Bad Weather Effect Removal in Images and Videos

BAD WEATHER EFFECT REMOVAL IN IMAGES AND VIDEOS

BY

PENGFEI P. KAN, B.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

© Copyright by Pengfei P. Kan, August 9, 2018

All Rights Reserved

Master of Applied Science (2018)	McMaster University
(Electrical & Computer Engineering)	Hamilton, Ontario, Canada

TITLE:	Bad Weather Effect Removal in Images and Videos
AUTHOR:	Pengfei P. Kan B.Eng., (Automation Engineering) Harbin Institute of Technology, Harbin, China
SUPERVISOR:	Dr. T. Kirubarajan
NUMBER OF PAGES:	xii, 49

To my father Mr. Jian Kan and my mother Mrs. Xiuhua Shen. Thanks for always supporting me!

Abstract

Commonly experienced bad weather conditions like fog, snow and rain generate pixel intensity changes in images and videos taken in outdoor environment and impair the performance of algorithms in outdoor vision systems. Hence, the impact of bad weather conditions need to be processed to improve the performance of outdoor vision systems.

This thesis focuses on three most common weather conditions: fog, snow and rain. Their physical properties are first analyzed. Based on their properties, traditional methods are introduced individually to remove these weather conditions' effect on images or videos. For fog removal, the scattering model is used to describe the fog scene in images and estimate the clear scene radiance from single input images. In this thesis two scenario are discussed, one with videos and the other with single images. The removal of snow and rain in videos is easier than in single images. In videos, temporal and chromatic properties of snow and rain can be used to remove their impact. While in single images, traditional methods with edge preserving filters were discussed.

However, there are multiple limitations of traditional methods that are based on physical properties of bad weather conditions. Each of them can only deal with one specific weather condition at a time. In real application scenarios, it is difficult for

vision systems to recognize different weather conditions and choose corresponding methods to remove them. Therefore, machine learning methods have advantages compared with traditional methods. In this thesis, Generative Adversarial Network (GAN) is used to remove the effect of these weather conditions. GAN performs the image to image translation instead of analyzing the physical properties of different weather conditions. It gets impressive results to deal with different weather conditions.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor Dr. T. Kirubarajan for his guidance, encouragement and support. Thank you Dr. Kiruba for your patience and help in the past two years.

I would also like to thank Dr. Ratnasingham Tharmarasa, Dr. Qingsong Wu, Dr. Yuanhao Yu, Yinghui Wang and Keqi Wei for their invaluable help and support and for providing me with many great ideas. I thank them for so generously sharing with me the knowledge and skills that I needed to succeed.

I would like to thank my friends David Schonborn, Saeid Rostami, Wenqi Mu, Xiafei Zhang and Xiaoxuan He for helping me transition into life in Canada and for their constant support at McMaster.

Last but not least, I want to thank my parents, all my friends, roommates and all the staff at the ECE department for all their support I relied on.

Notation and abbreviations

A — Intensity value of general sky light in images

A_h — Horizontal angle of view

A_v — Vertical angle of View

b — Breadth of rain or snow streaks

CNN — Convolution Neural Network

D — The possibility that the input image is ground truth

e — Euler's number, the base of the natural logarithm

f — Focal length of the camera

$G(z)$ — The generated image

GAN — Generative Adversarial Network

GA — Gaussian Filter

GF — Guided Filter

$I(x)$ — Intensity value of input image at pixel x

$I(x, y)_t$ — Intensity value of pixel at position (x, y) of frame t

I_{ref} — Intensity of reference image

$J(x)$ — Intensity value of recovered image at pixel x

min_c — Minimum intensity value among RGB channels

$M(x)$ — Intensity of rain or snow mask image at pixel x

n — Number of raindrops or snowflakes in unit volume

N — Number of raindrops or snowflakes

$p(x)$ — Intensity value of Perlin noise at pixel x

p_s — Size of the camera sensor pixels

prb — Probability density function of raindrops or snowflakes

Q — Precipitation intensity in millimeter per hour

r — Radius of raindrops or snowflakes

RGB — Red, green and blue channels in images

$t(x)$ — Intensity value of atmospheric transmission at pixel x

t_0 — The threshold value for computing the recovered clear image in fog removal

v — Terminal velocity of raindrops or snowflakes

θ — Falling orientation of raindrops or snowflakes

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

In computer vision, a common assumption is that the air is clear. Therefore, the intensity of each pixel in the image reflects the true brightness of the scene. Many existing algorithms are designed under this assumption. However, for outdoor vision systems, including the ones for tracking, navigation and surveillance, the impact of bad weather like fog, snow and rain need to be considered.

Fog can be classified as steady weather condition as its effect does not change with time. Fog is composed of particles and water droplets that float in the air. The appearance of fog in images is the result of interaction between lights from objects and the atmosphere that contains millions of water particles. Fog scatters and attenuates light from objects in the background and blur the scene of images. In contrast, rain and snow are dynamic weather conditions. Raindrops and snowflakes generate effect that changes with time in images or videos. Snow and rain contain snowflakes and raindrops that fall from the sky and usually have high terminal speed. And their high velocity make them appear as snow or rain streaks instead of individual snowflakes or raindrops in images or videos [4].

All these bad weather effects will degrade the quality of images and videos taken by the vision systems and will impair the performance of the algorithms in these systems. For example, fog decreases the scene depth of the image and obscures the objects in the field of view. Snow and rain generate snow streaks or rain streaks, which impair the detection of the edges or contours and cause many false alarms in tracking algorithms. Therefore, the goal here is to find applicable methods to remove the effect of these bad weather conditions from images and videos.

1.2 Previous Research

1.2.1 Fog Removal

Many methods have been proposed for fog removal with single images. It is an under-constrained problem and contains many unknown parameters that need estimating. In [5], Srinivasa et al, exploited the physics-based model to remove fog in images. Their method requires to select the sky region, a good color region and point out the vanish point in the image before it recovers the clear scene. To estimate parameters from input images instead of manual selection [6, 1] adapted the following scattering model to describe the foggy scene in images

$$I(x) = J(x)t(x) + A[1 - t(x)] \tag{1.1}$$

where $I(x)$ is the observed fog image intensity at pixel x and A is the general atmospheric light. $J(x)$ is the clear scene radiance, which is what we want to recover. $t(x)$ is the atmospheric transmission, which describes the exponential relationship with the scene depth.

$$t(x) = e^{-\beta d(x)} \tag{1.2}$$

where β is the scattering coefficient of the air, $d(x)$ is the scene depth at position x in images.

The scattering model in equation (1.1) describes two parts of light that reaches the camera. The first term $J(x)t(x)$ is the scene light attenuated by fog, and the second term $A[1 - t(x)]$ is the scattering of the general atmospheric light. In equation (1.1), only the observed image $I(x)$ is known. To recover $J(x)$, the main task is to figure out the atmospheric light A and the atmospheric transmission $t(x)$. It is challenging to estimate $t(x)$ since a single image does not offer sufficient information of the scene depth. In [1], He and et al, used the dark channel prior and image matting to estimate and refine atmospheric transmission $t(x)$. [3, 7, 8, 2] made the assumption that $t(x)$ should preserve edges with large depth jumps while pixels between sharp edges with similar scene depth should be smoothed. This is reasonable as objects with different scene depth usually have sharp edges to separate them. So, [3, 7, 8, 2] used edge preserving filters to estimate $t(x)$ successfully. In [3], Tarel et al, used median filter and He et al, in [2] applied guided filter and Gibson in [9] chose local adaptive Wiener filter to refine $t(x)$. Yu et al, used bilateral filter and weighted least square filter to estimate $t(x)$ in [7, 8].

1.2.2 Snow and Rain Removal

Compared with steady weather conditions like haze or fog, snow and rain are more complicated to remove from images or videos. Significant amount of research has been done on these dynamic weather conditions. Kshitiz and Shree [6, 5, 10, 11, 4] have focused on the relationship between vision and rain. They analyzed the physical properties of raindrops and the photographic and chromatic properties of rain streaks that appear in images and videos. Their research also focused on the relationship between camera parameters and the appearance of rain streaks. And they built a model to describe rainy scene that can be captured by camera. Furthermore, Kshitiz and Shree also proposed a novel method that adjusts the camera parameters to reduce the effect of rain when taking pictures. Most of their research focused on the removal of rain in videos, which has the temporal properties, as consecutive frames in videos offer prior information. Based on their research, [12, 13, 14] took advantage of this temporal dependency of videos to remove rain and snow.

However, it is harder to address this problem in single images as they lack the prior information that exists in videos. Xu et al, [15, 16] focused on the edge features of rain streaks and applied the Guided Filter [2] to remove rain or snow streaks from single images. But the drawback of Xu's method is that it blurs the background and cannot remove snowflakes that appear as white dots in images. Peter [17] proposed another method that processed rain and snow in frequency space as rain or snow streaks belong to the high frequency domain.

With the development and success of machine learning algorithms in recent years, they are used to address the weather effect removal problem. Kang et al, and Yang et al, [18, 19] both introduced dictionary learning and sparse coding to remove rain streaks in images. And [20, 21, 22, 23, 24] all applied deep convolutional neural network (CNN) to address this problem and got impressive results. There are two main challenges in their models. One is to build the network, and the other is to get the training dataset. In order to have training dataset, Fu et al, [20, 21], Zheng et al, [23] and Zhang et al, [24] generated synthetic rainy images from clear images so that they can have the ground truth to conduct the supervised training of CNN.

1.3 Organization

This thesis is organized as follows. In the first section, the background and previous research of this problem are discussed. In section 2, traditional methods based on the physical properties of various bad weather conditions are discussed. In section 3, the machine learning method of the Generative Adversarial Network is introduced. And finally, the conclusion is given in the last section.

Chapter 2

Traditional Methods for Bad Weather Removal

2.1 Introduction

In this chapter, methods based on the physical properties of fog, snow and rain are discussed. They are defined as traditional methods because these methods take advantage of special properties and features of each weather condition to remove weather impact in images or videos. Three traditional methods are introduced individually for fog, snow and rain removal.

For fog removal, the proposed method used scattering model to describe the fog image and applied Gaussian filter to refine the estimation of medium transmission $t(x)$. The maximum intensity value of the input image is chosen as the estimation of general sky light A. The proposed method is easier to apply and faster to process the removal of fog.

2.2 Fast Fog Removal of Single Image in the Log Space

To describe the fog scene in images, the proposed method adopted the scattering model in equation (1.1), where the atmospheric transmission $t(x)$ and the general atmospheric light A are the unknown parameters that need to be uncovered. In the proposed method, the atmospheric transmission $t(x)$ is first estimated and refined. Then the general atmospheric light A is estimated from the input fog image. Finally, with both these estimated parameters, the clear scene radiance $J(x)$ and the recovered foggy images can be calculated with the equation (1.1). Additionally, in the method proposed in this thesis, all images are normalized with intensity value range of (0, 1].

2.2.1 Estimating the Atmospheric Transmission $t(x)$

In the proposed method, atmospheric veil was first introduced to get the coarse estimation of atmospheric transmission $t(x)$. It is defined as [3]

$$V(x) = 1 - t(x) \tag{2.1}$$

Considering its physical properties, the follow constraints can be applied

$$V(x) \leq min_c I(x) \tag{2.2}$$

where $min_c I(x)$ is the minimal intensity in all the RGB channels of observed image at position x. Inspired by [1, 7], the atmospheric veil is estimated as

$$V(x) = \alpha \times min_c I(x) \tag{2.3}$$

The coefficient α in this paper is chosen as 0.95 to get a satisfying estimation result. Then the coarse estimation of $t(x)$ is

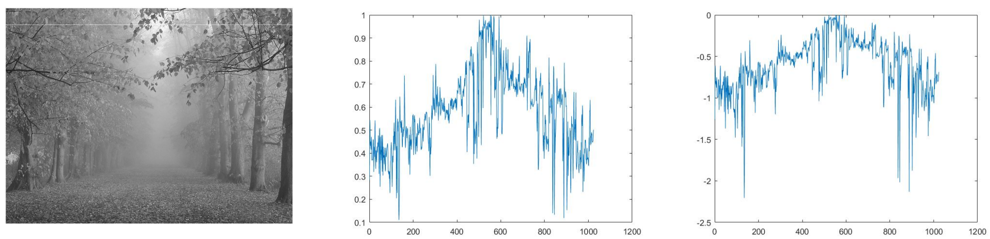
$$\hat{t}(x) = 1 - \alpha min_c I(x) \tag{2.4}$$

In the scattering model of equation (1.1), $t(x) = e^{-\beta d(x)}$ is exponential to the scene depth. It means that objects with the similar scene depth should have similar transmission value, while objects with different scene depth should have distinguished values of $t(x)$. In most situations, objects with different scene depth have sharp edges shown in the image to separate them. In other words, edges in $t(x)$ can be the boundary of the depth. So to estimate $t(x)$ more accurately, it is necessary to smooth pixels between sharp edges and preserve the edges that indicate huge depth jump at the same time. This is the reason why [3, 7, 8, 2] all applied edge preserving filters to refine the estimation of $t(x)$.

A more intuitive and faster way to do the edge preserving smoothing to refine $t(x)$ is introduced in this thesis. $t(x)$ can be transformed into log space as

$$logt(x) = -\beta d(x) \tag{2.5}$$

This shows the linear relationship between the scene depth $d(x)$ and the value of



(a) (b) (c)

Figure 2.1: Intensity Difference in Log Space: (a) grey scale image, (b) intensity distribution in common space, (c) intensity distribution in log space.

$t(x)$. The depth boundaries, like sharp edges, have intensity difference with their neighbor pixels. And in this thesis, all images are treated in double type, which means $t(x) \in (0, 1]$. Since the log function has very steep figure in this range, it enlarges the intensity difference between the edge pixels and their neighborhood. In Figure 2.1, one row of the gray scale image is sampled to compare difference between common and log space, which is indicated by the white line on the top of the image. Images (b) and (c) show intensity values of pixels on this line individually in common space and log space. It is shown that pixels in log space have larger intensity value difference around edges. The covariance of these pixels' intensity is about 0.0980 in log space, while covariance of the pixels in common space is only 0.0282.

In my method, Gaussian filter is used to smooth $t(x)$ in log space. Gaussian filter puts more weight to the pixels closer to the center of the filter kernel, it can smooth pixels close to each other with similar intensity value. Besides, in log space, the intensity difference between edges and their neighborhood has been enlarged. Hence, after the smoothing with the Gaussian filter, most sharp edges are still preserved.

Based on the above analysis, the raw estimate $t(x)$ was first transformed into the log space

$$L(x) = log\hat{t}(x) \tag{2.6}$$

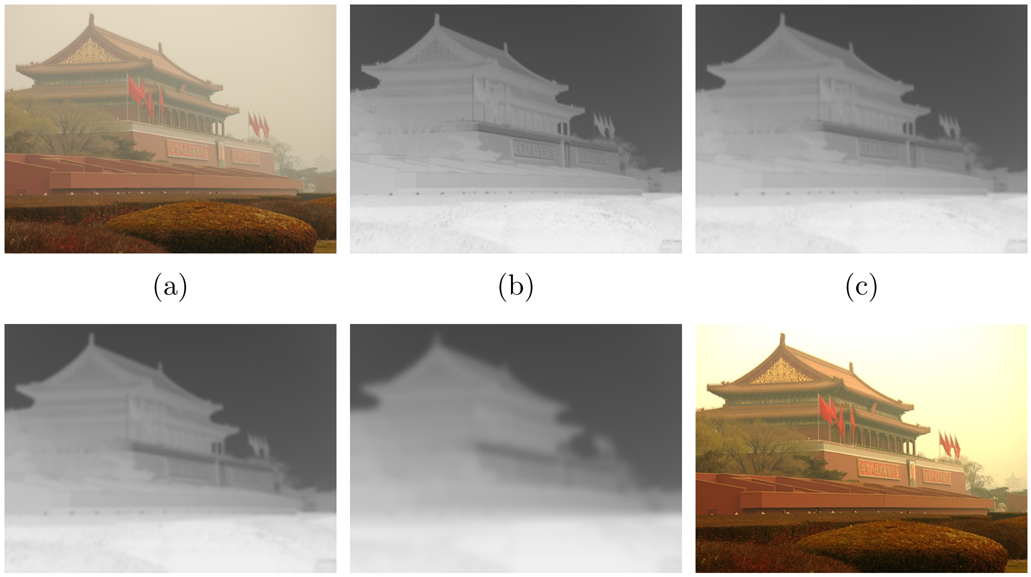
Then the Gaussian filter is applied to smooth $L(x)$

$$B(x) = GA(L(x)) = \frac{1}{|w_i|}\frac{1}{2\pi\sigma^2}\sum_{x_i \in w_i} e^{\frac{-l(x_i)^2}{2\sigma^2}} \tag{2.7}$$

where w_i is the patch of the image centered at position x_i, $|w_i|$ is the number of pixels in the patch, σ is the standard deviation of pixels in patch w_i and $l(x_i)$ is the position distance between the center x_i and other pixels in the patch. After smoothing, $t(x)$ needs to be transformed back to the common space.

$$t(x) = e^{B(x)} \tag{2.8}$$

In Figure 2.2, (a) is the original image. (b) is the refined estimation of $t(x)$ by applying 5×5 Gaussian filter. (c) is the refined $t(x)$ by applying 5×5 Gaussian filter three times. (d) is $t(x)$ refined by 15×15 Gaussian filter and (e) is refined by Gaussian filter with 41×41 kernel. It can be seen that (b) still contains too many sharp edges and objects in the same scene depth are not well smoothed. Image (e) is over-smoothed; the whole depth scene information is ruined. (c) and (d) are both acceptable. So, in the proposed method, small size Gaussian filter is applied multiple times in log space to get better refined estimation of $t(x)$.



(a) (b) (c)

(d) (e) (f)

Figure 2.2: Atmospheric Transmission $t(x)$: (a) input fog image, (b) $t(x)$ refined by 5×5 Gaussian filter, (c) $t(x)$ refined three times by 5×5 Gaussian filter, (d) $t(x)$ refined by 15×15 Gaussian filter, (e) $t(x)$ refined by 41×41 Gaussian filter, (f) recovered clear scene

2.2.2 Estimating the Atmospheric Light A

After estimating the atmospheric transmission $t(x)$, the general atmospheric light A still needs to be estimated to remove the fog. In images with sky area, A is the intensity of the sky pixels. In [7, 8], Yu et al, used the Canny edge detector and set brightness threshold to find the sky area to estimate A. However, this is only applicable for images containing sky areas, and it is computationally expensive. He et al, [1] estimated atmospheric light A with the dark channel prior. In [3], Tarel et al, first balanced the brightness of the image and then set A as the largest intensity in the image. In [25], Tan et al, directly estimated A as the maximum value of image pixels' intensity range.

Upon observation, images taken under foggy weather look brighter than the same scene under clear weather. This is because fog particles scatter and attenuate light and increase the average brightness of the whole scene. Hence, the scene that is recovered from the foggy image should have smaller brightness. So, if the atmospheric light value A is estimated as the largest intensity value in fog image, the recovered scene will be too dark. As a result, [3] applied tone mapping to make the results more visually pleasing.

Hence, to get more natural recovered image, the proposed method first found the largest intensity value t_{max} of the input image and then estimated the general atmospheric light as $A = k \times t_{max}$, where $k \in (0.8, 0.95)$. The smaller k will make the recovered scene brighter. In the method proposed in this thesis, $k = 0.85$ so that the result is natural and visually appealing.

2.2.3 Restore the Fog-free Image

After the estimation of atmospheric light A and the estimated atmospheric transmission $t(x)$, the fog-free scene radiance can be recovered based on equation (1.1).

$$J(x) = \frac{I(x) - A}{max(t(x), t_0)} + A \tag{2.9}$$

where t_0 is the lower boundary in case the denominator is zero. In this experience, $t_0 = 0.01$. And the restoration result of the proposed approach is in Figure 2.2 (e).

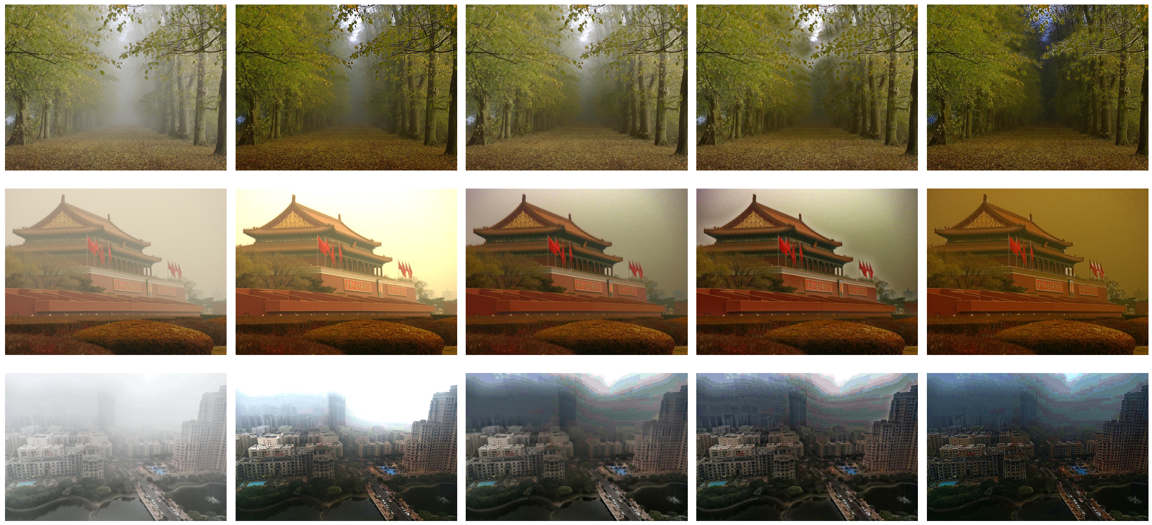


Figure 2.3: Fog Removal Results: columns from left to right, the original image, recovered scene by our method, the recovered scene by dark channel method [1], the recovered scene by guided filter [2], the recovered scene by median filter [3].

2.2.4 Result and Analysis

In the method proposed in this thesis, Gaussian filter was applied multiple times with small kernel size so that sharp edges can be preserved while pixels between edges will be smoothed. This generates better estimation of $t(x)$. In results of this thesis, Gaussian filter with kernel size of 5×5 was applied three times to refine atmospheric transmission $t(x)$. The result was compared with the output of methods proposed in [1, 2, 3].

Figure 2.3 shows the fog removed images using the proposed method in comparison with results obtained using He and Tan's methods. The proposed method in this thesis have brighter recovered scene, less blue hue in the sky area and less halo artifacts around edges compared with other results. Furthermore, the proposed method is significantly faster than existing ones. It takes only 0.487 seconds to deal with 1024×768 image with Intel Core i7 7th Gen 7700HQ (2.80 GHz). It is faster than [3] without tone mapping, approximately 8 times faster than [2] and nearly 100 times faster than [1].

2.3 Snow Removal in Videos

2.3.1 Temporal Property of Snow and Rain Fall in Videos

Compared to single images, videos are composed of multiple frames, which offer us more information. And the temporal properties of falling snow or rain in the video can be taken as the prior information to remove the effect of snow or rain. Snowflakes and raindrops both have high terminal speed, compared with the exposure time of common cameras. They generate considerable falling distance during the exposure time. Hence, snowflakes or raindrops usually appear as snow or rain streaks in images. Their high terminal velocity avoids the same streak to appear at the same position in consecutive frames. If the pixel $p(x, y)$ contains rain or snow at current frame in videos, the pixels at the same position in the previous and next frame will not be affected by rain or snow. Furthermore, due to the random distribution of raindrops and snowflakes, one pixel in the video will not always be covered by rain or snow streaks [10].

2.3.2 Chromatic Property of Snow and Rain Fall in Videos

Snow and rain can be detected in images or videos because snowflakes and raindrops generate intensity changes in the background scene. As snowflakes and raindrops reflect the light from the environment towards the camera, they increase the intensity in all RGB channels. They will make the pixels they pass look brighter than the background and appear white in color.

According to research by Garg and Nayar [4] on vision of rain, individual raindrops without motion blur will generate intensity changes that are independent of the background intensity value. If the raindrop causes motion blur in the image, the rain streak will have linear relationship with the background intensity. Furthermore, in [14], Zhang et al, observed that the intensity changes caused by rain streaks are different than that caused by object motion. Intensity changes in all red, green and blue channels caused by rain streaks have approximately the same value, while those caused by object motion are significantly different. Suppose the intensity of pixels in the current frame is $I_t(x, y)$ and the intensity of the same position in the previous and next frame are $I_{t-1}(x, y)$ and $I_{t+1}(x, y)$. Based on the temporal property of snow and rain, if the pixel of current frame contains snow or rain, then $I_{t-1}(x, y)$ and $I_{t+1}(x, y)$ will not be polluted by rain or snow. If the background is stable, then $I_{t-1}(x, y) = I_{t+1}(x, y)$. So the absolute intensity changes of $I_t(x, y) - I_{t-1}(x, y)$ and $I_t(x, y) - I_{t+1}(x, y)$ should be the same in RGB channels. If the background is moving, only pixels around edges will have significant intensity changes.

2.3.3 Snow Removal in Videos

Removing snow from videos is based on the above analysis of the temporal and chromatic properties. Considering the temporal property, the proposed method used three consecutive frames to detect snowflakes. To filter the false alarms of object motion, the chromatic property is applied to find true snowflake detections instead of the object motion pixels.

Taking three consecutive frames, a threshold of the intensity change is set to detect the candidate pixels of snowflakes and snow streaks [10]

$$I_t(x,y) - I_{t-1}(x,y) > c, I_t(x,y) - I_{t+1}(x,y) > c \tag{2.10}$$

where $I_c(x,y)$ is the intensity of pixel at (x,y) of current frame, and $I_p(x,y)$ is the intensity of pixel at the same position of the previous frame. c is the threshold value and in this experience, it was set at 3 following Nayar's method.

To filter out more accurate result, two intensity changes between the three consecutive frames are computed.

$$D_{t-1,t} = I_t(x,y) - I_{t-1}(x,y), D_{t,t+1} = I_t(x,y) - I_{t+1}(x,y) \tag{2.11}$$

These two intensity changes in each channel should have similar values, otherwise, they are treated as false alarms caused by object motion.

To remove these detected snow pixels in current frame, their intensities were replaced by the average intensity value of pixels at the same position of the previous and next frames.

$$I_{new}(x,y) = [I_{t-1}(x,y) + I_{t+1}(x,y)]/2 \tag{2.12}$$

2.3.4 Result and Analysis

This introduced method is tested with both stable background videos and moving background videos. In Figure 2.4, the first row shows results from stable background, it was taken by a cellular phone in the campus of McMaster University. The background is the Information Technology Building. The result shows that this method

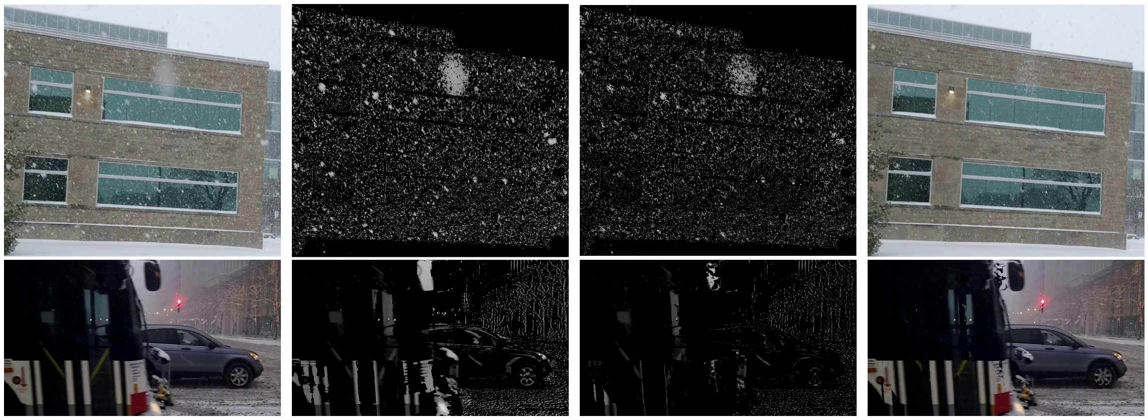


Figure 2.4: Snow Removal in Videos: columns from left to right, original image, detected candidate pixels with snow, refined pixels with snow, snow removed image. The first row is images with stable background and the second row is images with moving background

successfully detected most of the snowflakes in the image and obtained clear snow-removed result. The second row shows a video taken on the street, and the background is moving vehicles. The second image in the second row shows that using only the intensity change, candidate pixels of snow contain many false alarms caused by the motion of vehicles. After applying the chromatic property filtering, most of the false alarm pixels caused by object motions such as pixels on the car's front door and the bus's door are removed.

There are several drawbacks of the introduced method. One is that if the moving background is white such as the white bus showed here, it will be difficult to distinguish between pixels of snow and moving objects in background. Figure 2.4 also shows that there are more false alarms in the area of the white bus. Another problem is that this proposed method does not work well for rain or heavy snow. Compared with snowflakes, raindrops are more transparent and have higher terminal speed. They appear as rain streaks in images and cause motion blur and their intensity has linear relationship with the background's intensity. These properties make rain streaks difficult to detect as the intensity difference is too small to set an appropriate threshold to detect them. Furthermore, the defocus affects the visibility of rain streaks significantly [11]. When the camera focuses on specific objects in the rainy scene, rain streaks that are out of focus do not appear in the image. Hence, it is hard to detect rain streaks in videos.

2.4 Single Image Snow and Rain Removal

In videos, temporal and chromatic properties of snow and rain offer the prior information that can be used to remove their effect but those properties do not exist in single images. So the problem needs to be addressed in other ways. Peter et al, offered a novel idea for solving this problem [17] in frequency space. He processed single images as signals composed of both high and low frequency components. His method is independent of other supporting frames and is not affected by the background motion. Other methods focus on streaks generated by snow and rain in images. These streaks are edges and can be treated as high frequency component of the image while the background belongs to the low frequency part. So, Xu et al, [15] and Zheng et al, [26] used methods that segmented rain or snow images into high and low frequency parts to smooth out the streaks in the high frequency domain.

In [15], Xu et al, mentioned that rain streaks have higher intensity value than that of background pixels. Their color is close to white. This means that the intensity value of rain streaks in RGB channels are similar. In contrast, background pixels, especially edges, have significantly different intensity value in each RGB channels. So

Xu et al, computed the reference image for guided filter as

$$I_{ref}(x,y) = I_{b_{max}}(x,y) - I_{b_{min}}(x,y) \tag{2.13}$$

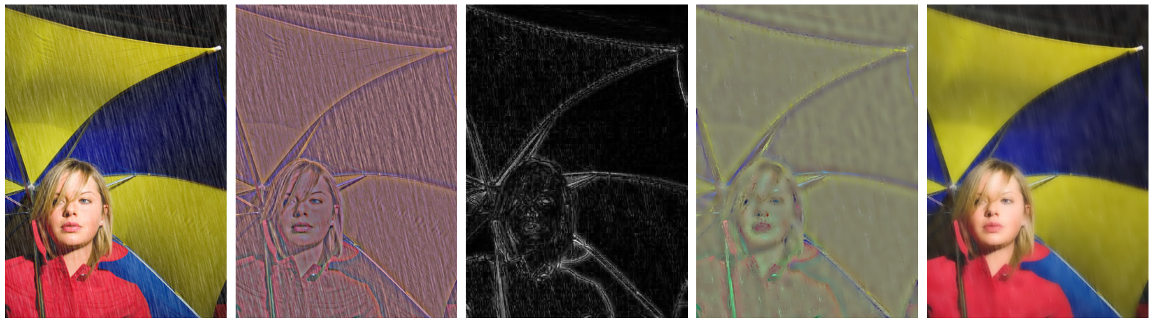
where $I_{b_max}(x,y)$ is the maximum intensity value among all RGB channels at position (x,y) and $I_{b_min}(x,y)$ is the minimum intensity value. The reference image did not have the rain streaks while preserving background edges. Then this reference image is used for the guided filter to smooth the original image. However, the reference image is not accurate enough with all background details. After applying guided filter to smooth the input image, the result is blurred.

Zheng et al, [26] used a similar method but he optimized it by using multiple guided filters. Their method first separated the high frequency part from the low frequency background with the guided filter. Then both the high frequency and low frequency parts are refined. Rain streaks in high frequency are smoothed with guided filter again. Edges in low frequency part are extracted as the final reference image. Method of Zheng et al, gets better result than that of Xu et al, but it is too computationally expensive.

2.4.1 Fast Single Image Rain and Snow Removal

The proposed method is the combination and compromise of methods mentioned above. It first applied guided filter to get the low frequency component of the input image. The high frequency part is subtracted from the input image by the low frequency part.

$$H(x,y) = I(x,y) - GF(I(x,y)) \tag{2.14}$$



(a) (b) (c) (d) (e)

Figure 2.5: High and Low Frequency Components: from left to right, input image with rain, high frequency component of the image, reference image for the high frequency, rain removed high frequency component, recovered image.

where $I(x, y)$ is the input image with rain or snow. The rain or snow streaks are contained in this high frequency component. The proposed method also applied guided filter to smooth away rain streaks and the reference image of the guided filter is computed based on equation (2.13). After the high frequency component was smoothed, it was added back with the low frequency component as the output.

The method proposed in this thesis keeps more details of the input image as it only processed the high frequency part. And it is much faster compared with Zheng's method.

2.4.2 Result and Analysis

Image (b) in Figure 2.5 shows that the reference image removes most rain streaks while keeping other non-streak edges of the background. After applying guided filter, image (c) shows that most of the rain streaks disappear in the high frequency component image.

The proposed method was tested with real snow and rain images. In Figure 2.6,

(a) (b) (c) (d)

Figure 2.6: Snow and Rain Removal in Single Images: (a) input image with snow, (b) snow removed image by the proposed method, (c) input image with rain and (d) rain removed image by the proposed method.

it can be seen that the proposed method removed most of the snow and rain streaks successfully. However, the drawbacks of the proposed method is also obvious. It performs better to remove snow streaks than rain streaks. As shown in images (c) and (d) of Figure 2.6, many rain streaks far away in the scene still remain in the image. This is because rain streaks cause motion blur and have non-linear relationship with the background intensity, which makes them difficult to be filtered away. Furthermore, the removal of snow or rain streaks caused the loss of some edge information in the background. It can be seen from Figure 2.6 that the name board became blurred in the processed image (b).

Chapter 3

Deep Learning Method for Bad Weather Removal

3.1 Introduction

Bad weather conditions like fog, snow and rain have their individual physical properties and generate different textures and appearance in images and videos. Traditional methods of bad weather removal focus on weather's features separately and can only deal with one weather condition at a time. However, in real scenarios, outdoor vision systems have no idea of what weather condition is contained in current input. This will limit the application of traditional methods. Moreover, traditional methods are based on the models that describe the features of these different weather conditions. But those models are not accurate enough to uncover all the details of given weather conditions. For example, the rainy scene is a nonlinear summation of the background and rain streaks. During heavy rain, humid air and water droplets splashed by raindrops will generate haze floating near the ground. Then the scene becomes a complicated combination of both rain and fog, which leads both traditional rain removal methods or fog removal methods to fail.

To address this problem, machine learning model has been introduced to translate rainy images into clear images directly. Fu et al. [20, 21] proposed two machine learning models: the traditional convolutional neural network and the ResNet network [27]. To reduce the computation cost of the network, Fu et al. first applied guided filter to segment the mask of rain streaks from the background and then input the rain mask images into the network for training. Their methods have impressive output and they skip the process of detecting rain streaks so that they are more flexible to deal with different levels of rain. But their methods need to subjectively decide the parameters for the guided filter. If inappropriate parameters are chosen for guided filter, it will not get the satisfying results.

Zheng et al. [23] proposed a method to remove the misty rain from single image. He used the convolutional neural network similar to Fu's but built a joint loss function that considered both the rain and the fog features. There are other methods that considered the heavy rain situation. Zhang [24] used a method with the supporting information of density levels. He labeled the density levels of rain in the training dataset, and used a multi-stream dense network to discriminate rain levels and remove rain from images. But none of them considered to deal with all kinds of bad weather conditions together.

With traditional methods, it is challenging to have one model that can deal with all these bad weather conditions as they have different physical properties. Hence, the applicable method is to abandon the focus of specific weather conditions and consider image to image translation. In this thesis, the Generative Adversarial Network is used to transform the images with fog, snow or rain effect into clear images.

3.2 Generative Adversarial Neural Network

In image to image translation, one challeng is to build the loss function to guide convolutional neural network to get the sharp and realistic images. Since this is a high level goal, it is different from dealing with local patches of images. Many traditional popular loss functions will cause the output images to blur [28]. Generative Adversarial Network (GAN) does not have this problem because it deals with images in higher level [29]. It not only trains the network to generate images, but also trains the discriminator to optimize the generator.

The framework of GAN consists of two parts: the generator G and the discriminator D. It is similar to a game between the generator and discriminator. In this case, the generator needs to generate fake images of clear weather as realistic as possible. And the discriminator should be trained to discriminate the fake generated images from the real clear images. Both G and D could only control their own parameters, but can use the output of each other for training. The solution of the game is a Nash Equilibrium [30].

The most common structure of generator G and discriminator D is deep CNN. For the discriminator, it is similar to a classifier. So in [31, 32] the structures of VGG16 or VGG19 were adopted with the addition of batch normalization layer and changing of activation layers. For example, in [33], Christian used blocks consisting of convolutional layer, batch normalization layer and Leaky ReLU layer to build the discriminator. The structure of generator varies a bit for different applications. In [33], He et al. used residual CNN, whose input will be added to the last layer to generate output [27]. This architecture is suitable for deep CNN layers and can avoid degradation. Phillip proposed a new idea of using U-Net [34] structure in [28]. U-Net is a novel architecture that has impressive performance with limited training samples.

Another important component of GAN is the objective function. The classic objective function is in the minmax form [29, 33, 32]

$$\min_{G} \max_{D} J(G, D) = \mathbb{E}_{x\ p_{data}}[logD(x)] + \mathbb{E}_{y\ p_y}[log(1 - D(G(y))] \tag{3.1}$$

where y is the input image with latent random variables, x is the ground truth. D represents the probability that the output image is from the ground truth x instead of from the generated data $G(y)$. The purpose of this loss function is to train D to successfully discriminate generated images between ground truth images. In other words, it trains $D(x)$ to get value near 0 if the input is generated image from generator and have value of 1 if the input is ground truth. At the same time, it trains the G to make $D(G(y))$ close to 1 to pass the discrimination. In other words, the generator is good enough to generate clear images. There are other forms of loss functions for GAN. Zhu et al. proposed a new loss function [31] that used least square error in the minmax function considering the stability of training procedure. Phillip et al. [28] applied $L1$ norm to replace log functions.

3.3 Synthesizing Training Dataset

The GAN model needs training samples that have ground truth of background and different weather images with the same background. In real scenarios, it is impossible to get images in different weather conditions with identical background. The movement of objects, the shaking of the camera and the change of light all cause intensity differences. Hence, to generate training dataset, the practical method is to synthesize weather effect in clear images. In this section, I introduced the proposed methods to generate training dataset.

3.3.1 Fog Rendering

Fog consists of tiny water particles floating in the air. These water particles scatter and attenuate the light reflected from objects. It decreases the visual distance and makes the objects appear blurred. The visibility of a fog scene has close relationship with the scene depth. Objects that are farther away are blocked by more fog particles so that they are less visible. Inspired by the scattering model used in fog removal, the proposed method of synthesizing fog effect adopted it to describe the fog image and synthesize the fog effect. To get better results, Perlin noise [35] at the end is introduced to generate texture of fog.

3.3.1.1 Model of Fog Scene

The scattering model is adopted to describe the fog effect in images.

$$F(x) = I(x)t(x) + A[1 - t(x)] \tag{3.2}$$

where $I(x)$ is the input clear image, $t(x)$ is the medium transmission and A is the general atmospheric light of the whole image. The estimation process of $t(x)$ and A is introduced in the fog removal part in the previous chapter. To simulate the fog

Figure 3.1: For Rendering: from left to right, clear image, medium transmission matrix t(x), scene radiance J, fog added image with 0.5 noise ratio, fog added image with 0.75 noise ratio

view better, Perlin noise [35] is added to get the final output.

$$fog(x) = (1 - \alpha)F(x) + \alpha p(x) \tag{3.3}$$

where $p(x)$ is the Perlin noise, which is a type of gradient noise that can produce natural appearing textures. α is the summation weight.

3.3.1.2 Perlin noise

Perlin noise has random appearance and is widely used in computer graphics to generate synthetic textures. The shape of Perlin noise is determined by its frequency and amplitude. In the proposed method, multiple frequencies and amplitudes were chosen to generate different shapes of the features and they were merged together to simulate the natural appearance of fog.

3.3.2 Rendering Rain and Snow

Rain and snow can be treated as particle systems, since both are made of water or ice particles in the air. In this section, atmospheric properties of raindrops and snowflakes, such as their size distribution in 3D space, terminal velocity and their intensity are first discussed. Then a project model to generate the rain or snow mask images is proposed based on their physical properties. Finally these mask images of snow or rain are added to input clear images to render the rain or snow effect in clear images.

3.3.2.1 Density and Distribution of Rain and Snow in Space

Raindrops and snowflakes are uniformly distributed in the 3D space [36]. In meteorology, people use the precipitation rate (millimeter per hour) to measure the intensity of the precipitation. Rain intensity is classified according to the rate of precipitation. Light rain has the precipitation rate smaller than 2.5 mm/hr. The moderate rain has precipitation among $2.5 - 10$ mm/hr and the heavy rain has precipitation rate more than 10 mm/hr [37]. We also classify snow intensity based on the precipitation rate. The light snow usually has intensity smaller than 1 mm/hr. Moderate snow's intensity is among $1 - 2.5$ mm/hr and heavy snow has intensity larger than 2.5 mm/hr.

3.3.2.2 Raindrop Shape, Terminal Velocity and Falling Orientation

Due to air pressure, when raindrops fall with high speed, their shape will be distorted. The bigger the raindrop, the more flattened the raindrop is at the side of its falling direction. According to Beard and Chuang [38], the shape of the raindrop can be described by a complicated 10th order cosine function. However, most of raindrops have radius smaller than 2mm. With this small size, the distortion of the sphere shape is very tiny [37]. Hence, raindrops can be assumed to have the sphere shape.

Another important property is the terminal velocity of raindrops, since it affects the appearance of the rain streaks in images. As raindrops fall, they will be affected by air resistance, which is proportional to the square of the velocity. Higher speed will cause larger air resistance force and the air resistance force will finally balance with the gravity of raindrops. So, when raindrops reach the ground, they have constant terminal velocity. Approximately, the terminal velocity of raindrops can be estimated as [39]

$$v = 200\sqrt{r} \tag{3.4}$$

where r is the radius of the raindrop. The terminal velocity of raindrops is usually multiple meters per second. When pictures were taken in rain, during the exposure time, raindrops that are close to the camera will move a considerable distance. And they will appear as rain streaks in pictures instead of individual raindrops.

In ideal situation, raindrops fall vertically. But raindrops are very light and can be easily driven by wind. Most of the time, raindrops fall with an angle. The value of the angle depends on how strong the wind is. Sometimes raindrops even fall nearly horizontally, but it is very rare to see in real scenarios. Hence, in the method proposed in this thesis, the vertical direction is defined as 0 degree, and raindrops' falling orientation degree is supposed to range from -45 to 45.

3.3.2.3 Snowflake Shape, Terminal Velocity and Falling Orientation

Compared with raindrops, snowflakes have more complicated shapes. Since snowflakes are forms of ice crystals, they have thousands of different shapes such as plates, needles, dendrites and columns. In this thesis, they are assumed as thin pieces of ice crystal. Additionally, snowflakes usually have bigger size than raindrops. However, since they are ice crystals, they are lighter and more sensitive to wind force. This makes them have smaller terminal velocity compared with raindrops. Based on [40], the terminal velocity of snowflakes can be estimated by the following equation

$$v = \frac{100}{r^{0.2}} \tag{3.5}$$

where r is the radius of the snowflake in the unit of meter.

Since snowflakes are more sensitive to wind force, their falling orientation has more variance among individual snowflakes. In snow fall, snowflakes have similar orientation in general but are not as ordered as raindrops. The kth orientation can be defined as

$$\theta(k) = \theta_{mean} + \theta_{var}(k) \tag{3.6}$$

where θ is the falling orientation of an individual snowflake, θ_{mean} is the general mean and θ_{var} is the variance of orientation degree of all snowflakes in the field of the view.

3.3.2.4 Raindrop Size

Raindrops have widely distributed size. The percentage of raindrops with one specific size is determined by the rain intensity. But generally, based on Marshall and Palmer's research [41], a major percentage of raindrops have radius less than 2 mm. It is rare to see raindrops with radius larger than 5 mm because when raindrops reach this large size, they are easily to get separated when falling in the air. The raindrop size distribution is modeled as

$$n(r) = 8000e^{-82Q^{-0.21}r} \tag{3.7}$$

where r is the radius of the raindrop, Q is the rain intensity given in mm/hr and $n(r)$ is the number of raindrops per unit volume in the space. After taking integral of $n(r)$ over all radius values, the general raindrop number density in the space can be described as

$$n = \int_0^\infty 8000e^{-82Q^{-0.21}r}dr \tag{3.8}$$

For the given rain intensity Q, the probability density function of raindrops' size distribution is described as

$$prb(r) = 82Q^{-0.21}e^{-82Q^{-0.21}r} \tag{3.9}$$

The probability density function of equation (3.9) shows that the majority of raindrops have very tiny size. Although millions of raindrops can be captured by the camera sensor when an image is taken, only small percentage of them will appear as rain streaks in the image.

3.3.2.5 Snowflake size and its distribution

Similar to raindrops, snowflakes also have a distribution that is exponential to the intensity. Based on Sekhon and Srivastava's research [42] the relationship between snowflake intensity and its number density in unit volume is given as

$$n(d) = 250Q^{-0.94}e^{-22.9Q^{-0.45}d} \tag{3.10}$$

where d is the diameter in mm of the snowflake and the Q is the snow intensity in mm/h. Based on the Liquid Water Equivalent (LWE) measurement, the common snow fall intensity is smaller than 5 mm/hr. $n(d)$ is the number density of snowflakes with specific size in unit volume. And the probability density function of this distribution can be get from

$$prb(d) = 1 - e^{-22.9Q^{-0.45}d} \tag{3.11}$$

It is very similar to raindrops's distribution as most of the snowflakes have small size. Common snowflakes have diameter smaller than 10 mm.

3.3.2.6 Projection Model

Both raindrops and snowflakes distribute in 3 dimension space but they are recorded by images and video frames only in 2 dimension space. To render the effect of rain or snow, a projection model is introduced to project raindrops or snowflakes from 3D space into the 2D surface.

3.3.2.6.1 The Appearance Shape of Raindrops and Snowflakes

The shape of rain or snow streaks depends on the size of the raindrops or snowflakes, the cameras parameters and the distance away from cameras' lens. Based on the optics theory [17], if the raindrop or snowflake is in-focus, the breadth of the streak that appears in the camera sensor is

$$b = \frac{r \times f}{z} \tag{3.12}$$

where r is the radius of the raindrop or snowflake, f is the focal length of the camera and z is the distance to the camera. And the length of the streak is

$$l = (r + vt)\frac{f}{z} \tag{3.13}$$

where v is the terminal velocity of the raindrop or the snowflake and t is the exposure time of the camera.

Besides, raindrops and snowflakes are transparent. They reflect large angle of light from the background and blur the area where they pass. Hence, their streaks is modeled as Gaussian motion blur

$$g(x,y) = \int_0^l e^{-\frac{(x-a\times\cos(\theta)-\mu_x)^2+(y-a\times\sin(\theta)-\mu_y)^2}{b^2}} da \tag{3.14}$$

where (μ_x, μ_y) is the start position of the streak in the image. b is the breadth of the streak, l is the length and θ is the orientation.

3.3.2.6.2 Field of View

Every image has its corresponding camera parameters such as the sensor size, the focal length and the exposure time. With these parameters, the field of view of the image can be specified. Since only raindrops or snowflakes that can be captured by the camera sensor matter, the distance of the field of view is computed as

$$Z_m = \frac{2r_m f}{p_s} \tag{3.15}$$

where r_m is the maximum radius of raindrops or snowflakes. And f is the focal length, p_s is the size of the sensor pixels of the camera. To appear in images, rain or snow streaks should cover at least one pixel on the camera sensor. So, raindrops or snowflakes that are farther away than Zm are ignored as they will not appear in images. Furthermore, the angle of the view is calculated as

$$A_h = 2\arctan\frac{w_s}{2f} \approx \frac{w_s}{f} \tag{3.16}$$

$$A_v = 2\arctan\frac{h_s}{2f} \approx \frac{h_s}{f} \tag{3.17}$$

where A_h is the horizontal angle of view and A_v is the vertical angle of view, w_s and h_s are the width and height of the camera's sensor.

The field of view is modeled as a pyramid area with the camera as the vertex. The total number of raindrops or snowflakes in this space is

$$N = \frac{1}{3} \times 2Z_m \tan(\frac{A_h}{2}) \times 2Z_m \tan(\frac{A_v}{2}) \times Z_m \times n \tag{3.18}$$

where n is the number density of raindrops or snowflakes. But N is not the number of streaks that finally appear in the image. Since raindrops and snowflakes vary in size, if the particle cannot generate a streak or dot wider than one pixel on camera's sensor, they will not appear in the rain or snow images.

3.3.2.6.3 The Rain or Snow Added Image

Rain or snow streak mask images can be rendered with the above model and then added to input images as

$$R(x) = I(x)\alpha + M(x)(1-\alpha) \tag{3.19}$$

where $I(x)$ is the input clear image, $M(x)$ is the rain mask image. α is the weight used to balance the brightness of the output image so that the result will not look over-whiten. And α is chosen as the general mean intensity value of the input image [43].

To generate the rain or snow mask image $M(x)$, which contains rain or snow streaks and has the same size as the input image, the image field of view is first estimated by equation 3.15, 3.16 and 3.17. Then the number of rain streaks or snow streaks are computed based on equation 3.18. Considering raindrops and snowflakes are uniformly distributed in space, the start positions of streaks are chosen randomly by uniform distribution. The breadth and length of the streak is computed by equation 3.12 and 3.13. At last, the streaks are drawn using equation 3.14 to have the blur appearance.

An image taken by iPhone 5s and the added rain and snow effect are demonstrated in Figure 3.2. The camera sensor size is $4.896 \times 3.673mm^2$, the pixel size is $1.5um$ and the focal length of the iPhones camera is $4.12mm$. The shutter speed is 1/406 second when the image was taken.

3.3.2.7 Weather Effect Added Results

Rendering of the fog effect in the image does not need the camera parameters. But to generate the snow or rain effect, the corresponding camera parameters of the input image are needed to estimate the number and shape of rain or snow streaks. However, for most images the corresponding camera details are unknown and it is hard to estimate those parameters if they are not recorded. In the proposed method, default camera parameter values for the projection model are set to generate weather effect. The popular 35mm full frame sensor with 24 million pixels is chosen as the default camera sensor. The sensor size is 36×24 mm and the pixel size is 6 um. To

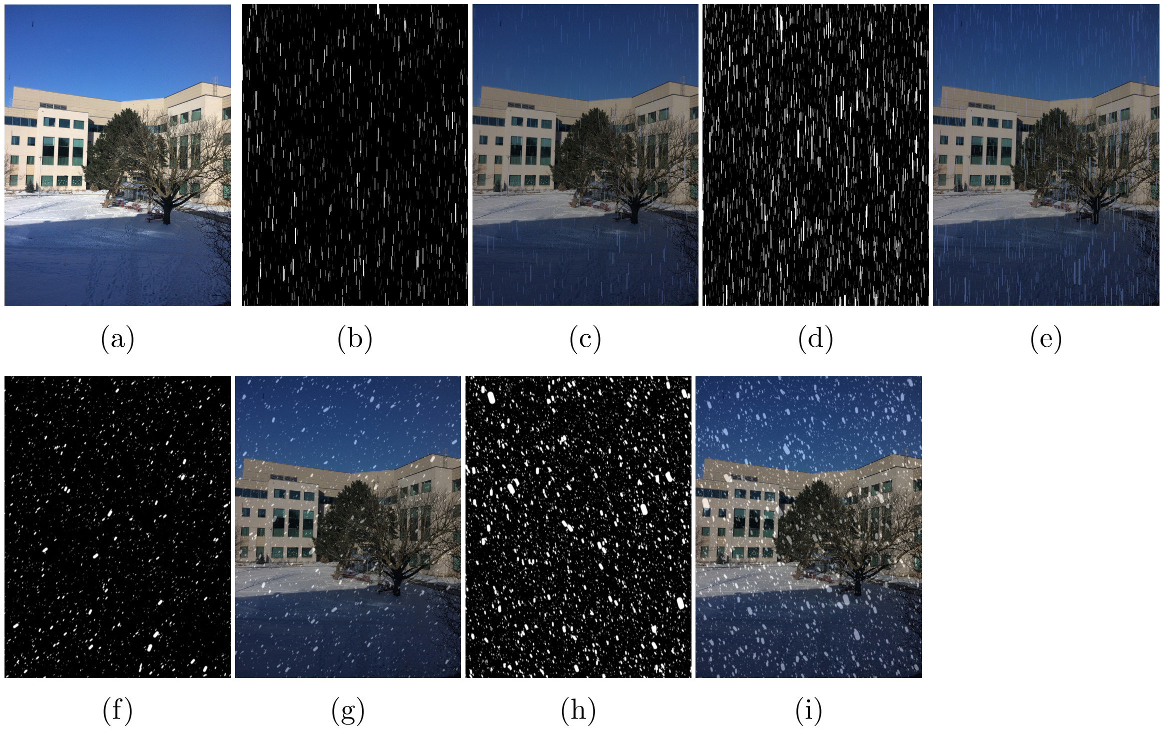


Figure 3.2: Rain and Snow Rendering: (a) clear image, (b) rain mask image with rain intensity of 5 mm/hr, (c) rain added image with rain intensity of 5 mm/hr, (d) rain mask image with rain intensity of 30 mm/hr, (e) rain added image with rain intensity of 30 mm/hr, (f) snow mask image with snow intensity of 1.5 mm/hr, (g) snow added image with snow intensity of 1.5 mm/hr, (h) snow mask image with snow intensity of 4.0 mm/hr, (i) snow added image with snow intensity of 4.0 mm/hr.

have a natural appearance in human view, 45 mm is set as the default focal length since it matches best with full frame camera sensor. Exposure time is chosen based on the shutter speed table from International Standard Organization. Using this projection model, the following weather effect added images are shown in Figure 3.3.

Figure 3.3: Synthesized Weather Effect Results: The first row is clear images, the second row is images with rendered fog, the third row is images with generated rain and the last row is images with snow added

3.4 GAN Network Model for Weather Effect Removal

GAN Network does the image to image translation instead of focusing on the specific weather properties. By training GAN with image pairs that contain ground truth and images with different weather conditions, the GAN model will learn features of rain or snow streaks and the background. Then the well-trained GAN model can generate clear images from input images based on its previous learning.

The framework of the GAN network consists of two main components: the generator and the discriminator. Considering Zhang et al. [44] got impressive result by using GAN to do the removal of rain, this thesis takes reference of his GAN model structure, which is adopted from [45]. The generator uses 6 convolutional layers and 6 deconvolutional layers. For each convolutional and deconvolutional layer, batch normalization is added. Batch normalization will make the parameters of current layer to have zero mean and unit covariance, which will decrease the influence from the previous layers and make the model more stable to train [46]. To improve the stability of the generator, in this thesis, LeekyReLU is used as the activation function for each layer in Generator because LeekyReLU avoids the spare gradient matrix, which damages the stability. Considering the limited training dataset [34] and recovering more details in deconvolutional layers [47], each convolutional layer's output is connected directly with the deconvolutional layer's input.For the structure of discriminator, it is similar to a common classifier. It classifies the input data between the generated group and the ground truth group. So, the discriminator has a deep CNN structure with 5 convolutional layers and the sigmoid layer as the final layer to output the possibility. To improve the stability of the whole GAN model, LeekyReLU is added to the first 4 convolutional layers in this thesis.

The general objective function to train GAN is in equation (1). Both D and G are discriminable functions. G represents the mapping from input image to the output fake image. D represents the probability that the output image is from the ground truth x instead of the generated data $G(z)$. The purpose is to train D to make $D(x)$ close to 0 while train G to make $D(G(z))$ approximate 1. The game between D and G ends at the saddle point of the minmax problem. And this minmax problem can be solved with Stochastic Gradient Descent (SGD) [29]. The generated images and the ground truth images are sampled to update the gradient of the discriminator. On the other hand, the output of the discriminator is also used to train the generator.

Furthermore, since in the image to image translation, the traditional pixel to pixel loss function like Mean Square Error (MSE) cannot learn the high level feature well [48]. Zhang et al. [44] applied a loss function that combined pixel to pixel Euclidean loss, adversarial loss and perceptual loss function for G and D. This thesis also adopted that to measure the high level features in images.

Compared with traditional methods, GAN model deals with heavy rain and snow much better. As shown in Figure 3.4, GAN model uncovers clearer background details and has more natural results while the traditional methods cannot handle the heavy rain and snow well.

3.5 Result and Analysis

The experimental results in Figure 3.4 are real images and the GAN model produces great results. There are no numerical criteria to evaluate the results since it lacks the

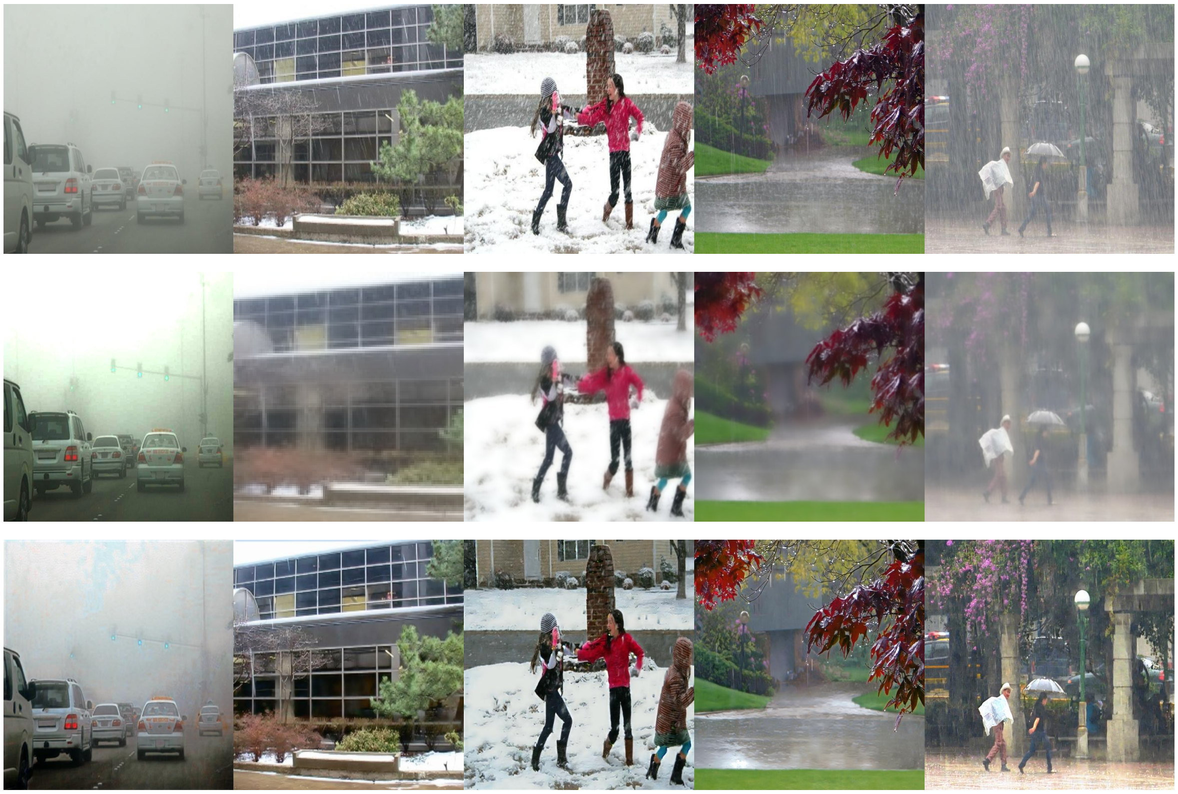


Figure 3.4: Results of GAN: first row from left to right is image with fog, image with light snow, image with heavy snow, image with light rain and image with heavy rain. The second row is images processed by traditional methods using log space method for fog removal and edge preserving filter for snow and rain removal. The last row is results from the GAN model

ground truth to compute common evaluation criteria such as the classic Peak Signal to Noise Ratio (PSNR), Structural Similarity Index [49] and Visual Information Fidelity [50]. These criteria all need ground truth of images to compute the value. Hence, the results could only evaluated visually here. Figure 3.4 shows that the GAN model removes most part of the fog, snow and rain effect in single images and preserved the background details well. And this GAN model can also be applied to videos since it can process each frame individually to remove bad weather conditions.

Chapter 4

Conclusion

Bad weather removal in images and videos is a challenging problem in computer vision. Physical models of common weather conditions like fog, snow and rain are complicated. And their appearance in images are not linear combination with the background. So it is hard to address this problem.

In this thesis, both traditional and machine learning methods of fog, snow and rain removal in images and videos are discussed. The traditional methods take advantage of the physical model and the photometric properties of these weather conditions. This thesis proposed a fast fog removal method based on scattering model and has better performance compared with other published methods. Methods of snow removal in videos, snow and rain removal in single images are also introduced in this thesis.

However, each traditional method focuses on one type of weather condition. Hence, the Generative Adversarial Network model is introduced in this thesis to solve the problem. It performs the image to image translation and generates clear images from input images polluted by bad weather conditions. GAN model produces great results and makes up for the limitation of traditional methods. Well-trained GAN model can deal with fog, snow and rain images well. This gives it advantage in real world application, as outdoor vision systems does not need to detect current weather conditions in images or videos.

There are also some drawbacks of the GAN model. It needs synthetic images with different weather effect to train the model. And it is hard to get large size of the high quality training dataset. This impairs the performance of the GAN model. It is also time consuming and unstable to train the GAN model. So, in my future work, I will focus on the improvement of the GAN model or other machine learning algorithms to address this problem better.

Bibliography

[1] K. He, J. Sun, and X. Tang. Single image haze removal using dark channel prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 2341–2353, 2011.

[2] X. Tang K. He, J. Sun. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1397–1409, June 2013.

[3] Jean-Philippe Tarel Nicolas Hautiere. Fast visibility restoration from a single color or gray level image. *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1956–1963, 2009.

[4] Kshitiz Garg and Shree K. Nayar. Vision and rain. *Int. J. Comput. Vision*, pages 3–27, October 2007.

[5] Srinivasa G. Narasimhan Shree K. Nayar. Interactive (de)weathering of an image using physical models. *ICCV Workshop on Color and Photometric Methods in Computer Vision*, 2003.

[6] Shree K. Nayar Srinivasa G. Narasimhan. Vision and the atmosphere. *International Journal of Computer Vision*, pages 233–, 2002.

[7] C. Xiao J. Yu and D. Li. Physical-based fast single image fog removal. *International Conference on Signal Processing*, pages 1048–1052, 2010.

[8] Q. Liao J. Yu. Fast single image fog removal using edge-preserving smoothing. *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1245–1248, 2011.

[9] K. B. Gibson and T. Q. Nguyen. Fast single image fog removal using the adaptive wiener filter. *International Conference on Image Processing*, pages 714–718, 2013.

[10] K. Garg and S. K. Nayar. Detection and removal of rain from videos. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, pages I–528–I–535, June 2004.

[11] K. Garg and S. K. Nayar. When does a camera see rain? In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pages 1067–1074, Oct 2005.

[12] Jie Chen and Lap-Pui Chau. A rain pixel recovery algorithm for videos with highly dynamic scenes. *Trans. Img. Proc.*, pages 1097–1104, March 2014.

[13] Jaikar Amol Kalia, Robin. Rain removal from videos using the temporal- spatial statistical properties. In *The 19th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, pages 173–180, 2011.

[14] X. Zhang, H. Li, Y. Qi, W. K. Leow, and T. K. Ng. Rain removal in video by combining temporal and chromatic properties. In *2006 IEEE International Conference on Multimedia and Expo*, pages 461–464, July 2006.

[15] J. Xu, W. Zhao, P. Liu, and X. Tang. Removing rain and snow in a single image using guided filter. In *2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE)*, pages 304–307, May 2012.

[16] Jing Xu Wei Zhao Peng Liu, Xianglong Tang. An improved guidance image based method to remove rain and snow in a single image. *Computer and Information Science*, 2012.

[17] Srinivasa Narasimhan Takeo Kanade, Peter C. Barnum. Analysis of rain and snow in frequency space. *International Journal of Computer Vision*, 2010.

[18] L. W. Kang, C. W. Lin, and Y. H. Fu. Automatic single-image-based rain streaks removal via image decomposition. *IEEE Transactions on Image Processing*, pages 1742–1755, April 2012.

[19] Y. Luo, Y. Xu, and H. Ji. Removing rain from a single image via discriminative sparse coding. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 3397–3405, Dec 2015.

[20] Xinghao Ding John Paisley Xueyang Fu Jiabin Huang Delu Zeng, Yue Huang. Removing rain from single images via a deep detail network. Number 17355172, Honolulu, HI, USA, July 2017. IEEE.

[21] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley. Clearing the skies: A deep network architecture for single-image rain removal. *IEEE Transactions on Image Processing*, pages 2944–2956, June 2017.

[22] Wenhan Yang, Robby T. Tan, Jiashi Feng, Jiaying Liu, Zongming Guo, and

Shuicheng Yan. Joint rain detection and removal via iterative region dependent multi-task learning. *CoRR*, 1609.07769, 2016.

[23] Huadong Ma Xinchen Liu Zheng Zhang, Wu Liu. Going clear from misty rain in dark channel guided network. 2017.

[24] He Zhang and Vishal M. Patel. Density-aware single image de-raining using a multi-stream dense network. *CoRR*, 1802.07412, 2018.

[25] R. T. Tan. Visibility in bad weather from a single image. *Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[26] Xianhui Zheng, Yinghao Liao, Wei Guo, Xueyang Fu, and Xinghao Ding. *Single-Image-Based Rain and Snow Removal Using Multi-guided Filter*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, 1512.03385, 2015.

[28] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, 1611.07004, 2016.

[29] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[30] Ian J. Goodfellow. NIPS 2016 tutorial: Generative adversarial networks. *CoRR*, 1701.00160, 2017.

[31] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, 1703.10593, 2017.

[32] Yijun Li, Sifei Liu, Jimei Yang, and Ming-Hsuan Yang. Generative face completion. *CoRR*, 1704.05838, 2017.

[33] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. *CoRR*, 1609.04802, 2016.

[34] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, 1505.04597, 2015.

[35] Ken Perlin. An image synthesizer. *ACM Siggraph Computer Graphics*, 1985.

[36] S. F. Clifford Ting-i Wang. Use of rainfall-induced optical scintillations to measure path-averaged rain parameters. *Journal of the Optical Society of America*, pages 927–937, 1975.

[37] Todd S. Glickman. *Glossary of Meteorology.* American Meteorological Society, 2nd revised edition edition, 2000.

[38] Catherine Chuang Kenneth V. Beard. A new model for the equilibrium shape of raindrops. *Journal of the Atmospheric sciences*, pages 1509–1524, 1987.

[39] Gilbert D. Kinzer Ross Gunn. The terminal velocity of fall for water droplets in stagnant air. *Journal of Meteorology*, 1949.

[40] T Nakamura C Magono. Aerodynamic studies of falling snowflakes. *Journal of the Meteorological Society of Japan*, pages 139–147, 1965.

[41] W. Mc K. Palmer J. S. Marshall. The distribution of raindrops with size. *Journal of the Atmospheric Sciences*, pages 165–166, 1948.

[42] R. C. Srivastava R. S. Sekhon. Snow size spectra and radar reflectivity. *Journal of the Atmospheric Sciences*, pages 299–307, 1970.

[43] Michael Werman Sonia Starik. Simulation of rain in videos.

[44] Vishwanath; Patel Vishal M. Zhang, He; Sindagi. Image de-raining using a conditional generative adversarial network. *Computer Science - Computer Vision and Pattern Recognition*, 2017.

[45] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, 1511.06434, 2015.

[46] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, 1502.03167, 2015.

[47] Xiao-Jiao Mao, Chunhua Shen, and Yu-Bin Yang. Image denoising using very deep fully convolutional encoder-decoder networks with symmetric skip connections. *CoRR*, 1603.09056, 2016.

[48] Justin Johnson, Alexandre Alahi, and Fei-Fei Li. Perceptual losses for real-time style transfer and super-resolution. *CoRR*, 1603.08155, 2016.

[49] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, pages 600–612, April 2004.

[50] H. R. Sheikh and A. C. Bovik. Image information and visual quality. *IEEE Transactions on Image Processing*, pages 430–444, Feb 2006.