loss (green), and adversarial loss (pink). G_s and G_f are generators that create shadow and shadow-free images. Whereas D_s and D_f denote discriminators, which distinguish generated shadow or shadow-free images to be real or not. I_s and I_f denote the actual shadow and shadow-free images. \tilde{I}_s and \tilde{I}_s^i are the generated shadow images, \tilde{I}_f and \tilde{I}_f^i denote the generated shadow-free images, and M_n , M_l and M_r denote the shadow masks as shown in Figure 1.

First, learning from real shadow images by using a shadow-free generator to generate a shadow-free image. Then use a shadow-free discriminator to examine that newly generated image is real shadow-free or not. Next, use the adversarial loss to optimize the generator and the discriminator. After that, they use a shadow generator to convert the generated shadow-free image back to its original and expect to be the same. To do that, they calculate the difference between the generated shadow-free and actual shadow image and binarizing the result to create a mask, which uses as guidance later to preserve the consistency between the generated shadow image, and it's original. They formulate shadow cycle consistency loss, which stimulates shadow generated image to be the same as its original.

For the last process of the first part, they create an image with no new shadows added by using an all-zero values

mask with a real shadow image to be input for the shadow generator. For this reason, they can regularize the generated image to be identical to its original by using the shadow identity loss.

So they can expect that the shadow generated image will have no new shadow added and conserving color composition.

Second, learning from actual shadow-free images to remove shadows by using a shadow generator to create a shadow image with guidance from the randomly selected previously produced mask. They generate many shadow image by using a different mask to generalize the deep model. Those images later are distinguished by the discriminator, whether it is a real shadow image or not. Then optimize the generator and the discriminator using an adversarial loss.

After that, they take a shadow-free generator to use a shadow generated image to create a shadow-free image. Then optimize the network by using shadow-free cycle consistency loss.

For the last process of this part, they put the actual shadow-free image as input into a shadow-free generator to generate another shadow-free image and later encourage the input to be as identical as an output image. To that aim, an identity loss will be used as a constraint so that the shadow-free generator will learn to remove shadows without disturbing the region that is not the shadow.

For the final loss function, they combine all loss from two parts above as a weighted sum of the adversarial loss, cycle consistency loss, and identical loss and then use minimax to optimize the whole of their framework.

Mask-ShadowGAN uses two sets of each generator and discriminator. For generators, they consisted of three convolution procedures, then there are nine residual blocks plus the stride-two convolution and do feature map sampling by including two deconvolution operations. After each convolution and deconvolution procedure, they include the instance normalization operation. As mentioned above, there are two generators in this network. The first one is a shadow generator responsible for creating the shadow on images and requires a shadow-free image and a shadow mask to be put in. The second is a shadow-free generator that is accountable

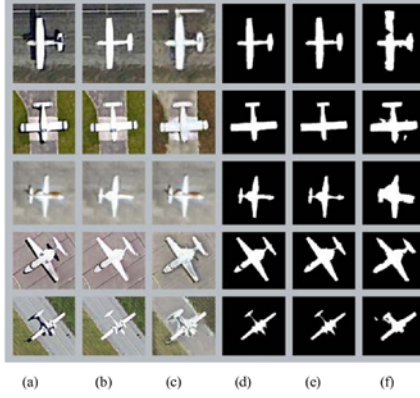


Figure 2. The five highest score results: (a) Original image, (b) Manually shadow removal image (ground truth), (c) GAN generated of (a), (d) Aircraft shape of (b), (e) Aircraft shape of (c), (f) Aircraft shape from Meeboonmak [4]

for generating shadow-free images by feeding three channels of shadow images. Both of these generators create three channels of residual images that will use as the final shadow or shadow-free image with the input image.

For the two discriminators, they use PatchGAN to determine which image patches are valid or not for shadow-free and shadow images.

IV. EXPERIMENTAL RESULTS

A. Dataset

We gathered both shadow and shadow-free aircraft images from Google Earth, which include 1400 and 233 aircraft for shadow and shadow-free images, respectively.

Due to outnumber for shadow-free images, we use the image augmentation process to increase their quantity. We rotate them for 90, 180, and 270 degrees then add them up to the shadow-free collection, which adds up to 932 images. After that, we divide the shadow image set into 84% for training and 16% for testing. We leave no touch for shadow-free collection because it is for training only. We then resize images to 400x400 pixels for use in the training and testing process.

To evaluate, when we get the results from the fed testing images, there is a need for their identical ground truth to be used to test to see if the shadow is gone or not, which is difficult to obtain, especially when images collected from Google Earth. Therefore, we manually remove the shadow from shadow images using Photoshop to create ground truth images instead.

B. Evaluation Result

To evaluate the model's performance, we use root mean square error (RMSE) defined by



Figure 3. The five lowest score results: (a) Original image, (b) Manually shadow removal image (ground truth), (c) GAN generated of (a), (d) Aircraft shape of (b), (e) Aircraft shape of (c)

$$RMSE = \sqrt{\frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i, j) - K(i, j)]^2} \quad (1)$$

where I and K denote the generated shadow-free images and the ground truth images. And we use the Jaccard index defined as

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \quad (2)$$

where A and B denote aircraft shape of generated shadow-free images and aircraft shape of ground truth images. Normally, the lower RMSE, the better result produced, as opposed to the Jaccard index.

We followed the Mask-ShadowGAN shadow removal evaluation by calculating RMSE between the ground truth and the generated shadow-free image in LAB color space. After that, we compute the similarity score between the shape of aircraft in the ground truth and the generated shadow-free image in a binary manner. We do custom thresholding, dilation, and erosion operation to extract most of the shape of the aircraft. We compared our results with the results from the Meeboonmak [4] algorithm. We did not compare with other methods because Hu *et al.* [8] already compared and concluded that's Mask-ShadowGAN provided better results among them.

We collect five of each highest and lowest score by Jaccard similarity from the testing set. The results are shown in Figure 2 and Figure 3 respectively.

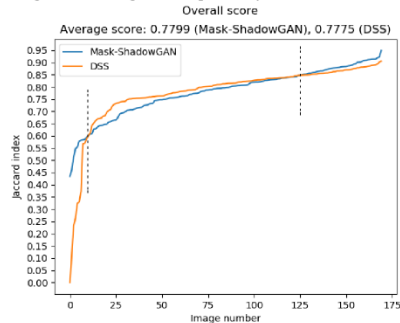


Figure 4. Overall score graph and average score

Then we plot all Jaccard index according to the individual images into a line graph, which represents overall performance, as shown in Figure 4. Also, we indicate the average score of this framework as the graph title.

As we see, the highest score images from Figure 2, the generated shadow-free may not preserve enough detail to be the same as its original, and the RMSE score from Table I is not low as expected. Despite high RMSE, we can see that the Jaccard similarity score is quite high as the model still preserves the aircraft's shape the same as its original, which is good. Compare to ground truth shape, most of the generated aircraft's shape is still similar to the ground truth. When compare to Meeboonmak [4], most of the aircraft's shape is still better intact, while some of our images have a hole or have a torn part like a tail. If we take a look at generated images one more time, we will see that there are some shadow left but thinned, which caused less impact than the full shadow.

For the lowest score images from Figure 3, we can see that the generated aircraft's shape has various issues such as torn or missing parts, bloated or thinned parts, or merge with the background object. If we look into generated shadow-free images, we will see that many images have the artifact, and the shadow is still present but more faded. Despite corrupted things, there exists appropriate stuff like some images can regain its part back, unlike its ground truth. The problem mentioned above, such as the artifact will make aircraft parts look bloated. From the score in Table II, most images perform better when evaluate using RMSE than images in Fig. 3, but the Jaccard similarity score is low due to their ground truth cannot be extracted to complete the shape in some images.

The line graph in Fig. 4 illustrates the overall Jaccard index similarity score of the individual image generated from 2 frameworks: Mask-ShadowGAN (blue) and Meeboonmak [4] (orange), which indicates average scores: 0.7799 and 0.7775, respectively. We divided a graph into three portions to show the area of the lower or higher score of the Mask-

ShadowGAN and Meeboonmak method. In the middle portion, the Meeboonmak algorithm performs quite well. Though, in the first and last part, Mask-ShadowGAN reaches a higher score. We can see that Mask-ShadowGAN achieves better in the average (located above graph), nethermost, and uppermost scores.

V. CONCLUSION

In this paper, we adopted the Mask-ShadowGAN and applied to our aircraft images dataset. From the experimental result, we can see that GAN may not entirely remove the object shadow but still makes it faded out or thinned. Although, the experimental result shows us that generated shadow-free aircraft's detail may be blurred out or blended in with the aircraft body but, its shape is still intact.

In future work, we will improve the model that will thoroughly remove shadows and preserve the object detail. So that the removed shadow aircraft image can be further used for other purposed such as aircraft type classification.

REFERENCES

- [1] A. Gatter, *Classifying self-cast shadow regions in aerial camera images*, 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Auckland, New Zealand, pp. 1-6, 2018.
- [2] M. M. Rahman, G. Mcdermid, T. McKeeman, J. Lovitt, *A workflow to minimize shadows in UAV-based orthomosaics*. Journal of Unmanned Vehicle Systems 7.2, pp. 107-117, 2019.
- [3] W. Wei and J. Zhang, *Remote Sensing Image Aircraft Detection Technology Based on Deep Learning*, 2019 11th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, pp. 173-177, 2019.
- [4] N. Meeboonmak and N. Cooharajanane, *Aircraft Segmentation from Remote Sensing Images using Modified Deeply Supervised Salient Object Detection with Short Connection*. International Conference on Mathematics and Computers in Science and Engineering (MACISE), pp. 184-187, 2020.
- [5] Q. Hou, M. Cheng, X. Hu, A. Borji, Z. Tu, P. Torr, *Deeply Supervised Salient Object Detection with Short Connections*. IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 815-828, 2018.
- [6] J. Wang, X. Li, L. Hui, J. Yang, *Stacked Conditional Generative Adversarial Networks for Jointly Learning Shadow Detection and Shadow Removal*. In CVPR, pp. 1788-1797, 2018.
- [7] L. Zhang, C. Long, X. Zhang, C. Xiao, *RIS-GAN: Explore Residual and Illumination with Generative Adversarial Networks for Shadow Removal*. AAAI Conference on Artificial Intelligence (AAAI), pp.12829-12836, 2020.
- [8] X. Hu, Y. Jiang, C. Fu, P. Heng, *Mask-ShadowGAN: Learning to Remove Shadows from Unpaired Data*. IEEE International Conference on Computer Vision (ICCV), pp. 2472-2481, 2019.
- [9] S. Xie, Z. Tu, *Holistically-Nested Edge Detection*. 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, pp. 1395-1403, 2015.
- [10] K. Simonyan, A. Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*. International Conference on Learning Representations, arXiv:1409.1556v6, pp. 1-14, 2015.
- [11] J. Johnson, A. Alahi, L. Fei-Fei, *Perceptual Losses for Real-Time Style Transfer and Super-Resolution*. European Conference on Computer Vision, pp. 694-711, 2016.
- [12] P. Isola, J. Zhu, T. Zhou, A. A. Efros, *Image-to-Image Translation with Conditional Adversarial Networks*. In CVPR, pp. 1125-1134, 2017.

BIOGRAPHY



Sirapavee Ganyaporngul

Date of birth: 1 April 1999

Email: 6033662623@student.chula.ac.th

Reserve email: sirapavee@gmail.com

Educational background: Studying a Bachelor of Science program, Faculty of Science, Department of Mathematics and Computer Science, Chulalongkorn University

This senior project is accepted and presented at the 2021 IEEE 8th International Conference on Industrial Engineering and Applications (ICIEA 2021) during April 23-26, 2021.