IMAGE FORENSICS BASED ON REVERSE ENGINEERING

by

ZHONGHAI DENG

JINGYUAN ZHANG, COMMITTEE CHAIR
DAVID CORDES
SUSAN V. VRBSKY
YANG XIAO
MING SUN

A DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Computer Science Department
in the Graduate School of
The University of Alabama

TUSCALOOSA, ALABAMA

2014

ABSTRACT

Today with the advent of low-cost imaging devices, such as smart phones, digital cameras and surveillance video systems, digital images become quite common in our everyday life. People tend to believe the scene they have seen, even if the scene is presented in the form of an digital image, as a proverb says, 'Seeing is believing'. However are those images really trustworthy as people have thought? In this multimedia world, with the wide-spread availability of those sophisticated image-editing software, such as PhotoShop and Gimp, it is easy for people to modify images to hide some information or to add a non-existing scene. These manipulations usually leave no visual clues in the tampered image. As a result, the above proverb no longer holds.

To address this problem, 'digital image forensics' was developed. Digital image forensics aims to verify the authentication and integrity of a digital image, without the knowledge of any prior information about the questioned image. It mainly includes two tasks: to determine whether an image is authentic and to identify the source camera of an image.

What distinguishes the original image from the manipulated image is the acquisition process inside the digital camera, which should naturally be the only reliable solution to conquer this problem. In this work, we analyze some key operations along

the image acquisition pipeline, and use the cracked information to perform forensic tasks. The contributions can be grouped into three categories: white balance(WB) , color demosaicking and defocus aberration blurs.

The thesis starts with exposing which white balance algorithm has been applied in the imaging pipeline. The theoretical basis lies on the fact that, given an image, applying the same white balance operation again would not change the image. With the proposed approach, the average accuracy of source camera identification is 99.3% for 5 cameras of different brands, 98.6% for 17 cameras of different models, and 98.5% for 15 cameras equally from 3 models. This is the first time white balance has been used in source camera identification, and it leads to an almost perfect result.

Most commercial cameras have only one *CCD/CMOS* sensor, which produces just a gray scale image. In order to get a colored one, cameras apply a process called demosaicking. This thesis estimates the model and parameters of the demosaicking process to detect forgery. With this method, we can identify which part of the image that is inconsistent with the rest, in the form of their corresponding estimation error. This is the first time that the copy-move area from another image can be exposed using demosaicking.

The third part of this thesis aims at integrity verification using image defocus blur. We can calculate the image defocus aberration, and estimate its depth information. Also from defocus aberration consistency, we can determine whether an image has been altered. This is the first time defocus blur has been used to perform forensic

task. The proposed method increases the average accuracy of splicing detection to 81%, while the best existing published result using the same database is only 68.8%.

ACKNOWLEDGEMENTS

This thesis could not be completed without the support from my families, friends, and advisors My advisor, Jingyuan Zhang, has dedicated enormous energy and countless hours to helping me through every step of my academia study. It is his invaluable advice and encouragement that guides me to overcome obstacles and setbacks. It is his patience and generosity that allow me to explore the forensic area, which leads to my final dissertation topic. His attention to details has helped me hone my ideas and improve the quality of my writing, including this dissertation. I also would like to thank Dr. Xiaojun Qi, who introduces me to this image forensic area.

Thanks go to my wife, Juan Liu. Without her help, life would be much harder for me, and I would never have gone so far, and this dissertation would be impossible. I want to thank my children Linna Deng and Kaile Deng, who are the biggest rewards in my graduate years, for the joys they brought to me. I would also like to thank my parents-in-law, Hengyu Liu and Xiumei Wang, for their kindly help in raising my kids.

From all the faculty in UA, from the classes I took, from their working style that I observed, from their direction, and talking with them, I learned a lot of precious guide in life and research.

Dr. Susan Vrbsky has been the graduate coordinator ever since I came to UA. Whenever I had questions about my graduation study, she is always there to help, with sun-shining smile and quick definite answers. I also appreciate her willingness to serve on my my dissertation committee. I would also gratefully thank Dr. David Cordes, Dr. Yang Xiao and Dr. Min Sun, for their generous support in my graduate study and for their constructive suggestions to my dissertation.

The staff in the Computer Science department have also been invaluable during my time at the University of Alabama. I consider each of them my friend and colleague. The support from Mrs. Jamie Thompson and Kathy DeGraw kept me focused during all these years. Another person I must mention was David Engle, with whom I had worked for two years. It was his work attitude that greatly influenced me.

Last but not the least, to my parents and Grandparents, who raise me, educate me and support me. In loving memory of Grandma Qizhi Wu.

Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

Seeing is Believing.

—Western Proverb

Digital images are everywhere in our lives, especially on the Internet. People tend to believe what they have seen, even by the eye of a digital camera, as it is a 'loyal recorder' of the world. However, with the ease of using image manipulating software, such as Gimp and PhotoShop, images are no longer as reliable as before.

For example, in 2007 a farmer in China claimed that he had captured some images of the 'South China tiger', which had been thought to be extincted in the wild. However, it was later found that the tiger was added from a portrait. The farmer, without any professional training, just made some simple digital manipulations, and effectively hoaxed everyone. Another example can be seen from Figure 1.1, which seems to be taken from the 'real' world, but it is actually composed from 16 images. From those examples, we can see that we are in urgent need of techniques to determine the integrity and authentication of digital images.

FIGURE 1.1. Photomontage: Composite of 16 images

1.1. Forgery of Images

Sometimes people may not treat simple image manipulations as forgery, such as image rescale, contrast enhancement, and image retouching as well as image header alteration. Even the head portrait beautification is actually one type of image forgery.

Strictly speaking, digital image forgery encompasses everything that can be done to a photo, such as resizing, rescanning, sharpening, etc. However, these are also a fake, making the image 'telling a story' rather than 'telling the truth'. For example, Figure 1.1(from Internet) shows you a 'real' world by compositing 16 images into a single one. Further, statistical records show that up to 80% images on the media have been manipulated.

In 2010, the magazine "The Economist" reported the 'Deepwater Horizon oil spill', with the title 'Obama Vs BP', with a cover photo to show Obama's depression. But this tuns out to be created from an image that Obama was just lower his head to see the spilled beach (or just listening, see right of Fig 1.2). And use the simplest copy-paste to hide scenes to express the feeling that Obama is depressed before the beach. Common image forgeries includes the copy-move attack within an image;

FIGURE 1.2. BP Oil Spill. Left: Magazine Cover. Right: Original image

combining two images to create a fake one. As illustrated by Figure 1.1, 16 images are combined together to show a scene that does not exist in the real world.

1.2. Practical Need for Forensics Techniques

Before the advent of this digital media age, forensics techniques are mostly defined as the application of the science to the law. Over the last decade, as the number of crimes involving computer has grown at breakneck-speed, digital forensic techniques evolve to analyze the computer-related crime data at courts, and provide the security-related assistance. Those techniques that include digital images are named as 'image

forensics', besides assisting the criminal investigation, its main task is to help people to determine the integrity and authentication of an questioned image.

People are often challenged with the authentication and of digital images. For example, in the case that two people arguing about the ownership of a digital image, we need to decide whose camera produced the controversial image. This problem was referred as image integrity validation, which involves determining whether the digital image has been modified, and if possible, what kind of manipulations are performed. And the above 'Obama & BP-oil' report on "The Economist" are often referred as image authentication, which means to identify the source imaging device. In the image forensics field, researchers are more concerned about manipulations that will remove some information, or express something that is unreal.

To address above problems, there are two types of methods: active one and passive one. The former is usually called watermarking, while the later is usually named image forensics.

1.3. Active Solution: Watermarking

Digital watermarking is the process to embed information into a digital image, so as to authenticate its owner. The embedded information could be a piece of a text, or an image, and could even be visible like a stamp on an image. Visible watermarking offers the weakest protection and the attacker can easily dodge it. Thus researchers are more interested in invisible watermarking.

All watermarking system would require two main steps: watermark embedding, and watermarked detecting [60]. The owner could claim the copyright of an image by extracting his/her embedded message from it. Some modern cameras, such as Epson PhotoPC 3000Z and Kodak DC-290 have added irremovable watermarks to the output image.

However, it is not practical to require every camera to have this feature. Also, adding the watermarking would lower the image quality. So people prefer passive techniques named 'image forensics', because it requires no additional hardware or cost.

1.4. Passive Solution: Image Forensics

Image forensics aims to verify the integrity/authentication of digital images without requiring any pre-embedded messages. Its theoretical basis mainly include 'image statistics', 'physical property of an object', and 'trace of tampering', etc. Methods in image forensics could be divided into two categories: integrity validation and image authentication.

1.4.1. Integrity Validation. Integrity validation means to verify whether the image is intact. Similar to the manipulation types, common validation techniques includes: near duplicated region detection [6, 39, 57]; splicing detection [13, 48, 64] interpolation and geometric transformation [43, 58, 61]; double JPEG compression detection [47, 70]; lighting inconsistency [18, 41]; chromatic aberration[51, 52], etc. In this work, we will try to validate image image integrity using the demosaicking

and image blur inconsistency. Unlike other methods, our methods are based on the image acquisition process occurred inside cameras.

1.4.2. Image Authentication. Given an image, image authentication means to determine its source camera. An apparent yet simple solution is to use the EXIF (*Exchangeable Image File*) header of an image [53]. EXIF header includes the make and model of the camera, X,Y resolution, flash, orientation, date&time, and sometimes, the quantization matrix used in JPEG compression. As the image header is saved in text format, an attacker could easily replace it

Rather than relying on the easy-to-manipulate meta-data of image header, a good identification method shall make use of the image acquisition process. Some successful methods used 'image statistics'[54], 'lens aberration' [12, 62], 'noise pattern' [21], and demosaicking [59]. These methods received some good experimental results.

However their exploration is far from enough. In this work, we will perform image authentication by estimating white balance, which has never been proposed, and by identification of demosaicking across three color planes instead of individual planes used by existing methods.

1.4.3. Image Forensic Steps. Usually, it is a four-step process for applying digital forensics techniques on a consistent manner:

(1) **Data Collection.** The image data is collected and labeled from all the possible source. The data types varies due to the forensic tasks. For authentication problem, besides the questioned image, we may need the suspected

digital camera, or some images from it. In the case that we do not have the camera, an identical camera with the same make and model might also works.

(2) **Data Forensics.** This process is the key part of the whole task. Use various methods, techniques and tool, we find our interested result. For example, we might find the questioned copy-move part to be more interesting

(3) **Data Analysis and Report** This process need the manual analysis to detect and do to the further and finalized test, to document the experiments and techniques, to derive useful informations that address our problems and give a final conclusion. Unlike other forensic applications which can give definite result, there is chances that our analysis and conclusion might be incorrect. Thus this type of report usually includes the probability that we think our conclusion is true. This is partially due to the fact that variety of images, and the ease of manipulations. Thus there is no definite law that some image is 100% true.

1.5. Contributions

In this work, we propose to solve the image forensics problems using three operations along the imaging pipeline, and conduct experiments to prove their effectiveness. These three studied operations are white balance, color image demosaicking and defocus blurs.

1.5.1. White Balance. For the first time, we propose to use the white balance to do image forensics tasks. We discover the 'idempotence' property of the white balance (WB) algorithm, which means that if applying the same white balance to an image for the second time, the image would not change.

In our experiment, we try to identify the source camera by estimating the white balance algorithm. As we might not be able to identify the exact algorithm used, we use quality metrics (IQM) from the rebalanced image to check whether the applied algorithm is the one used in the camera. The estimated result can then be used to identify the source camera. Our work is the first image forensic job to do source camera identification using white balance. Also, by approximating the WB algorithm, we can determine whether the target image belongs to a suspected camera, while requiring only one training image from that camera.

1.5.2. Demosaicking. Current copy-move detection methods all use the similarity checking to discover the copied part, which is based on an assumption that the copied part comes from the same image. But many times, this assumption does not hold. In the case that copied image part comes from another image, we can use our proposed demosaicking method to expose this manipulation.

The second aspect of our work tries to estimate the color filter array (CFA) interpolation algorithm, also known as demosaicking. In reality, the CFA interpolation is highly non-linear[30], though all existing forensic methods assume that it is nearly linear.

Till now, all existing solutions estimate the demosaicking using a single color plane. However, a sophisticated digital camera interpolates the missing R/B values using their surrounding G values in addition to R/B values. Furthermore, all other researchers assume only one interpolation algorithm across the whole image, but digital cameras have a specific interpolation algorithm for edges to make them sharp.

Our work will estimate the interpolation across the three color planes, and treat the edge area and smooth area separately. After estimation, we can use the estimated coefficients as features to identify which camera produces the given image. Also we can test interpolation consistency within the image to detect forgery patches. To the best of our knowledge, our work is the first that uses the demosaicking to identify the spliced image part.

1.5.3. Defocus Blur. The third part of our work is to make use of the blur caused by lens imperfection to do forensics tasks. Every digital camera has lens to focus the light on the sensor, and the object lies at focus is clearer than that out of focus. The further it is away from the focused point, the blurrier its image would be. Therefore, we can use the blurriness of the image to roughly estimate the object distance and size.

We further use the image blur to check the integrity of the questioned image. By calculating the image blur kernel size along the edges, we extract the depth information. Also, we assume that the depth change should be smooth along the edges, and have limited abrupt change. Thus by checking the consistency along the edges of an object, we can validate if an image has been modified. The only related work requires

human to circle areas at the same distance. Our work will perform the validation fully automatically. In our second experiment, we use the blur kernel size along the edges and test whether an image has been sliced.

1.6. Outline of this Thesis

The rest of this thesis is organized as follows. In Chapter 2, we explain some essential definitions of the digital images, and the image acquisition process inside a commercial camera. In the following three chapters, we first describe the three operations, white balance, demosaicking, and defocus aberration of the image acquisition process. For on each operation, we first explain the corresponding background knowledge, followed by the observation for our approach, as well as the theoretical basis. Then we explain how our methods can be used to do image forensics and conduct experiments to show their effectiveness. This thesis ends with the Chapter 6, where we look back the what we have done, conclude our experimental results and present the future work direction.

CHAPTER 2

Background: Image Acquisition Process

> Know yourself and know your enemy; you'll never be defeated.
>
> —Sun Tzu Quotes

For both forensics tasks: to identify the image integrity and to authenticate the image source, we need to understand how the 'digital camera' produces an image and what kind of properties an image has. Simply pressing the button will trigger a sequence of operations inside a camera, usually named as 'image acquisition process'. As our work is based on three operations in this process: white balance, color filter interpolation and image focusing, we need to understand how a camera applies them on raw signals, and their effects on the output image. In the chapter, we will go over the basic image acquisition pipeline inside a digital camera.

2.1. Image Acquisition Process

Though complicated and varying from brand to brand and model to model, all imaging acquisition processes share some basic operations, Figure 2.1. However, the detailed parameters of these operations are kept secret by camera manufacturers. To

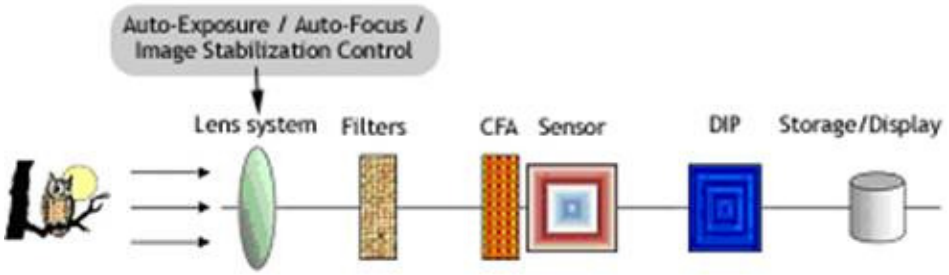


FIGURE 2.1. Image acquisition process in digital cameras

generate an image, we make use of the lights. Light has physical properties including intensity, propagation direction, frequency or wavelength spectrum, and polarisation.

We know that visible light is emitted and absorbed in tiny "packets" called photons, exhibiting properties of both waves and particles. Among them, the property that makes a gray-scaled world is the intensity, which is the only property that a sensor can record.

Light coming from the outside world passes through the camera lens, and a series of filters, including color filter array (CFA), the anti-aliasing filter, and infrared rejection [1], before it reaches the sensor (CCD/CMOS), where it is converted into digital signal.

There are mainly two kinds of sensors for digital cameras, the charge-coupled Device (CCD) array and the complementary metaloxidesemiconductor (CMOS). The

sensor contains millions of photon-sensitive pixels that are arranged in rows and columns, and will convert the light into electrical signal. Electric signal from the sensor is subsequently quantized to digital signal by an A/D converter, and processed by a digital image processor (DIP).

This process introduces a basic problem called digital sampling and quantization. Sampling is directly related to the size of the sensor. In today's market, a common property describing a digital camera is the 'number of pixels', usually is 1 million, which verbosely means the size of the image, meaning the $imageWidth\ X\ imageLenghth$. And this could often be used as the size of the sensor, though actual sensor size might be a little larger on both dimension.

Also, we should be aware of the fact that each sensor has its size, and the sensor at position (x, y) would gather the light around (x, y), *i.e.* the quantized light value is the convoluted result at (x, y). This convolution size is usually referred as 'kernel size'.

The operations in DIP include demosaicking, image correction, gamma correction, defocus, image enhancement, white balance, JPEG compression, etc. In this work, we focus on white balancing, demosaicking, and defocus operations, and use them to perform forensic jobs.

2.2. White Balance

Before explaining white balance, we need to first understand the digital image format for a color image.

2.2.1. Image Color Model. The simplest digital images are white-black, which contains only 0 and 1 in their 2D dimension. If we allow the value to vary from 0 to 255, we have the gray image. Usually we represent this using 8-bit quantization, which is often refers as higher 4-bits, and lower 4-bits.

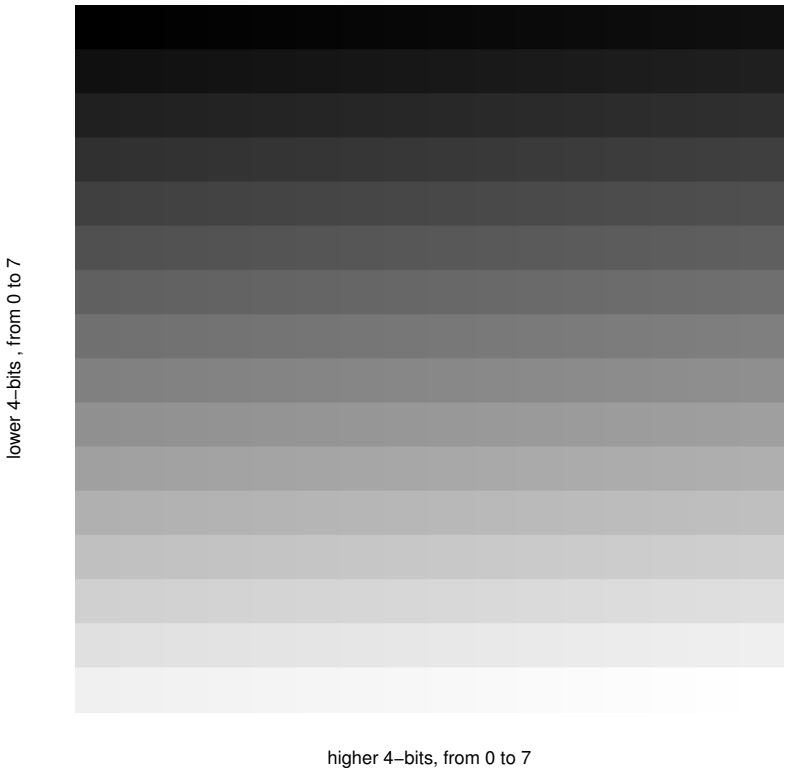


FIGURE 2.2. Gray image scales using higher and lower 4-bit quantization

In order to represent a color image, we can use a mixture of the primary colors: red, green, and blue, Figure 2.2. Thus we further extend the above gray scale model into a color model. Here, the most common way is to associate each pixel position with three values, corresponding to the three lighting frequency spectrum. This model is named the *RGB* color model. Another model uses four secondary

colors: cyan, magenta, and yellow, which are more commonly seen in color printer. Besides colors in a specified primary color, another common color model is HSV, or *blur*, *saturation*, *value* model. The hue corresponds to the color; saturation to how much the color is mixed; and value to the luminance value of this color. In this thesis, we deal with only the RGB color images.

2.2.2. Why WB is needed. We know that our eyes will capture a white paper as white under daylight, tungsten and fluorescent and *etc*. White Balance (WB) is the process that removes the unrealistic colors cast, so that the color will appear the same under difference lighting condition. This process simulates the function of human eyes.

(a) Image afeter White Balance (b) Un-balanced image

FIGURE 2.3. White Balance Illustration

In short, people need WB to get the image appears as accurate as possible, see Figure 2.3. Thus WB will record what people see in their brain, instead of what the image is in reality.

2.2.3. Types of White Balance. Here are some common types of White Balance settings inside cameras.

- **Auto** — the camera will guess the best WB, and apply it.
- **Tungsten** — fit for tungsten (incandescent) lighting conditions, such as bulb lighting, which generally make the colors cooler, see Figure 2.3(b)
- **Fluorescent** — under fluorescent lighting, it image will look cooler, thus this will warm it up.
- **Daylight/Sunny** — simply treat the lighting condition as the 'normal' white colored.
- **Cloudy** — this warms all things than 'daylight' mode.
- **Flash** — use the camera flush as the lighting source, which can be a quite cool light. Thus Flash WB mode will warms up shots a touch.
- **Shade** — the light in shade is generally more bluer than daylight, thus this mode will compensate the blue channel down a little.

2.3. Image Demosaicking

A demosaicking is a digital image process 2.1 used to reconstruct a full color image from the sampled output via the image sensor. The sensor is usually covered with a color filter array (CFA),

The CFA will only allow one color to pass, thus the sensor at each position will output only one color value. But a color image need at least three values, thus the missing value would be interpolated by this demosaicking process.

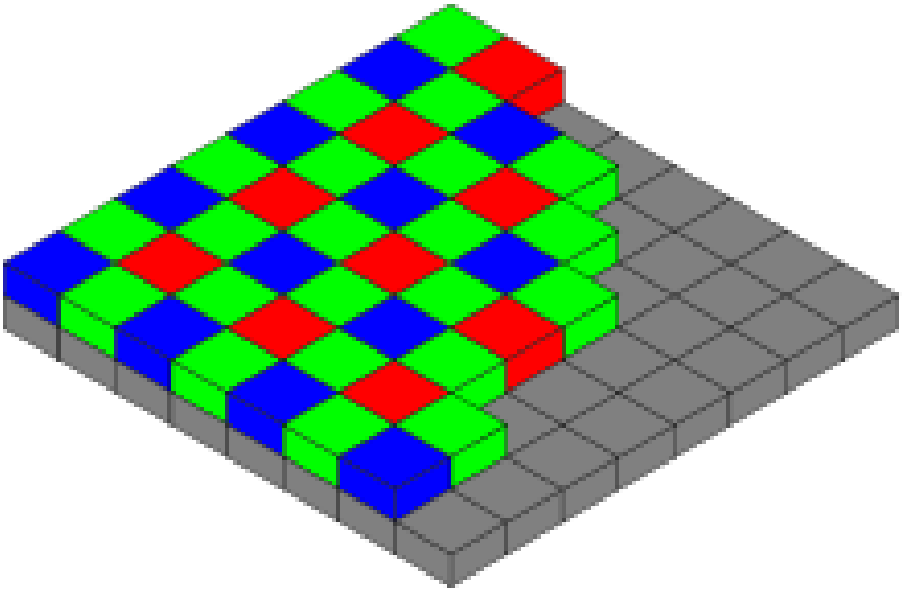
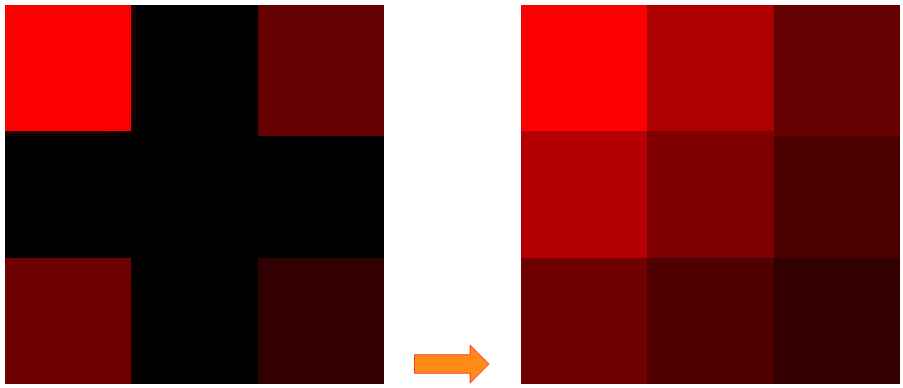


FIGURE 2.4. Bayer CFA pattern on sensor

The simplest algorithm is called nearest-neighbor interpolation, which simply copies an adjacent pixels of the same color channel. This is unstable and unsuitable for a commercial camera. Another simple method is the bilinear interpolation, by which, the missing color would be interpolated from its adjacent two or four neighbors.



(a) Sensor output on Red Channel (c) Interpolated Red channel

FIGURE 2.5. Bilinear Interpolation

2.4. Image Focus System

A camera's focusing system will capture a scene at some distance at the most clearest level, by setting the lens appropriate to the distance of the subject. Usually, people adjust the clear level by the image sharpness.

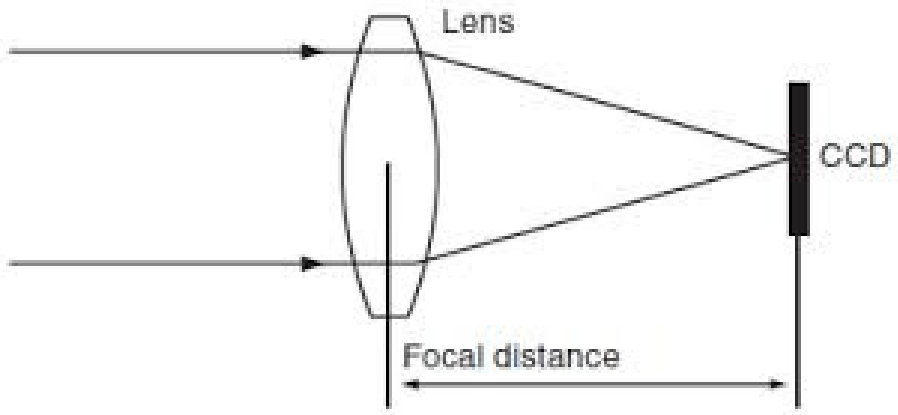


FIGURE 2.6. Camera Focus via Lens

On most cameras, we can focus by setting the focal distance, Figure 2.6, which could be done by adjusting the rings around the lens. This process is done by people's experience and intuition. In today's camera, this focusing process can be done by autofocus. But still, due to the fact that the focusing system is not perfect, thus generate image blur coming out of defocus.

CHAPTER 3

White Re-balance

> Aim for the stars because when you aim for the stars, you will reach the moon. When you aim for the moon, you fall; rest in the clouds. When you aim for the clouds, you clench the tree tops. When you aim for the tree tops, you fall on your butt. Maintain high hopes and you will succeed.
>
> Author Unknown

White balance is the process that makes objects in various lighting conditions look approximately the same, mimicking the function of human eyes. It is reasonable to believe that the white balance (WB) is performed near the end of the DIP part, and our experiments help to solidify this. Although it is feasible to reveal which WB algorithm is performed, it is theoretically impossible to reverse it because of the information loss in WB process. In this work, we first identify the WB algorithm that is performed on a specific image, then use these extracted information to do forensic.

3.1. What is White Balance

In a digital camera, light from a scene that reaches the sensor is transformed into a color space, such as RGB space. This light is the product of the spectral surface reflectance and the spectral power distribution of the illuminant. After the light reaches the sensor, transformation is applied using three *camera sensitivity functions* that each responds to specific part of the light spectrum. White balance is defined as the problem of disentangling the effects of the light source from the resulting RGB-image without changing the actual contents of the image. Generally, this problem is attacked by first estimating the color of the light source $\mathbf{e} = (e_R, e_G, e_B)^T$ (which is assumed to be spectrally constant across the scene), followed by transformation of the RGB-image to impose a canonical illuminant $\mathbf{c}$, usually a white light source. (*i.e.* $\mathbf{c} = (1, 1, 1)^T$)

3.1.1. Illuminant Estimation. All existing WB algorithms are based on one or more assumptions, because estimation of the illuminant is an under constrained problem, *i.e.* the amount of information required to solve the problem is larger than the amount of information available. Most commercial cameras are doing white balancing based on the best-known *Gray-World* assumption. Under this assumption, the average color of an image that is recorded under a white light source is achromatic.

One algorithm based on this assumption simply sets the color of the light source to the average color of the image [8], as any deviation from gray, by assumption, is caused by the illuminant. Alternatively, rather than computing the average of *all*

pixels, using only the center pixels of a segmented image has been shown to improve the accuracy [3, 4].

Another simple yet well-known algorithm is based on the *White-Patch* assumption [45]. This assumption states that the maximum response in any image is caused by a white patch (*i.e.* a perfect reflectance). A simple method, known as *Max-RGB*, under this assumption computes the maximum responses in either of the three channels R, G and B and sets the color of the light source to this value.

In real-world images, both assumptions are likely to fail. Therefore, Finlayson and Trezzi[24] propose to compute a weighted average of the pixel values, assigning higher weights to pixels with higher intensities. The proposed weight-function is based on the Minkowski-norm p, which implies that the Gray-World and the White-Patch are two special cases given $p = 1$ and $p = \infty$, respectively.

Another extension of these low-level statistics-based methods is proposed by van de Weijer et al. [66], and is based on the *Gray-Edge assumption*. This method works with derivatives of images (i.e.edges) rather than with the original pixel-values. Another type of methods is based on the assumption that in real-world scenes, only a limited number of colors can occur. Such methods are called *gamut-based methods* and originate from [25]. The first step is to use a training set of images or patches to learn this limited set of colors, called *canonical gamut*. The second assumption of this method is that the set of colors of an input image (i.e. the input gamut) is representative for the gamut of the light source under which that input image was recorded. This crucial assumption is used to compute a set of possible mappings that

transforms the input gamut completely inside the canonical gamut. Finally, out of the set of possible mappings, one final estimate is selected to transform the input image so that it appears to be taken under a canonical light source. Although the complexity and the explicit requirement of a training phase limit the applicability of this method, it's elegance and performance induce many variations [3, 23].

3.1.2. Chromatic Adaptation. After the color of the light source is estimated, the image can be transformed. This transformation will change the appearance of all pixels, so that the image appears to be recorded under a white light source, e.g. $D65$. This can be achieved by *chromatic adaptation*,[16]. Most adaptation transforms are modeled using a linear scaling of the cone responses, and the simplest form independently scales the three color channels [22, 69]:

$$\begin{pmatrix} R_c \\ G_c \\ B_c \end{pmatrix} = \begin{pmatrix} d_R & 0 & 0 \\ 0 & d_G & 0 \\ 0 & 0 & d_B \end{pmatrix} \begin{pmatrix} R_e \\ G_e \\ B_e \end{pmatrix}, \tag{3.1.1}$$

where $d_i = \frac{e_i}{\sqrt{3 \cdot (e_R^2 + e_G^2 + e_B^2)}}$, $i \in \{R, G, B\}$. A more accurate representation would first sharpen the cone responses before transformation, (e.g. Bradford transform [44] or CMCCAT2000), which is defined as:

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = M_{CMC}^{-1} \begin{pmatrix} d_X & 0 & 0 \\ 0 & d_Y & 0 \\ 0 & 0 & d_Z \end{pmatrix} M_{CMC} \begin{pmatrix} X_e \\ Y_e \\ Z_e \end{pmatrix}, \tag{3.1.2}$$

where d_X, d_Y and d_Z are computed from the tristimulus values of the true white illuminants, by multiplying the corresponding XYZ vectors with M_{CMC}. The matrix M_{CMC} is given by :

$$M_{CMC} = \begin{pmatrix} 0.7982 & 0.3389 & -0.1371 \\ -0.5918 & 1.5512 & 0.0406 \\ 0.0008 & 0.0239 & 0.9753 \end{pmatrix}. \tag{3.1.3}$$

Note that this transform is defined for tristimulus values XYZ, so an RGB-image will have to be converted to XYZ space before applying this transformation, and back to RGB at the end.

3.2. Theoretical Basis

Any digital camera performs a certain type of white balance inside the camera. Even if a user turns off the auto white balance, the camera would still perform some fixed color correction operation. The proposed method is based on the observation that most white balance algorithms will have little or no effect on the image if they are applied to the image for the second time. This observation illustrates the 'idempotence' property of white-balancing method, which is the theoretical base of our work. An operation with 'idempotence' property will produce the same output if executed once or multiple times, *i.e.* given an image im, we have

$$WB(WB(im)) = WB(im). \tag{3.2.1}$$

This promises that, if white-balancing is the last operation inside the camera, and we happen to choose the same WB method as the image has undergone, the output image will not change. For example, for max-RGB algorithm, once we have chose the max-RGB as the white color and make the transformation, we will have the white color in the balanced image. Thus, the second time we apply the max-RGB algorithm, we will find the 'true' white color, thus will make an identical transformation. In other words, in Equation 3.1.1, we have $d_R = d_G = d_B = 1$

Another example is for the most widely used Gray-world assumption, *i.e.* adjusting the image so that the average color is gray will expect the same average color. Thus applying this method the second time will have no effect on the image, since the average of the image is already set to gray. Furthermore, we observe that the methods that are based on the same assumption trend to produce similar results. For example, methods based on the gray-world assumption may receive the same illuminant estimation, thus more likely gives similar results. Finally, we observe that methods based on different assumptions tend to produce bigger color changes.

There are two questions(assumptions) that we must address to solidify our theory. First, is the A=WB performed at the end of imaging pipeline? The answer is no, since at least, the JPEG compression happens after it. But the lucky thing is that, by analyzing the imaging pipeline, we can assert that that some major operations in DIP happen ahead of white-balance, including infrared rejection, gamma correction, demosaicking, lens aberration, anti-aliasing, *etc*. Also, our experiments show that for many images, we can find the AWB algorithm that has little effect when performed

again, for example, average mean square error on three channel < 0.5. Secondly it is reasonable for us to attribute this difference to JPEG compression. Sometimes for a high quality images (compression quality $\leq 98\%$), we can find the exact WB algorithm that will be side effect free. From above, we can reasonably assume that white-balancing is performed near the end of the imaging pipeline, thus the proposed method does not suffer the side effects from other operations applied inside DIP.

3.3. Proposed Methods

Based on the above theory, we can thus perform the white balance again to obtain the rebalanced image. By checking the difference between the original image and re-balanced image, we can determine whether we choose the right WB method. Image forensics based on white balance has never been studied. In this work, we propose some variations, solving the forensics task from different direction, and thereafter divide them into the following stage:

3.3.1. Method 1: Source Camera Identification Using Re-balanced Residue. This method has 4 steps: re-balance image, calculate the image quality metrics from the rebalanced residue, feature selection and classification. First, we need to calculate apply the white balance method on the original image.

3.3.1.1. *WB Methods.* Followings are the WB methods we have used. For each image, we apply various kinds of AWB methods based on different assumptions Since the majority of cameras uses gray-world assumption, we apply more variations of

gray-world methods. For simplicity, we only use methods based on gray-world assumption(including white-patch and gray-edge):

- Gray-World: with six color adaptation methods;
- White-Patch: where the percentage of white is set to $1/255, 0.01$, and max-RGB with smoothing;
- Shades-of-Gray;
- Gray-Edge: with differentiation order 1 and 2.

The six adaptation methods used in combination with the Gray-World method are the diagonal (Eqn. 3.1.1), von Kries, Bradfor, Sharp, CMCCAT2000 and XYZ model [49].

3.3.1.2. *Image Quality Metrics.* In this second step, we calculate the features come from image quality metrics (IQM) from the rebalanced image. IQMs are metrics that are used to evaluate the image quality. And multiple IQM features form a pattern for a single WB algorithm.

The underlying philosophy is that for a re-balanced image, the less change from re-balancing, the better quality it will be. To extract these metrics, by using original image as the baseline, various IQM [2, 15] are extracted from the rebalanced image. Following is the overview of the IQM we have used in our experiments.

- Mean Absolute Error (MAE)
- Mean Square Error (MSE)
- Normalized MSE (NMSE)
- Peak-Signal-to-Noise Ratio (PSNR)

- Czekznowski Correlation
- Angle Mean
- Image Fidelity
- Structural Content
- Correlation Quality
- LP norm (p=1,2, $+\infty$)
- HVS real lab distance
- HVS similarity weight
- Median Block Weighted Spectral Distance (max,mean,median)

3.3.1.3. *Feature Selection.* Feature selection is used to reduce the noise in the features and to eliminate outliers. For simplicity and computational reasons, we use the sequential backward feature selection (SBS) algorithm. This method attempts to optimize certain criterion by removing features, given an initial candidate set of features. In our implementation, the ensemble of features of 17 camera models is used as the initial candidate set, and the average prediction rate in validation set is used as the optimization criterion. Since SBS eliminates four features from the initial set of features, in all experiments we thus use a feature vector of dimension 512.

3.3.1.4. *SVM Classification.* We use support vector machine (SVM) of the RBF kernel to test the effectiveness of our proposed features, with $C = 2^7, \gamma = 2^{-7.5}$. To be specific, we use LibSVM package [10], since we mainly focus on multiple camera identification. A brief description of SVM could be found at appendix.

3.3.2. Method 2: Identify WB Algorithm. In method 1, we do not consider the fact that every camera will perform different WB algorithms under different illuminant conditions. Keeping in mind that there are multiple white-balance methods in a camera. Many times, we even do not how many different WB algorithms have been used in your image database from a specific camera.

Auto white balance would let the camera detect the lighting condition and guess the best algorithm, within a limited range (usually between color temperature of 3000K/4000K and 7000K). To better simulate the balancing algorithm used, we need first cluster the features from the same WB algorithm into one group by clustering.

Although we could find how many WB methods are pre-set inside a camera from the its user manual, we still cannot guarantee that these methods are all included in our experiment dataset. But it is reasonable to assume that the features after the same WB method would follow a Gaussian distribution. Therefore, we can use the G-means clustering method[20] to group images from the same WB method. In this way, we can reasonably identify which type of WB method is performed. And to find the best fitted parameters of the WB method, we employ the greedy search algorithm in the parameter space, as the parameters are usually limited.

3.4. Experiments

It is very useful to investigate the WB method used. One of its successful application is image/video authentication. In the first experiment, we use the image

residue from the re-balanced images, and use their pattern to expose which camera takes the questioned image.

3.4.1. Source Camera Identification baded on Rebalanced Residue. Given an image M, we try to identify which camera device produces it. In this experiment, images come from 'Dresden Image Database' [28], and we use all the camera devices available in the database, up to 29 cameras devices, with 17 models, and 8 brands (Table 3.1). We use only the first 169 images for each camera device, as this is the minimum number of images available per device. The main reason that we use this database is that, almost every camera took the picture of the same scene, and under the same lighting condition. Note all images we use are in JPEG format, and each time, we randomly choose 60% images as training samples and the rest 40% for testing, unless otherwise explicitly mentioned. Further, all classification accuracies listed in this paper are the average of 250 running results.

Brand	**Model**	**Alias**	**Brand**	**Model**	**Alias**
Agfa	DC-504	A1	Nikon	D70	N1
	DC-733s	A2		D70S	N2
	DC-830i	A3		D200	N3
	505-x	A4		S710	$N0_i$
	530s	A5	Rollei	325XS	R1
Canon	Ixus55	C1	Olympus	μ1050SW	O1
	Ixus70	C2	Casio	EX-Z150	$S0_i$
	A640	C3	FujiFilm	J50	F1
Kodak	M1063	$K0_i$	-	-	-

TABLE 3.1. Camera Models (subindex are device ID of the same model)

For camera models having only one device, we omit the device ID. For 'Kodak M1063', 'Nikon CoolPixS710' and 'Casio EX-Z150', we have five camera devices, we

	C2	S0	K0	N3	N1	O1	A1	R1	A4	C3
C2	**99.1**	0.12	0.47	-	-	-	0.15	0.02	0.03	0.06
S0	0.01	**99.2**	0.05	0.14	0.28	0.15	-	0.15	0.07	-
K0	0.04	0.12	**99.0**	0.05	0.51	0.14	-	-	0.10	0.01
N3	-	0.02	0.01	**99.7**	0.09	0.06	-	0.04	0.05	-
N1	0.03	0.42	0.04	0.38	**98.8**	0.30	-	0.04	0.01	-
O1	0.57	0.09	0.01	0.42	0.06	**98.6**	0.18	0.01	0.03	-
A1	0.04	0.02	0.02	0.09	0.01	0.12	**99.3**	0.32	0.05	-
R1	-	-	-	0.04	-	0.02	0.02	**99.2**	0.69	-
A4	0.45	-	0.08	0.38	-	0.01	-	0.28	**98.7**	0.08
C3	-	-	-	-	-	-	-	0.38	-	**99.6**

TABLE 3.2. Confusion Matrix for 10 Camera Identification

use subindex $i = 0, 1, ..., 4$ to identify device 0 to 4, and omitting their device ID indicates device 0.

I. Identification Source Cameras with different Brands. It is very likely that different manufacture use different white balance algorithm, or different parameters if WB algorithm happens to be the same.

For feature based camera model identification, Gloe *et al.* [27] did a comprehensive evaluation on existing best performing methods using the same forensic database.

Experiments show that the average classification accuracy is 99.14%, with the standard deviation of 0.3658, the lowest running as 97.80%, and the highest as 100%. Detailed results are shown in Table 3.2, where the left most column is the true source camera while the top row is the predicted result, and each entry is the prediction percentage.

II Identification Source Cameras with different Models. We also evaluate the performance over all 17 models in 'Dresden Image Database'. The average predicting

Camera	Accuracy	Camera	Accuracy
A1	99.4588	C1	98.8235
A2	98.2824	C2	98.9176
A3	99.0824	C3	99.4118
A4	99.7882	R1	99.4118
A5	98.4706	N1	99.5765
S1	99.3176	N2	98.9882
F1	100.000	N3	100.000
K0	98.7294	N0	99.3647
O1	99.3882	-	-

TABLE 3.3. Prediction percentage for all available 17 cameras of different models

accuracy is 99.27%, with the lowest running as 96.89%, and the highest as 100% (Table 3.3).

Comparing this with Table 3.2, we can see that while the camera number increases, the performance does not degrade.

From these experiments, we can observe that our proposed method can distinguish among camera models as well as camera makes. Moreover, this method scales well for an increasing number of different cameras.

These results can be explained as follows. First, as the feature vectors are derived from the image quality metrics, they are inherently consistent with each other. This is reflected in the result of the SBS feature selection method, which eliminates only four features. Second, all automatic white balancing methods are based on the gray-world assumption, which is considered to be the most often used method inside digital cameras. Finally, it is likely that white balancing is the at the end of the digital image processing (DIP) pipeline, thus our method does not suffer the side effects from other processes applied inside the DIP.

	$S0_0$	$S0_1$	$S0_2$	$S0_3$	$S0_4$
$S0_0$	**98.56**	0.58	0.00	0.35	0.51
$S0_1$	0.16	**98.50**	0.44	0.54	0.36
$S0_2$	0.18	0.32	**98.94**	0.38	0.18
$S0_3$	0.11	0.22	0.44	**99.06**	0.16
$S0_4$	0.51	0.32	1.13	0.52	**97.52**

TABLE 3.4. Confusion Matrix for 5 'Casio EX-Z150' devices

III. Identification over Cameras of the Same Model. Currently, the most challenging problem of camera identification is to distinguish among camera devices of the same model. By randomly choosing 60% images for training, we have an average accuracy of 98.52%, with the lowest running accuracy being 96.47%, and the highest being 100%. Similarly identification over 5 camera devices from 'Kodak M1063' gives an average accuracy of 98.57% and 'Nikon CoolPixS710' of 98.57%.

These results (Table 3.4) clearly depict that even for devices of the same model, our method can correctly identify the source camera of a given image.

To explain this, we did more investigations. First, for 'Nikon CoolPixS710' , if we open images 'CoolPixS710_1_13228.JPG' and 'CoolPixS710_2_13645.JPG', ('_1_' and '_2_' are device ID.), we could find that the image from device 1 is brighter than that from device 2, even their content are the same. This observation holds for all images having the same scene from these two devices, even though they are taken at the same time and under the same lighting condition.

Next, by checking their 'MakerNotes', we find that the selected auto-focus position varies between devices, even they are of the same model (Table 3.5). We believe that the trivial differences like this, cause the differences in scene illumination estimation process, and finally affect the output images. To better understand how these

File Name	Exposure Time	AFPoint	AFPoints InFocus	Relative AF-Position
0_12830	1/400 sec	3	8	Mid-left
1_13228	1/400 sec	4	16	Mid-right
2_13645	1/400 sec	0	1	center
3_14082	1/320 sec	8	256	Lower-right
4_14500	1/320 sec	3	8	Mid-right

TABLE 3.5. Device difference among 'Nikon CoolPixS710'

differences affect the feandatures extracted, we reduce the dimension to 2 by principle component analysis (PCA) (Figure 3.1).

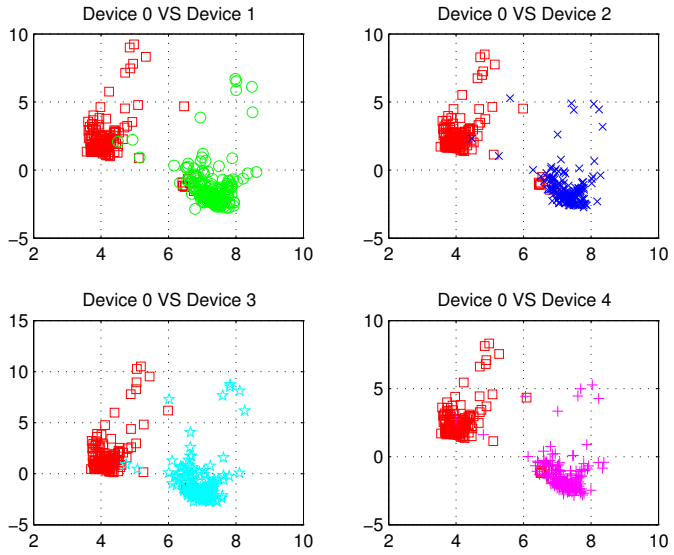


FIGURE 3.1. Comparison of device 0 to other 4 devices of 'Casio EX-Z150'

To be more general, we also did the test over all 29 camera devices listed in Table 3.1, with 60% images for training, experiment gives an average accuracy of 98.10%, with the lowest running as 96.55%, and the highest as 99.04%. Due to the use of a huge number of camera devices including 15 devices out of 3 camera models, the performance degrades a little, but the difference is acceptable. Compared with

all other existing methods, our proposed methods could have a surprisingly high identification accuracy. Though its computational complexity at training stage is a bit of high, image authentication does not require real time at all, so it can not be a drawback for our algorithm.

3.5. Conclusion

In this work, we propose a novel method to identify the source camera by using the WB residue pattern. Experimental results on a large data set show the proposed method is very effective. Moreover, the prediction accuracy almost does not degrade as the number of different cameras increases, demonstrating the scalability of the proposed method. Further, we study the image

And we also show that even for different devices of the same model and brand, the proposed method is still able to distinguish among them.

Although we only do the source camera identification, the same idea could be applied to various applications, including but not limited to copy-move detection and steganalysis, as well as reverse engineering.

CHAPTER 4

Demosaicking Exposure

Men can be judged by people around them.

Western Proverb

Most cameras use only one sensor to record the lighting. To get the color image, we need the demosaicking operation. Different cameras use different demosaicking algorithms. However, we still can estimate the demosaicking algorithm, by calculating how much each pixel could be represented by its surrounding elements. As this is the core operation inside a camera, we thus can use it for multiple purposes, *e.g.* identifying source camera, copy-move detection *etc*. In this chapter, we only show the experiments of copy-move forgery detection.

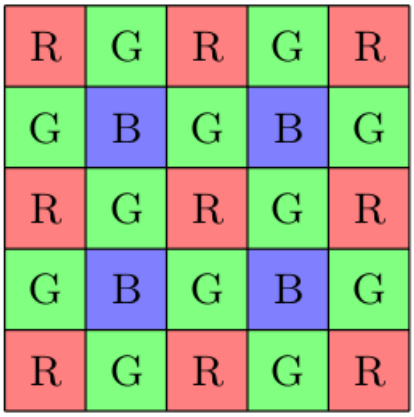


FIGURE 4.1. Bayer Color Filter array

4.1. Background

4.1.1. Color Filter Array. In this part, we first explain the structure of color filter array (CFA), and then discuss some popular interpolation algorithms. Typical photon sensors detect light intensity with little or no wavelength specificity. Therefore, a sensor would only produce a 2D gray scale image, but a color image needs at least 3 bands.

The key part of any digital camera is the sensor (CCD/CMOS), which transform light signals into electric ones. As a CCD pixel is intrinsically monochromatic, in order to get a color image, we could use 3 CCD sensors and a dichroic beam splitter prism, that splits the image into red, green and blue components. Each of the three CCDs is arranged to respond to a particular color[7]. But sensor is the most expensive part in the camera, therefore this method is only employed on some advanced cameras. Because these sensors are sensitive to only the light energy, *i.e.*, they are blind to the color of lights, thus to generate a colorful image, most cameras employ a color

filter array (*CFA*) overlaid on the sensor. On each sensor unit, CFA will allow only one color band to pass through, therefore, each sensor unit record only one color value. To build the full color image, the missing values are interpolated from the neighborhood available sensor readings. This interpolation process is often referred to as demosaicking, which is still an open problem, and is highly non-linear [30].

To reduce the cost of getting a color image, most manufacturers use one single sensor with the help of color filter array (CFA), which is always placed in front of the sensor to select the band of wavelengths arriving at the CCD/CMOS array [55]. Digital color cameras generally use a Bayer mask over the CCD sensor, see Figure 2.4. Each four-pixel square (Bayer filter mosaic) is arranged in a square grid of photo-sensors, arranged in 50% green, 25% red and 25% blue square block, and two green values must lies diagonally. Because human eyes are more sensitive to light of medium wavelength, the green takes half of the sampling rate [7]. The missing values are interpolated from its neighborhoods, for this specific case, traditional linear interpolation algorithm does not work well.

There are also other CFA filters, include but not limited to the Cyan-Yellow-Green-Magenta (CYGM) pattern, the Red-Green-Blue-Emerald (RGBE) pattern, and the Cyan-Magenta-Yellow (CMY) pattern [46]. For example, CMYG filter would allow each of the four colors sampled equally within a 2-by-2 square grids. However, the most common and widely used one is still the Bayer CFA, and in our database, digital cameras are all use Bayer CFA pattern.

The most common CFA pattern is the Bayer color filter array, Fig 4.1. Nowadays most cameras employ the RGB color space, meaning that every position on an image has 3 color values: Red, Green, Blue.

With the simple interpolation algorithm, each color channel is interpolated independently, using neighboring values from the same channel. Simple examples are nearest neighbor, bilinear and bicubic interpolation. Some sophisticated algorithms would perform the interpolation across color channels. For example, green channel are interpolated alone, but red and blue channels are partially interpolated from green neighborhood. Advanced algorithms would take all 3 neighboring values in consideration, and are adaptive to the local or global characteristic of the image.

4.1.2. Demosaicking Algorithms. From above, we know that each sensor unit records only one color value, and the missing values are interpolated from the neighboring sensor readings to build the full-color image. This CFA interpolation are also named demosaicking, used to change from sensor color-space to display color-space, which can be either sensor GRGB to display RGB, or sensor CMYG to display RGB.

Note that the color demosaicking algorithm is still an active research field [29, 33, 35, 65, 71], and is highly non-linear [30]. According to Gunturk [30], practically, there are two groups of demosaicking methods used in digital cameras. The first one is based on heuristic approaches, and the second one formulates demosaicking as a restoration process.

Heuristic Approaches. This group of methods try to do a filtering operation and filling the missing color values, generally based on some reasonable assumptions. They

could be spatially adaptive, and might exploit correlations among the color channels. The following are a list of popular assumptions.

- EDGE-DIRECTED : Traditional interpolations mainly fails at textured regions. Thus, this method would adaptively calculate the local edge information to decide using which part of neighbor pixels to use.
- CONSTANT-HUE-BASED : Based on the assumption that the hue (color ratios/differences) within an object in an image is constant.This method is extensively used for the interpolations of the R and B channels.
- WEIGHTED AVERAGE is an improved version of edge-directed method, but instead assign the weight for each near pixels, mainly by its edge direction.
- SECOND-ORDER GRADIENTS AS CORRECTION TERMS: This is another improved version of edge-directed method, but only on green plane. According to the second gradients for R or B color plane, and make some correction on Green plane.
- HOMOGENEITY-DIRECTED: instead of choose interpolation direction based on edge information, we can choose different measures, *i.e.* local homogeneity, to choose between horizontal or vertical interpolation.
- PATTERN MATCHING is reasonable to be believe that some patterns in more common in the data, so we can try to find the pattern and fit it to one of several templates.

- VECTOR-BASED: Considering each pixel as a vector in (R,G,B) space, and the interpolation is designed to minimize the angle or the distance among the neighboring vectors.

Reconstruction Approaches. This group of methods assume the inter-channel correlation of the prior image and solve an optimization problem to reconstruct the missing color values iteratively by optimizing the correlations.

- REGULARIZATION uses spatial smoothness and color correlations terms in the objective function, and iteratively do the minimization.
- PROJECTIONS ONTO CONVEX SET APPROACH iteratively forces similar high-frequency components among the color channels to ensure the data consistency.
- BAYESIAN APPROACH formulate the demosaicking problem as a posteriori probability(MAP) process, where spatial smoothness and constant hue assumption are used as regularization terms.
- ARTIFICIAL NEURAL NETWORK APPROACH, the ANNs approach use training images to learn the parameters to be used in the reconstruction part.

Another interpolation group creates its model of image acquisition process, and formulates the demosaicking problem as a reverse problem. Due to its computational complexity, this type of methods is unlikely to be employed in real life cameras, but it is still academically valuable. Some digital cameras, especially the single-lens-reflex (SLR) cameras, *e.g.* Nikon D70s, provide raw image data captured by the sensor, thus the interpolation could be done later.

From all above, we can see that in a digital camera, the interpolation process is highly nonlinear. In the image acquisition pipeline, this interpolation result is the base of many operations, *e.g.* noise reducing, white balance, deblurring, JPEG compression, *etc.*

4.2. Splicing Detection

Image forensics is still in its infancy. Although splicing detection is a key problem, not a lot of work has been done. Hsu and Chang [38] check the consistency of camera characteristics among different areas in an image, but they only received 70% precision. Chen *et al.* [11] extract features from the sharp transitions introduced by the spliced image part. Their features mainly comes from the moments of wavelet characteristic functions and 2D phase congruence, but their experiments are only about 80% precision. Similarly, Shi *et al.* [64] extract features from moments of characteristic function of wavelet sub-band and Markov transition probabilities.

Most of existing methods use grayscale images or luminance components of color images, thus inevitably lose some information comparing to the color image. Further, based on benchmark database: 'Columbia Image Splicing Detection Evaluation Dataset' [36], blind detection precision [11, 56, 64] are not satisfying. Though Shi *et al.* [64] claims the 92% correctness, they use too many features on their testing image set, and what was worse, they cannot figure out the altered area if an image is spliced.

In this work, we propose a new blind and effective splicing detection approach based on the image demosaicking inconsistency. Based on the fact that at each image pixel, camera hardware can only generate one color value, and the other two are interpolated from their neighborhoods, we can imagine that different cameras would employ different interpolations algorithms, thus spliced parts could be recognized from the original part. Even if two cameras employ the same interpolation algorithm by coincidence, their generated images would surfer different post-processing, which would disturb the interpolations at two different degrees.

In section 4.5, we present our model find the interpolation relationship among pixels. And then, for each pixel, we calculate an error rate, which would be used to help determine whether this location is the spliced from other images. Section 4.7 presents two experiments in detail, identifying spliced image from untouched ones, and exposing the spliced area within an single image. Finally, conclusion and discussion are provided in Section 6.

4.3. Related Work

Due to the fact that demosaicking should be done at the beginning of the DIP process, their results are highly distorted by following operations, such as noise removal, color correction, JPEG compression *etc*. Thus we are, theoretically, not able to expose the exact interpolation method.

Researchers still try to make use of the interpolation in more smart ways. First, as CFA pattern varies, Kirchner [42] proposed an efficient estimation of CFA pattern

configuration, by assuming the linear demosaicking algorithm. But generally, this assumption is not valid, and this configuration has been included in the image header information. Also, his experiments are done on synthesized images rather than the read digital images.

Bayram *et al.* [5] did the source camera identification based on the traces of the proprietary interpolations algorithm. They define a set of image characteristics and train a set multi-class SVM classifier. However, they use only two cameras in the experiments, and get a quite low accuracy. Bayram *et al.* [63] later improved their result by incorporating methods to detect the interpolation artifacts in smooth images parts. But still, their result is a trumpery. Long and Huang [50] also did the source camera identification using the demosaicking, by exploring the spatially periodic inter-pixel correlation, and formulate it in the quadratic form.

None of the above really exposes the interpolation process, or reveals its parameters. In 2005, Popescu and Farid [59], use the Expectation-Maximization (EM) method to quantify the specific correlation introduced by CFA interpolation. In their model, they also assume the linear model for the periodic correlation introduced by CFA interpolation. Though they also assume the linearity and find the interpolation on a single plane.

4.4. Proposed Methods

From above, we can see that, to make use of demosaicking, only some preliminary work are done. Although it is theoretically impossible to reveal the exact method because of the information loss in the acquisition pipeline, we propose our own methods to approximate the real demosaicking process in the that the image has undergone.

Dependency among 3 Color Planes. Since all the published works are done on a single color plane, but in many demosaicking methods, the missing R/B values are partially interpolated from neighboring G values.

For a color image, we use I to stands for one color channel. As borrowed from [59], we also assume that $I(x, y)$ belongs to one of two models: (1) M_1 if the sample is linearly correlated to its neighbors, satisfying:

$$I(x, y) = (\alpha * M(x, y))^{\gamma} \tag{4.4.1}$$

Or (2) M_2 if the sample is not correlated to its neighbors, i.e., is extracted directly from sensor matrix M.

Here, in our experiments, we directly use the GRGB pattern on the tested image, further research should first determine their CFA pattern and its layout. As their pattern could be found in camera's user manual.

The EM is a two-step iterative process to estimate the model parameter: E-step: (Expectation-step) the probability of each sample belonging to each model; and M-step: the update the model parameters, and the probability that $f(x, y)$ belongs to

model M_1 is estimated by Bayes' rule:

$$Pr\{I(x,y) \in M_1 | I(x,y)\} = \frac{Pr\{I(x,y)|I(x,y) \in M_1\} Pr\{I(x,y) \in M_1\}}{\sum_{i=1}^{2} Pr\{I(x,y)|I(x,y) \in M_i\} Pr\{I(x,y) \in M_i\}} \tag{4.4.2}$$

Where the prior probability of $Pr\{G(x,y) \in M_2$ is set to be $1/2$, and both $Pr\{B(x,y) \in M_2$ and $Pr\{R(x,y) \in M_2$ are $1/4$.

We also assume that

$$Pr\{y_i | y_i \in M_1\} = \frac{1}{\sigma\sqrt{2\pi}} exp\left[\frac{-\left(y_i - \sum_{k=-N}^{N} \alpha_k y_{i+k}\right)^2}{2\sigma^2}\right] \tag{4.4.3}$$

Note that σ , the Gaussian distribution in Equation 4.4.3 is estimated in M-step.

In the M-step, the coefficient α is approximated according to the minimum least-square rule

$$E(\alpha) = \sum_{x,y} w(x,y) \left(I(x,y) - \alpha * I(x,y)\right)^2 \tag{4.4.4}$$

Further, to have a sharp edge, cameras would treat the edge separately, and apply a specific interpolation along the edge. Due to this, we also incorporate to treat edge from smooth regions, and trying to employ their demosaicking algorithm.

4.5. Experiments Approach

The underlying philosophy of the proposed method is quite simple: in an untouched image, the whole image undergoes the same demosaicking process inside the

camera. Thus if we estimate the demosaicking algorithm, the error rate should be approximately the same. For a manipulated image, the spliced part comes from different images, and undergoes different demosaikcing algorithms. Further, the spliced part is often rotated, re-sized to fit the target location, which would makes the interpolation relationship different from original one. Thus in the original image, by checking the error rate on each pixel, we can find the spliced parts.

4.5.1. Interpolation Estimation Model. For simplicity, we assume all the CFA patterns are in the '**RGGB**' form, column first. We use the quadratic form to approximate the demosaicking algorithm.

4.5.1.1. *Region Selection.* To avoid dealing with the border pixels, we ignore 6 pixels on on the left and right sides, the same for upper and lower border. Also, it is reasonable for us to assume that on smooth areas, horizontal neighbors and vertical pixels should have the same weight during interpolation. Thus, when considering interpolation area, we ignore the bordering areas with thickness of 5 pixels. For computational efficiency, we simplify the kernel interpolation filter in the shape of diamond with the vertical and horizontal lines as its diagonals. We know that most cameras would employ a edge sharping operation to make the edge sharper, and most sophisticated cameras would employ different demosaicking algorithm for smooth region and edges. Thus, to get a better approximation result, we only deal with the smooth region to find the interpolation model, though all the pixels on edges can only be used as the interpolation candidates.

4.5.2. Estimation Model. Although interpolations process inside camera is nonlinear, we still assume its linearity to simplify our process, but enhanced it to the quadratic model. We also assume that in all cameras, missing Green values are interpolated from a diamond form with radius of 5. Looking further into the diamond shape of the green channel, for any position whose green value is missing, its surrounding diamond area have 36 Green candidates, Fig 4.1. Taking into the symmetry into consideration, there are only 12 independent pixel candidates with flexible coefficients. Thus without loss of generality, we can treat pixels on the upper right part of the diamond as independent, Fig 4.2.

Also, as we consider the quadratic form of the interpolation model, we need consider the difference between two symmetric points, say a, b about the questioned location. In our model, we actually add the expression $(a - b) \Rightarrow ab$ into candidates. For simplicity, we consider only the green channel, and ignore the Red and Blue channel. From above analysis, for each candidate pixel p, assuming its independent points $x_i \;\; i = 1, ..., 12$ on the upper right side, then the interpolation function in our model is :

$$F(p) = \sum_{i=1}^{12} \left(a_i * \sum S^2(x_i) + b_i * \sum S(x_i) \right) + \sum_{i=1}^{12} k_i * S0(k_i) + c \tag{4.5.1}$$

$$\tag{4.5.2}$$

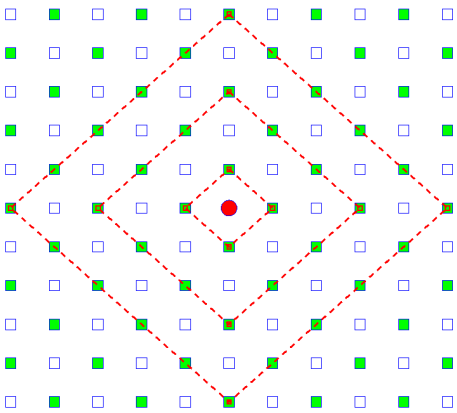


FIGURE 4.2. Interpolation Candidates Model

Where $S(x_i)$ is a function that returns pixels $x_i^{'}$ that are symmetric to x_i about origin or two diagonals, inclusive; $S0(x_i)$ returns pixels $x_i^{'}$ that are symmetric to x_i only about origin, none inclusive; and c is a constant.

For the pixels whose values are generated directly from sensor, their error should be zero, making them as black holes. So to facilitate our spliced region exposure, we use bicubic interpolation method to fill out these holes. Thus we can see that there are actually only 37 independent coefficients to be estimated from Equation 4.5.1.

4.5.3. Error Ratio Map. Given an image im, after above interpolation estimation, we got a map that shows the error our estimation for each pixel.

$$Err(im) = ||SG(im) - F(SGim))|| \tag{4.5.3}$$

And by considering the fact that with the same error at position (i, j), for example $Err(i, j) = 3.4$, for pixel with original value of 255 and original value of 0, means

significant difference. We also take the estimation error rate into consideration.

$$ErrRatioMap(im) = ||SG(im) - F(SGim))||/SG(im) \tag{4.5.4}$$

Where $SG(:)$ is the function that extracts the smooth area on green channel, solely determined by the original image. Note that in the above process, the interpolation coefficients depends on the reverse image results, To describe the image error ratio map,we use the some common statistical metrics: maximum, mean, coefficients of variance, spread, skewness, kurtosis and Chi-square.

To avoid the divided by zeros case in Eqn. 4.5.4 and to cease turbulence, we simply exclude those points whose original value is only 0 or 1. Also, due to the fact that different images have different size, thus simply to normalize the error ratio map via

$$normalizedMap(im) = ErrRatioMap(im) * c/length(im) \tag{4.5.5}$$

Where c is a constant, assigned as 10000 in this experiment, and $lenght(im)$ is a function that returns the number of pixels in image im. Also, sequential backward feature selection is used to find the optimal feature set.

4.5.4. Classifier. In this experiment, we employ the LIBSVM [9] and use RBF kernel. Grid searching is used to find the best parameters from the classifier.

4.6. Experiments

Find the interpolation inside camera is very useful in image forensics field. One basic application is that it can be used to identify the source imaging device.

In the first experiment, we use it to find out which the source camera. Another major image manipulation is image copy-move attach, which is widely used to hide scenes unwanted. We will explore this application in the second experiment.

4.6.1. Source Camera Identification. In this experiment, given an image, we try to identify the source camera given an image, providing that we have a series of images from the target camera. As Bayram*et al.* [5] indicated, find the demosaicking trace in frequency domain could help, thus we make use of the sparking point, as they are the frequency that appear many times across the image.

For these cameras of 5 different brands , we can achieve more than 95% correctness. But for cameras of the same brand and model, it will do the identification merely on luck.

Since, we can reasonably expect that the images come from cameras of the same model/brand will exhibit the quite close relationship. But, due to it uniqueness, as each camera will only employ one type of interpolation method, they gives solid support in determining two image comes from different cameras.

4.6.2. Across-image Copy-Move Detection. In the filed of image forensics, detecting copy-move attack is the most essential technique. But it seems that available techniques focus on the copy-move attack that is performed within the same image.

Thus most copy-move detections are based on the similarity of the suspected regions, but if the suspected region comes from another image, *i.e.* cross-image copy-move, current methods can not help. We develop a new blind forensics method to detect the copy-paste attack from some unknown image into current one.

As to the cross-image copy-move attack is not mentioned. Till today, the only cross-image forgery detection is to use the lighting inconsistence [18, 41]. And nobody has utilized the imaging acquisition process to do the copy-move detection. This paper tries to use the CFA interpolation inconsistence to find the copy-move attack. Note that previous researchers [50, 59] over simplified assume that the interpolation is performs inter-channel, which is also not true.

The theoretical base of this paper is that if a copy-move forgery is performed, no matter it is inner-image or cross-image, the CFA model may not be the same. Even both cameras for the source and destination image share the same CFA model, for example, in the case that copy-move copy-move attack is performed inner-image, the period of the CFA mask is very likely to be inconsistent.

In this experiment, we first make use of the CFA filter, by reversing its demosaicking process to get the identification of a camera, and then using Hidden Markov Model to estimate the probability that some region adapting another CFA model/period. The theoretical basis is that different images use different *CFA* interpolation algorithms, and sometimes, using a different *CFA* filter. The novelty of this paper is that it first estimate the interpolation coefficient across color channels, and we also

first use HMM model to find the interpolation inconsistence between image part to find the copy-move attack.

Results and Analysis. Current experiments using EM algorithm is trapped in a dog house, and the result is error prone.

4.7. Experiments

In our experiments, we apply the proposed method to solve two problems in image forensic : spliced image identification and exposing spliced area. To better justify the proposed method, we use the benchmark forensic database in experiments 4.7.1 and 4.7.3. As we focus on authentic/spliced image data, the only available public database comes from Columbia University [37], containing 183 authentic images from 4 cameras, and 180 spliced images. Note that different from common copy-move manipulation, whose forged parts come from the same target image; in image slicing forgery, forged part comes from a different source image.

4.7.1. Separate Spliced Image from Untouched Ones. In this experiment, we try to separate the spliced image from those untouched images. Its underlying theoretical basis comes from the assumption that, if an authentic image use the same demosaicking method, then the error ratio map should be smooth. On the other hand, as the source image tends to employ a different demosaicking method, even it happens to have undergone the same interpolation method, its post processing operations should be different. Therefore adding the spliced part would disturb the estimation error map. Further, these spliced image parts are very likely to have been

	Authentic	Spliced
Authentic	88.38%	11.63%
Spliced	9.20%	90.80%

TABLE 4.1. Confusion Matrix to identify the spliced image

rotated and re-sized, *etc*. Thus, it is reasonable to assume that the image error ratio map should have more turbulence than the authentic one. For simplicity, we merely use the histogram based features.

4.7.2. Identification Performance. We run 10 times to get the average performance of the proposed method. The average performance could be seen at

Our overall classification rate is 89.59%, which outperforms their highest accuracy 87.55% in [37]. Due to the fact that the number of sliced image are a bit more than that of authentic image, the true positive (TP) rate slightly higher than the true negative (TN) rate.

4.7.3. Expose Spliced Area. Our method goes further to label the forged region for a sliced image. Since the spliced part have undergone different demosaicking algorithm from the original image, thus if we exclude the spliced parts and approximate the interpolation via the original part, we will received much better approximation result, *i.e*., the interpolation residue would shrink significantly.

4.7.3.1. *Region Labeling.* Due to speed consideration, instead of considering all pixels in experiment 4.7.1, in this experiment we divide the image into 32-by-32 non-overlapping blocks. On each iteration, we label one block as the 'suspected' one, while its 8 neighborhoods are labeled as 'ambiguous', and ignored them when doing the regression.

For block (i, j), we calculate the interpolation coefficients using the rest pixels, say $imRest(:,:)$, and get a new estimation error map. Comparing the same part between the new error map with the one over the whole pixels, we could use their difference $Dif(i, j)$ to quantify the possibility the 'suspected' part could be a spliced part.

$$Dif(i, j) = |average(Err(imRest) - newErr(imRest))| \tag{4.7.1}$$

Where $Err()$ is the error map over the whole image, and $newErr()$ is the one that works only on the rest part of the image $imRest$, excluding the 'spliced' block and its neighborhoods. After the iteration, we use the block with largest difference, and check the $DifRatio$ to determine whether this part is large enough to make it a 'spliced' part.

$$DifRatio(i, j) = Dif(i, j)/mean(Err(im)) \tag{4.7.2}$$

4.7.3.2. *Spliced Region Growing.* Once we determined that the 'suspected' block is the 'spliced' part, we need expose the whole spliced region. Starting from the seeding block, we test the 4-way direct neighbor blocks, and determine which class (origninal/spliced) they belongs to. For simplicity, we still use the average of error map to determine the

$$classRatio(:,:) = mean(Err_{orig}(im))/mean(Err_{slic}(im)) \tag{4.7.3}$$

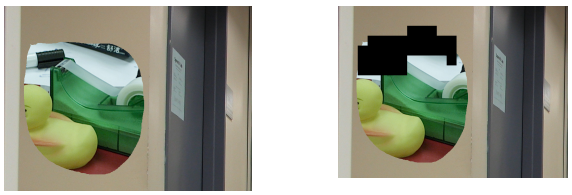
Where the $Err_{ori}(im)$ means the error map if treated as the original image, and similarly for $Err_{slic}(im)$. Note that after each iteration, so long as the spliced region

has changed, we need recalculate the interpolation coefficients for both original and spliced image parts.

$$f(block_k) = \begin{cases} Original, & \text{if classRatio} > 1.5 \\ Spliced, & \text{if classRatio} < 0.667 \\ More\ discussion, & otherwise \end{cases} \tag{4.7.4}$$

In the case the $classRatio < 0.667$, we would divide the current questioned block into two equal sized parts, and check the belongings of each part. For example, in the case that the left part of the block is adjacent to spliced area, then the the block is divided into left and right part, both with the size 32-by-16. Similarly with the upper and lower adjacent case. If the questioned pars is adjacent to 'spliced' area on both left and upper side, then we choose the case that is discovered first.

By repeating the above step, we thus growing the spliced image into roughly the true sliced region. One example could be seen in figure 4.7.3.2.



(a) Spliced Image (b) Seeding Blocks for Growing

FIGURE 4.3. Staring growing block illustration

4.8. Discussion

Till now, our source camera identification using WB exposure does not give out an satisfying result.

Partially, because our WB result are too good.

And coding for copy-move detection always gives errors, deadly loop *etc*.

Future Work. To decrypt the correlation, EM method is not the only choice, in our experiment we also employ the independent component analysis, (ICA), method description could be found at appendix.

However, using ICA, as we are dealing with images, but ICA are only use for 1-D variable. If we simply transform each 2D image into the vector form, the size of transition matrix would be too huge. Thus we temporary solution is to use the $64 * 64$ square area to identify the interpolation algorithm.

Further, we can incorporating other Factors to help improve our result. Still using the EM algorithm, although Popescu did it by assuming prior probability distributions. But which could be not necessary to be true. Also, we need consider others factors, such as white balance, and gamma correction. As to the JPEG and noise removal process, we can reasonably treat their effects are zero-mean noise, thus do not need to consider their effects.

For simplicity, in this stage, our experiment only taking consideration of the white balance effects. For each color value $k,\ k = 0, 1, 2, ..., 255$, we assume the linear transformation done by WB algorithm. Thus, we have the new value $a_k * k + b,\ a > 0$.

Combining the new color values inside by the previous algorithms, we can get a better result

CHAPTER 5

Blur and Defocus Estimation

> Whatever people did, they left a trace.
>
> Unknown Author

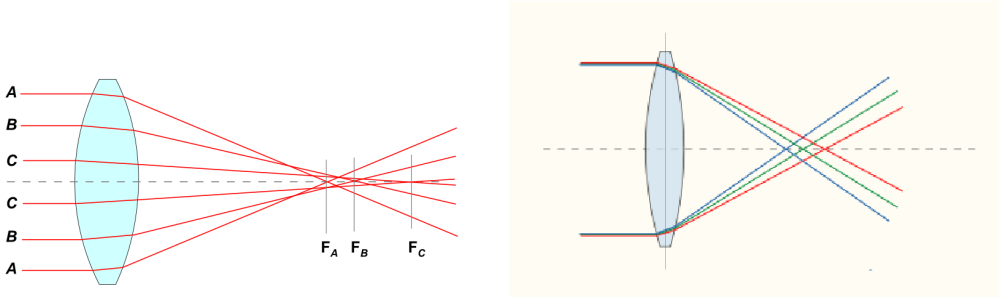
In our work, we assume all the aberration are due to the defocus imperfection, *i.e.* out-of-focus aberration, which is caused simply by out of focus. This aberration comes from the basic property of convex lens, because lens are always spherical imperfect. Another frequently referred phase is Depth of Field (DOF), which indicates the physical distance/depth, such that objects within that range could be taken acceptably clear. On the other hand, objects falling out of DOF would not be clear enough, *i.e.* 'blurred'.

We can thereby use the blurs from the focus error as a trace to estimate the distance of an object in the image. Because its distance can strongly influence the perceived scale of the captured images, we thus can estimate the size of the objects in an image. In the experiments, we use the image size and the degree of blurs to do the splice detection.

5.1. Focus System and Aberrations

Cameras use lens to focus light, but due to their optical properties, only objects within a limited range of distance from lens could be reproduced clearly. Therefore, to take a clear picture of an object, a camera needs adjust this range, which is often referred to as 'change the camera's focus', *i.e.*, adjusting the distance between its lens and sensor. This could be done automatically(auto-focus) or manually. However, there is always some degree of distortion or aberration introduced by the lens, including the spherical aberration, and chromatic aberration.

Theoretically, lens could be in the perfect shape, to simulate human eyes. But close-to-ideal, non-spherical lens are often extremely expensive. Therefore, real lens are often in spherical shape, though not ideal, much easier to be ground and polished. However this shape of lens will cause spherical aberration. Spherical aberration causes beams parallel to, but distant from the lens axis to be focused in a slightly different place than beams close to the axis, which cause a blurring of the output image, Figure 5.1(a). Chromatic aberration is caused by the fact that glass lens has different



(a) Spherical Aberration (b) Chromatic Abberation

FIGURE 5.1. Two example of Lens Aberration

dispersion rates with the lights of different wavelengths. Therefore light emitting from the same positions would be focused at the position before or after the the sensor, which causes the received signals on sensor be blurred, see Figure 5.1(b).

There are also other types of aberration caused by lens, such as filed curvature, coma, barrel and pincushion distortion and astigmatism. No matter what kind of aberration, we can see that the object at focus is more clear and sharp than those out of focus. Further, we notice that the further an object is away from the focus position, the more it will be blurred. Thus we can quantify the blur to estimate the distance and the size of the object in an image, which helps to reveal forgery.

5.2. Defocus Blur Background

Very few works are done to estimate the focus error. The attempt by Wang *et al.* [68] requires human to manually marked objects at the same distance and compare their blur consistency, which actually does not calculate the object distance.

Once again, besides blur from various lens aberrations, there are many causes for the image blur, lighting condition, scene texture, lens diffraction, signal quantization of JPEG compressing, various noises, and various operations inside camera . And out of focus aberration is only type of lens aberrations. However, as this is the main cause for image blurs, we ignore all other kinds of blurs, and focus on out-of-focus blur for simplicity.

To make use of the focus/defocus operation in cameras, our work flow is as follows. First, we find some metrics that can describe how much the blur is, based on simply

a particular part of an image. Second, according to the blur information, we need approximately estimate the distance of the objects in image when image is taken, as well as their size. Then, we can use above estimated information, and combine others to do image forensics task, such as scene validation. But before that, let us first make clear the basic lens imaging process, and how the camera calculate the sensing result from CCD/CMOS.

5.2.1. Lens Focusing Model. We here discuss more on the camera lens focusing, and set up a mathematical model for the light aberration inside camera [31].

In the real world, many cameras would employ multiple lens to get a better focus result. To ease theoretical reasoning, we assume there is one virtual lens that would give the same result. In a typical imaging device, the lens is parallel to the imaging plane, either film or CCD sensor array. Assume point P is at focus, and the light from it would come through different part of lens, and ideally focused on the image plane at another point P'. Assume that the object distance is p, the image plane distance is q, and the focal length is f. We all know the the following formula:

$$\frac{1}{q} + \frac{1}{p} = \frac{1}{f} \tag{5.2.1}$$

This formula illustrate the case that the light along the lens axle would be focused on the sensor. In this way, the object distance could be calculated once we know focused distance q, and focal length f.

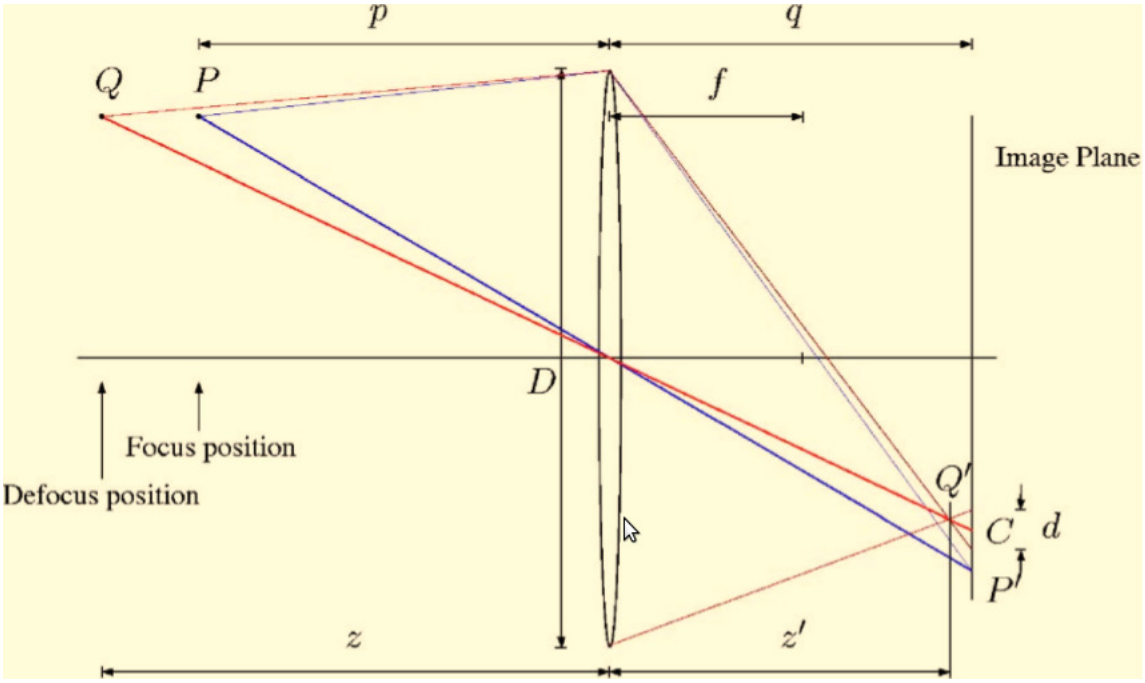


FIGURE 5.2. Math Model for Defocus Aberration

However, for a point out of focus, say Q, its image on the other side will not exactly on the sensor, which causes the out-focus-blur. And the further its portrait is away from the sensor, the more blur it will be on the output image. Therefore, the blurs caused by this mixed projection could be used to estimate the distance. For instance, the light coming from a point with a greater distance z from the lens (if $z < p$, the proof is similar.), we have

$$\frac{1}{z} + \frac{1}{z'} = \frac{1}{f} \tag{5.2.2}$$

where z' is the distance of the virtual focused position, which should be smaller than the actual focal length f. Therefore, light from point Z would form a circle at image

plane, assume its diameter is d, we have

$$\frac{d}{D} = \frac{q - z'}{z'} \tag{5.2.3}$$

Where D is the diameter of the lens. Substituting Eq. 5.2.3 into Eq. 5.2.2, we have the diameter d of the blurred circle as

$$d = Dq\left(\frac{1}{f} - \frac{1}{z} - \frac{1}{q}\right) \tag{5.2.4}$$

Note that D, f is given, and d would be estimated from image blur, and z is our final goal. But though q is the distance from the lens to the image plane, it is dynamic and unknown. Substituting Eq. 5.2.4 with Eq. 5.2, we have

$$d = \frac{Dq}{p - f}\left(\frac{1}{p} - \frac{1}{z}\right) \tag{5.2.5}$$

Therefore, we can see that the diameter d is directly related to the object distance z. For simplicity Eq. 5.2.5 could be rewritten as

$$d = c - \frac{cp}{z} \quad where \;\; c = \frac{Df}{p - f} \tag{5.2.6}$$

5.2.2. Proposed Method. From above, we can see that the more object is away from the lens focus, the more blur circle would be on the sensor. Thus, if we can determine the blur scale of an object, we can theoretically determine its distance from the lens focus point.

Further we know $D, f,\ and\ p$ can be treated as given. (D could be found at manual book, and f, p could be found from the image EXIF header information.)

Rewriting Eq.5.2.6 as a function of d, then the depth z of the object is

$$z = \frac{cp}{c-d} \quad where\ \ c = \frac{Df}{p-f} \tag{5.2.7}$$

Please note that, for the focus distance p , many cameras would only provide the 'auto', indicating that the 'auto-focus' has been used. However, with the camera at hand, we can calculate the focusing range p, with a fixed lens position at q, simply from the focus calibration.

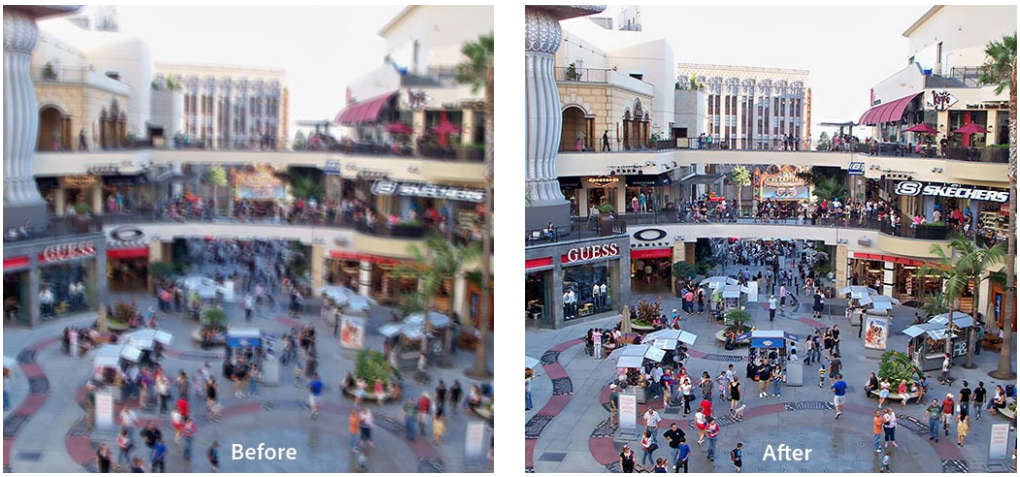
From all above, we therefore can acquire the distance z of an object,a fragment of an image in other word, from observing the blur circle size. However, due to the existence of defocus, manufacturers also try to overcome the blurred image, mainly using the point spread function inside camera.

5.2.3. Motion Blur and its Deblur. In the real digital camera, blurring artefact's is un-avoidable. Thus, deblurring, defined as the process (operation) to making a blurring image shart again, is an essential operation inside digital camera.

The deblurring problem can be formulated in the spatial variants and invariants settings. As the blur, theoretically, comes from the un-focused lighting, as well as the motion blur, which can, in most case, formulated as the convolution process around the sensor position. Due to the strong dependency on the convolution, (See Figure 5.2), deblurring is sometimes called deconvolution.

Though deblurring is an inherently ill-posed problem, people can only guess the PSF/ blur kernel. But we might better determine the PSF/ blur kernel format, according to the blur types.

We know that there are two main types of blur: motion blur, and defocus blur. Generally, the PSF/blur kernel is in a rectangle format, (see Figure 5.4).



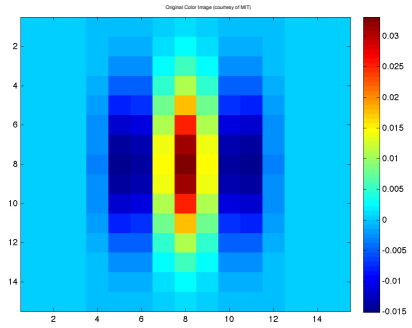
(a) deblur before (b) deblur after

FIGURE 5.3. Deblur image using Gaussian PSF (Done by Adobe Photoshop)

As to the defocus blur, if we re-examine the Figure **??**, we can see that the defocus blur kernel should be in a circular form, or at least centrosymmetry. For example, Figure 5.3 assume the 2D Gaussian Distribution, and use the estimated Point Spread Function to do deblurring, which gives an excellent result.

5.2.4. Point Spread Function. From Figure 5.5 and Equation 5.2.11, we can see that the

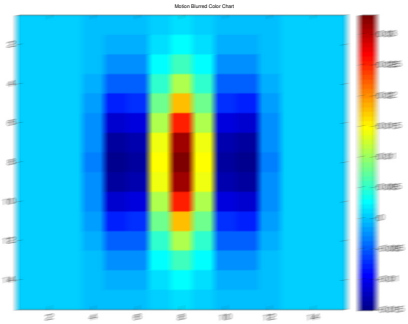
To offset the blurs, the observed image $g(x, y)$ is generally modeled as result of a 2-D convolution, which is characterized by its point spread function (PSF), $h(x, y)$.



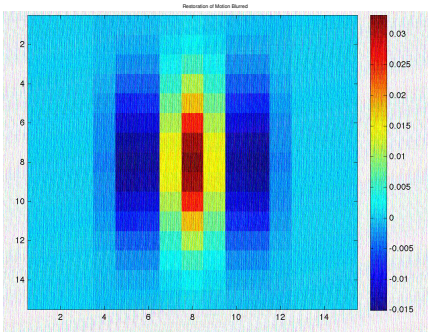
(a) Original Color Chart

0	0	0	0	0	0	12	24	36	48	29
23	50	62	74	86	98	86	74	62	50	23
29	48	36	24	12	0	0	0	0	0	0

(b) PSF of Motion blur (Result multiplied by 100)



(c) Blur chart



(d) Blind deblur result

FIGURE 5.4. Motion Deblur Example, with Blind Deblurring

Assuming the raw data from sensor is $f(x, y)$, then, the deblur operation can be formulated as

$$g(x, y) = \int_{-\infty}^{+\infty} h(x - \alpha, y - \beta) f(\alpha, \beta) d\alpha d\beta \tag{5.2.8}$$

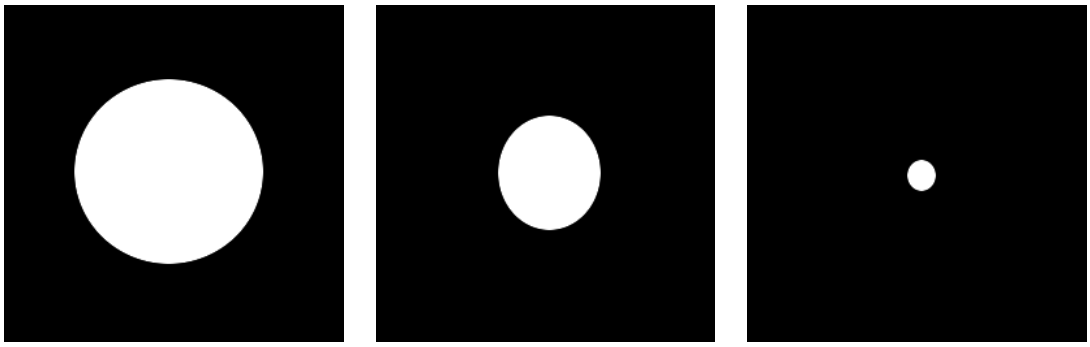
where $h(x, y)$ is a linear shit-invariant PSF. Considering a lossless camera system, we have

$$\int_{-\infty}^{+\infty} h(x, y) d\alpha d\beta = 1 \tag{5.2.9}$$

One simple PSF function for out-of-focus blur is the pillbox function

$$h(x,y) = \begin{cases} \frac{4}{\pi d}, & if x^2 + y^2 \leq \frac{d^2}{4} \\ 0, & otherwise \end{cases} \tag{5.2.10}$$

Where d is the diameter of the blur circle as mentioned in Eq. 5.2.5. We can see that every output point is the interpolation result of its surrounding point in its blur circle. (See Figure 5.5)



(a) Large Blur kernel with d = 6 (b) Median Blur kernel with d = 4 (c) Small Blur kernel with d = 2

FIGURE 5.5. Blur Kernel size of PSF

However, this function treats all its neighbors equally important, which is not generally the truth. A more convincing, and thus widely used 'blur kernel' ($h(x,y)$) assumes the Gaussian distribution of the lightings inside its blur circle.

$$h(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{5.2.11}$$

Where σ is the spread parameter, indicating how much blur occurred in that part. This model give a much better deblur result, see Figure 5.3.

It is shown that σ is proportional to the blur circle diameter d, *i.e.*,

$$\sigma = kd \quad for \;\; k > 0 \tag{5.2.12}$$

and can be determined by an appropriate calibration procedure [34]. In this way, the Eq.5.2.7 could be re-write as

$$z = \frac{c'p}{c' - \sigma} \quad where \;\; c = \frac{kDf}{p - f} \tag{5.2.13}$$

Note that till now, to calculate the exact depth, we need to know the coefficient k through calibration, requiring that the camera should be available, which is generally not satisfied. However, as we know that the most clear/sharp part should be the focused part, thus searching for the lens position or focal length of best focus, where the depth is known, and then get the coefficient k.

Moreover, in our research, we do not require the exact depth,which is, although theoretically applicable, unreliable due to the heavy noises. Rather, we are more concerned about the relative depth of the image. From all above discussion, we can see that the blur circle diameter d, or the Gaussian factor σ is the key for depth estimation. Then, we problem transfered to quantify how much a specific part of image is blurred.

5.3. Blur Estimation

Although the image has already been de-blurred, for object out of focus, the camera still only give an estimate, thus its clearness cannot compete with the object on focus. This allows us to identify the blurred part and clear part.

Though auto-focusing and defocus are not new concepts, there is still no compelling theory of defocus estimation from the output image(s) alone. However, from the forensics point of view, we believe that the blur metrics should, at least, satisfy the following 3 conditions.

(1) Using only its local points: As we need estimate different patches.

(2) Robust to noise. As fakers would add noise to image.

(3) Can deal with complex and simple scene structure.

To achieve these, we can use local pixels in spatial domain will guarantee condition 1. As to condition 2, we generally need first estimate the noise inside the image, and exclude these noise when deciding the blur. Condition 3 could be met if we use edge information. In our experiments, we use two types of method to quantify the blurs in a small patch of image.

5.3.1. Elder-Zucker Method. One famous and convincing model is explained by Elder and Zucker [14]. This method works in intensity channel, which is the green channel in RGB color space, or Y channel in $YCrCb$ color space. They also assume

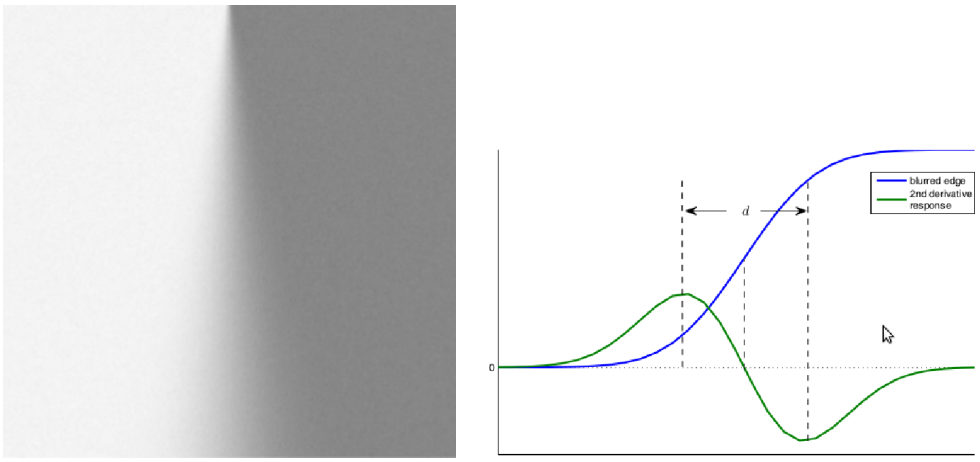
the blurriness follows a Gaussian kernel model.

$$g(x, y, \sigma_b) = \frac{1}{2\pi\sigma_b^2} e^{-1(x^2+y^2)/2\sigma_b^2} \tag{5.3.1}$$

Where, σ_b is the blur amount. To estimate the noise, we first assume the noise is the additive, zero-mean, white noise. Also, this kind of noise hold across the whole image, for simplicity, we estimate the noise as

$$n_0 = \frac{median|y_{diag}|}{0.6745} \tag{5.3.2}$$

Where y_{diag} is the HH subband at the highest level in wavelet domain, and the constant 0.6745 makes the estimate unbiased for the normal distribution.



(a) Blurred Edge

(b) Amplitude model of the blurred edge, and its second derivative

FIGURE 5.6. Blurred Edge, horizontal intensity and its 2^{nd} derivative.

As illustrated by Figure 5.5.1, the left figure is the heavily blurred image. For simplicity, assume the image is vertically equalized, thus the edge would be also

vertical, thus its intensity, from left to right is shown in figure 5.6(b). We assume the edge is located at the zero-crossing in its 2^{nd} derivative, and the blur amounts to distance between the two second derivative extrema with opposite signs.

This method is very successful. Though when estimating the edge location, as well as the blur amount σ, they assume that pixels on either side of edge is of the same value, which is generally not the case. Especially there is another edge around. Therefore, by iteratively update the edge location and blur amount, we sharp the images in a more complex scene.

5.3.2. Simple Blur Estimation. In image processing and computer vision world, the simplest method often beats more complicated one like in expert system and artificial neural network. So, we will first try the simple method, using some simple blur metrics.

Most of the existing blur metrics are based on the assumption that blur would make the edge wider, making it less sharp [17]. Thus if we assume that the edge should be only a sharp line/curve, by calculating the edge width and sharpness, we obtain a blur metrics without reference. First, we use the Canny edge detector to find the edges, we only select edges of the patch larger than 25. And the width of an edge is defined as the distance of the two local extreme points P_1, P_2, which lie along the line that perpendicular to the edge direction and lies on different side of the edge. And the width of the edge $width(x, j)$ at position (i, j) is defined as

$$width(x, j) = |P_1 - P_2| \tag{5.3.3}$$

Therefore, we can define that the blurriness of a patch of a image as

$$blurriness = \frac{1}{L}\sum_{i=0}^{N}\sum_{j=0}^{M} width(i,j) \tag{5.3.4}$$

where L is the number of the strong edge pixels.

Meanwhile, while assuming the stationary, additive, zero-mean white Gaussian noise, we estimate the noise variance from the largest flat region in our image.

5.4. Probabilistic Inference of Distance From Blur

Our work is largely inspired by the work of Held *et al.* [32], and we make some adaption for our forensics purpose.

Although in Section 5.2, we have formulate an equation such that we can directly calculate the depth of image from its blur. However, there are multiple objects are various distance, so, before we estimate their distance, we should know that segment each part out.

In this section, we use the probabilistic inference model to assign the image part, and then discuss its validity/integrity by comparing this two models.

5.5. Experiments

5.5.1. Experiment 1: 3D Image Reconstruction from Single Image. From the image EXIF header, we can always extract the image depth of field (DOF) value of an image, as well as the focal length. Also, there is also a F number, which means the $\frac{f}{D}$ value.

So, we first expose the image part, or objects that is on focus, which is achieved by comparing the sharpness metric inside each image part.

And then, the absolute depth value could be calculated as

$$u = \frac{fv}{v - f} \tag{5.5.1}$$

Where f is the focal length, which can be extracted from the image EXIF header. And from figure **??** we have

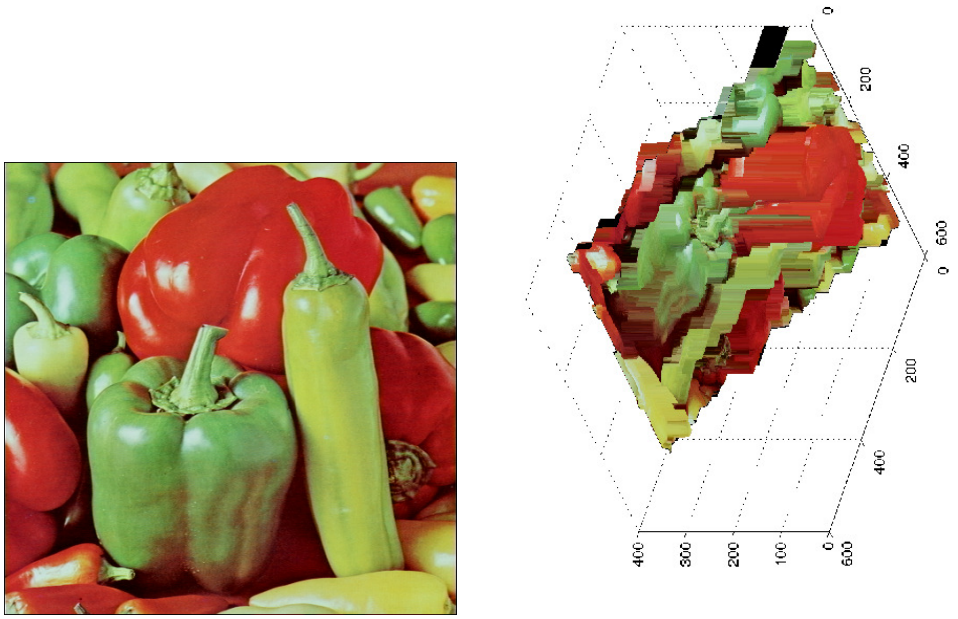
$$\frac{2R}{D} = \frac{S}{v} \tag{5.5.2}$$

Where $D = \frac{f}{F_{num}}$, and F_{num} could also be extracted from the EXIF info. Thus we have

$$v = s * \frac{2R + D}{D} \tag{5.5.3}$$

As we are only interested in the relative depth of the image, *i.e.* the absolute value of the object at focus is not needed. Therefore, in the following sections, we shrink the value s, so that distance of the object at focus is set to be 1, and other distances are compared with it.

5.5.2. Experiment 2: Splicing Detection with Sharp Edges. By assessing the blurriness, we can reveal some hidden information inside camera, such as the distance from lens and sensor plane.(though this is often dynamic.) The most inspiring application is to test the validity of the distance of an object, which is usually need the human reenforcement. By using only the blur estimation form an image, there is no much can be done. So, in our first experiment, by calculating the depth of each



(a) Blurred Edge

(b) Amplitude model of the blurred edge, and its second derivative

FIGURE 5.7. Blurred Edge, horizontal intensity and its 2^{nd} derivative.

object, we can place them in the 3D stereo view, which would make distinguishing the image rationality an easy cake.

So, our first experiment is based on the our distance estimation, and reconstruct a 3D scene by placing the the detected object inside. The first two dimensions are the estimated object size, x, y, and the third dimension is the estimated object distance. And at the validation step, we require the humans to participate the invalidation.

By facilitating participant, we allow the viewing point, originally is the camera position, be able to move, and rotate our viewing point.

5.5.3. Image Splicing Benchmarks Database. In this splicing detection dataset, we use the benchmark database from [37], which containing 363 images,

with 183 are authentic, and 180 as spliced. All these images comes from four cameras: Canon G3, Nikon D70, Canon EOS 350D and Kodak DCS330. And these authentic images are mainly taken the still images, thus exclude the motions blurs to the largest extend. And those spliced image comes from the authentic ones by simple copy-move operation.

5.5.4. Image Segmentation. In order to find the images, we need first segment the objects apart from others. Simple edge finding algorithms like Canny algorithm does not satisfy our requirement, thus to find this, we use the method from [19] to effectively label the objects. And using the edges around each object to calculate how much this position is blurred.

As we may know, we can use blur information to do the image segmentation by their distance from the camera. As objects at different depth should be separated, unless with the only reason that the whole part of image blur is decreasing or increasing with a constant speed for a large part of image, such as the image of a blacktopped highway.

Our image segmentation method is solely based on the colors, and borrow the idea from Wang *et al*. [67].

One modification in our experiments is that when calculating the kernel size, we work along the direction that is vertical to the edge tangent line.

5.5.5. Feature Extraction and Selection. After estimating the blur kernel size at each position, we extract the features from this kernel size array. In our experiment setting, we simply ignore the edges with kernel size lower than 80.

The first type of feature comes from the sorted edge kernel size , without considering the segmentation result. Assume the sorted kernel list are B

- range = max(B) - min(B)
- var = variance(B)
- std = standard deviation of B
- skewnessFea = kurtosisFea(B) (see appendix)
- kurtosisFea = kurtosisFea(B) (see appendix)
- entropy = entropy of B
- binCount = bin counts after for the blurs with bin range from 0 to 450

This type has 10 features. The other type of features are calculated along the object edges. By considering that the edges along an object should only undergone only a limited time of sudden change, we calculate the depth change for each objects. This type of feature has 4 features. So together we have 14 features for any image.

5.5.6. SVM Classification. We use penalty factor C as 400, and radial basis function with γ as 0.005 in LIBSVM. Also, we use 60% percentage image for training and 20% for feature selection and the left for testing. With 10 fold cross validation over 100 run times to record the average accuracy.

5.5.7. Classification Result and Analysis. In this setting, we get the output classification accuracy up to 81%, outperforms the work [37] by 15% in average accuracy.

From above experiments, we can see that we can estimate the image depth from a single image, which is a big success. However, when detecting the spliced region, compared with the CFA method, this blur-based method does not promising.

However, if we merely need to detect whether an image has been blurred, this method give an reliable result. After studying the failure cases, we find that images of the following types explains most of the errors.

(1) Authentic Image with no edges, see Figure 5.5.7

(2) Authentic Image with too many structured edges , see Figure 5.5.7

(3) Spliced Image with with good consistence of edge distance. see Figure 5.5.7

5.6. Discussion

need calculate blurriness along the edge.

for image part that has no edge information, such as the blue sky, our method would mark them as background, and do not estimate their distance.

need to use the most clearness part to estimate the focus length.

And we do not do the forensics by validating estimated its distance and size

(a) (b)

(c)

FIGURE 5.8. Example Images that are prone to be mis-classified.
a: Authentic images with little or no edges **b:** Authentic images with too many edges **c:** Spliced images with good consistence of edge distance.

CHAPTER 6

Summary

In the last 10 years, digital imaging devices almost completely took the place of analogue ones. While the digital images become common and widely used in all aspects of our lives, their integrity and authentication comes into question because of the advanced image manipulation software.

In this thesis, we studied the image acquisition process inside a digital camera, which is the key point that differentiates the digital photography from the artificial one. Our work focused on the three operations along the pipeline: white balance, image interpolation/demosaicking as well as the image blur due to lens imperfection.

First, by making use of the white balance residue, the source camera identification problem can be solved at a very high accuracy. What is more valuable is that our experimental result does not degrade as the number of different cameras increases, demonstrating the scalability of the proposed method. Also, even for the camera devices of the same model and brand, our proposed method is still able to distinguished among them, at a competitively high accuracy.

Secondly, we find the hidden residue pattern left by the image demosaicking process. By estimating the interpolation residue, we can check the image's integrity. In case that some part of the image is cropped from another source, we can expose this part. Two experiments prove the effectiveness of our approach.

Thirdly, we explore the image blurs in conjunction with the object depth. We assume blurs mainly comes from the lens focus-imperfection, thus excluding some blur types, such as motion blur. Our approach is based on the fact that the more an object is blurred, the farther it is from the camera focal point. Our approach can determine the spliced image, with its accuracy outperforming all the comparable published results.

We should admit that digital image forensics is still in its infancy, with only a few known types of forensic techniques. This thesis proposes three categories of solutions from three different angles, which is considerable progress in image forensics. Although we only present experiments on the very basic forensics tasks, the proposed solutions should work on almost all types of tampering. We hope that our work can inspire more progress in the image forensic area, and trigger to have more advanced tools developed to detect the variety of image manipulations. With these tools, digital images can be proved to be reliable and trustworthy.

Bibliography

[1] J. Adams, K. Parulski, and K. Spaulding. Color processing in digital cameras. 18(6):20–30, 1998.

[2] I. Avcibas, N. Memon, and B. Sankur. Steganalysis using image quality metrics. *Image Processing, IEEE Transactions on*, 12(2):221–229, 2003.

[3] K. Barnard, V. Cardei, and B. Funt. A comparison of computational color constancy algorithmspart i: Methodology and experiments with synthesized data. *IEEE transactions on Image Processing*, 11(9), 2002.

[4] K. Barnard, L. Martin, A. Coath, and B. Funt. A comparison of computational color constancy algorithmspart ii: Experiments with image data. *IEEE transactions on Image Processing*, 11(9):985, 2002.

[5] S. Bayram, H. Sencar, N. Memon, and I. Avcibas. Source camera identification based on cfa interpolation. In *Proc. IEEE International Conference on Image Processing ICIP 2005*, volume 3, pages III–69–72, 2005.

[6] S. Bravo-Solorio and A.K. Nandi. Passive forensic method for detecting duplicated regions affected by reflection, rotation and scaling. In *European Signal Processing Conference*, pages 824–828, 2009.

[7] M. Brown. *Advanced Digital Photography*. Media Publishing, 2004.

[8] G. Buchsbaum. A spatial processor model for object colour perception. *Journal of the Franklin Institute*, 310(1):1–26, July 1980.

[9] C.C. Chang and C.J. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[10] T.Y. Chang, S.C. Tai, and G.S. Lin. Distinguishing photographic images and computer graphics by using characteristics of color filter array.

[11] W. Chen, Y.Q. Shi, and W. Su. Image splicing detection using 2-d phase congruency and statistical moments of characteristic function. *Security, Steganography and Watermarking of Multimedia Contents IX, Proceeding. of SPIE, San Jose,*

CA, USA, 2007.

[12] K.S. Choi, E.Y. Lam, and K.K.Y. Wong. Automatic source camera identification using the intrinsic lens radial distortion. *Optics Express*, 14(24):11551–11565, 2006.

[13] J. Dong, W. Wang, T. Tan, and Y. Shi. Run-length and edge statistics based approach for image splicing detection. *Digital Watermarking*, pages 76–87, 2009.

[14] J.H. Elder and S.W. Zucker. Local scale control for edge detection and blur estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(7):699–716, 1998.

[15] AM Eskicioglu and PS Fisher. Image quality measures and their performance. *Communications, IEEE Transactions on*, 43(12):2959–2965, 1995.

[16] M.D. Fairchild. *Color appearance models*. Wiley, 2005.

[17] M.C.Q. Farias and S.K. Mitra. No-reference video quality metric based on artifact measurements. In *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, volume 3, pages III–141. IEEE, 2005.

[18] Micah K. Johnson Hany Farid. Exposing digital forgeries by detecting inconsistencies in lighting. *ACM Multimedia and Security Workshop 05 New York, New York USA*, 2005.

[19] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[20] Y. Feng. *PG-means: learning the number of clusters in data.* PhD thesis, 2006.

[21] T. Filler, J. Fridrich, and M. Goljan. Using sensor pattern noise for camera model identification. In *Proc. 15th IEEE International Conference on Image Processing ICIP 2008*, pages 1296–1299, October 12–15, 2008.

[22] G.D. Finlayson, M.S. Drew, and B.V. Funt. Color constancy: generalized diagonal transforms suffice. *Journal of the Optical Society of America A*, 11(11):3011–3019, 1994.

[23] G.D. Finlayson and S.D. Hordley. Improving gamut mapping color constancy. *IEEE Transactions on Image Processing*, 9(10):1774–1783, 2000.

[24] G.D. Finlayson and E. Trezzi. Shades of gray and colour constancy. In *IS&T SID's Color Imaging Conference*, pages 37–41. IS&T - The Society for Imaging

Science and Technology, 2004.

[25] DA Forsyth. A novel algorithm for color constancy. In *Color*, page 271. Jones and Bartlett Publishers, Inc., 1992.

[26] T. Frese, C.A. Bouman, and J.P. Allebach. A methodology for designing image similarity metrics based on human visual system models. In *Proceedings of SPIE/IS&T Conference on Human Vision and Electronic Imaging II*, volume 3016, pages 472–483. Citeseer, 1997.

[27] T. Gloe, K. Borowka, and A. Winkler. Feature-based camera model identification works in practice. In *Information Hiding*, pages 262–276. Springer, 2009.

[28] Thomas Gloe and Rainer Böhme. The 'dresden image database' for benchmarking digital image forensics. In *SAC '10: Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 1584–1590, New York, NY, USA, 2010. ACM.

[29] B.K. Gunturk, Y. Altunbasak, and R.M. Mersereau. Color plane interpolation using alternating projections. *IEEE Transactions on Image Processing*, 11(9):997–1013, 2002.

[30] B.K. Gunturk, J. Glotzbach, Y. Altunbasak, R.W. Schafer, and R.M. Mersereau. Demosaicking: color filter array interpolation. *Signal Processing Magazine, IEEE*, 22(1):44–54, 2005.

[31] E. Hecht. Optics (4th edn), 2002.

[32] R.T. Held, E.A. Cooper, J.F. OBRIEN, and M.S. Banks. Using blur to affect perceived distance and size. *ACM transactions on graphics*, 29(2), 2010.

[33] K. Hirakawa and T.W. Parks. Adaptive homogeneity-directed demosaicing algorithm. *IEEE Transactions on Image Processing*, 14(3):360–369, 2005.

[34] B. Horn. *Robot vision*. The MIT Press, 1986.

[35] M.F. Hossain, M.R. Alsharif, and K. Yamashita. An effective edge-adaptive color demosaicking algorithm for single sensor digital camera images. In *Emerging Intelligent Computing Technology and Applications: 5th International Conference on Intelligent Computing, ICIC 2009 Ulsan, South Korea, September 16-19, 2009 Proceedings*, page 317. Springer, 2009.

[36] C.W. Hsu, C.C. Chang, C.J. Lin, et al. A practical guide to support vector classification, 2003.

[37] Y.-F. Hsu and S.-F. Chang. Detecting image splicing using geometry invariants and camera characteristics consistency. In *Proc. IEEE International Conference on Multimedia and Expo*, pages 549–552, July 9–12, 2006.

[38] Y.F. Hsu and S.F. Chang. Image splicing detection using camera response function consistency and automatic segmentation. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 28–31. IEEE, 2007.

[39] Hailing Huang, Weiqiang Guo, and Yu Zhang. Detection of copy-move forgery in digital images using sift algorithm. volume 2, pages 272 –276, dec. 2008.

[40] A. Hyvärinen and E. Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–430, 2000.

[41] M. K. Johnson and H. Farid. Exposing digital forgeries in complex lighting environments. 2(3):450–461, September 2007.

[42] M. Kirchner. Efficient estimation of cfa pattern configuration in digital camera images. In *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 7541, page 35, 2010.

[43] Matthias Kirchner. Fast and reliable resampling detection by spectral analysis of fixed linear predictor residue. In *MM&Sec '08: Proceedings of the 10th ACM workshop on Multimedia and security*, pages 11–20, New York, NY, USA, 2008. ACM.

[44] KM Lam. Metamerism and colour constancy. 1985.

[45] E.H. Land. The retinex theory of color vision. *Scientific American*, 237(6):108–128, December 1977.

[46] Tran Van Lanh, Kai-Sen Chong, S. Emmanuel, and M.S. Kankanhalli. A survey on digital camera image forensic methods. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 16 –19, july 2007.

[47] Bin Li, Y. Q. Shi, and Jiwu Huang. Detecting doubly compressed jpeg images by using mode based first digit features. In *Proc. IEEE 10th Workshop on Multimedia Signal Processing*, pages 730–735, October 8–10, 2008.

[48] Z. Lin, R. Wang, X. Tang, and H.Y. Shum. Detecting doctored images using camera response normality and consistency. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 1087–1092. IEEE, 2005.

[49] BJ Lindbloom. Chromatic adaptation evaluation. *Bruce J. Lindbloom, Tech. Rep*, 2007.

[50] Yangjing Long and Yizhen Huang. Image based source camera identification using demosaicking. In *Proc. IEEE 8th Workshop on Multimedia Signal Processing*, pages 419–424, 2006.

[51] J. Lukáš, J. Fridrich, and M. Goljan. Detecting digital image forgeries using sensor pattern noise. In *Proceedings of the SPIE*, volume 6072, page 15. Citeseer, 2006.

[52] J. Lukas, J. Fridrich, and M. Goljan. Digital camera identification from sensor pattern noise. 1(2):205–214, June 2006.

[53] Pan Feng Luo Weiqi, Qu Zhenhua and Huang Jiwu. A survey of passive technology for digital image forensics. *Frontiers of Computer Science in China*, pages 166–179, 2007.

[54] S. Lyu and H. Farid. How realistic is photorealistic? *IEEE Transactions on Signal Processing*, 53(2):845–850, 2005.

[55] H.R. Miller. Color filter array for ccd and cmos image sensors using a chemically amplified thermally cured pre-dyed positive-tone photoresist for 365-nm lithography. In *Proceedings of SPIE*, volume 3678, page 1083, 1999.

[56] T.T. Ng, S.F. Chang, and Q. Sun. Blind detection of photomontage using higher order statistics. In *Circuits and Systems, 2004. ISCAS'04. Proceedings of the 2004 International Symposium on*, volume 5, pages V–688. IEEE, 2004.

[57] A.C. Popescu and H. Farid. Exposing digital forgeries by detecting duplicated image regions. *Technical Report, TR2004-515, Dartmouth College, Computer Science*, pages 100–1000, 2004.

[58] Alin C. Popescu and Hany Farid. exposing digital forgeries by detecting trace of resampling. pages 100–1000, 2005.

[59] Alin C. Popescu and Hany Farid. Exposing digital forgeries in color filter array interpolated images. *IEEE Trans On Signal Processing*, 53 no.10:3948–3959, 2005.

[60] V.M. Potdar, S. Han, and E. Chang. A survey of digital image watermarking techniques. In *Proceedings of IEEE third international conference on industrial informatics, INDIN05*, pages 709–16, 2005.

[61] S. Prasad and KR Ramakrishnan. On resampling detection and its application to detect image tampering. In *Multimedia and Expo, 2006 IEEE International Conference on*, pages 1325–1328. IEEE, 2006.

[62] K. San Choi, E.Y. Lam, and K.K.Y. Wong. Source camera identification using footprints from lens aberration. *Digital Photography II SPIE*, 6069(1):172–179, 2006.

[63] Nasir Memon b Sevinc Bayram a, Husrev T. Sencar b. improvements on source camera-model identification based on cfa interpolation. *Proc. of the WG 11.9 Intl. Conf. on Digital Forensics*, 9, 2005.

[64] Yun Q. Shi, Chunhua Chen, and Wen Chen. A natural image model approach to splicing detection. In *MM&Sec '07: Proceedings of the 9th workshop on Multimedia & security*, pages 51–62, New York, NY, USA, 2007. ACM.

[65] C.Y. Tsai and K.T. Song. A new edge-adaptive demosaicing algorithm for color filter arrays. *Image and Vision Computing*, 25(9):1495–1508, 2007.

[66] J. van de Weijer, T. Gevers, and A. Gijsenij. Edge-based color constancy. *IEEE Transactions on Image Processing*, 16(9):2207–2214, 2007.

[67] Nan Wang and Haizhou Ai. Who blocks who: Simultaneous clothing segmentation for grouping images. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1535 –1542, nov. 2011.

[68] Xin Wang, Bo Xuan, and Si long Peng. Digital image forgery detection based on the consistency of defocus blur. pages 192 –195, aug. 2008.

[69] G. West and M.H. Brill. Necessary and sufficient conditions for von kries chromatic adaptation to give color constancy. *Journal of Mathematical Biology*, 15(2):249–258, 1982.

[70] Jing Zhang, Haiying Wang, and Yuting Su. Detection of double-compression in jpeg2000 images. In *Proc. Second International Symposium on Intelligent Information Technology Application IITA '08*, volume 1, pages 418–421, 20–22 Dec. 2008.

[71] L. Zhang and X. Wu. Color demosaicking via directional linear minimum mean square-error estimation. *IEEE Transactions on Image Processing*, 14(12):2167–2178, 2005.

Appendix

0.1. Appendix

6.1.1. Image Quality Metrics. In this section, we give brief descriptions of some image quality features in Table **??**. Given an original RGB image $C(i, j, k)$ of size M-by-N, where $i = 1, 2, ..., M$, $j = 1, 2, ..., N$ and $k = 1, 2, 3$ indicating its color plane. And $\hat{C}(i, j, k)$ stands for the re-balanced image.

A. Minkowsky Metrics. Minkowsky metrics (M) could be used to assess the similarity of two images, by averaging the pixel-by-pixel difference.

$$M_\gamma(k) = \left\{ \frac{1}{M * N} \sum_{i=1,j=1}^{M,N} |C(i, j, k) - \hat{C}(i, j, k)|^\gamma \right\}^{1/\gamma} \tag{1}$$

$\gamma = 1$ corresponds to Mean Absolute Error (MAE), $\gamma = 2$ corresponds to rooted Mean Square Error (RMSE), meaning that $MSE(k) = RMSE^2(k)$. When $\gamma = +\infty$, we actually get the maximum difference.

To get the Normalized Mean Square Error (NMSE), we normalize the square error by the quadratic of the baseline image.

$$NMSE(k) = \frac{M * N * MSE(k)}{\sum_{i=1}^{M} \sum_{j=1}^{N} C^2(i, j, k)} \tag{2}$$

Similarly, Peak Signal to Noise Ratio(PSNR) is actually the reciprocal of RMSE, and scaled by its logarithm.

$$PSNR(k) = 20 * log_{10} \frac{255}{RMSE(k)} \tag{3}$$

B. Correlation Metrics. Another types of metrics, focused on non-negative components, is the Czekanowski Correlation (CC) [2].

$$CC = 1 - \frac{2 * \sum_{i=1,j=1}^{M,N} \sum_{k=1}^{3} min\left(C(i,j,k), \hat{C}(i,j,k)\right)}{\sum_{i=1,j=1}^{M,N} \sum_{k=1}^{3} \left(C(i,j,k) + \hat{C}(i,j,k)\right)} \tag{4}$$

Another correlation based metric is the Angle Mean (AM), assuming at position (i,j), the RGB values of image C is denoted by bold character $\boldsymbol{C}$(i,j) as a 3-D vector.

$$AM = 1 - \frac{1}{M * N} \sum_{i=1,j=1}^{M,N} \frac{2}{\pi} \frac{\left\langle \mathbf{C}(i,j), \hat{\mathbf{C}}(i,j) \right\rangle}{||\mathbf{C}(i,j)|| * ||\hat{\mathbf{C}}(i,j)||} \tag{5}$$

Similarly, Correlation Qualities (CQ) and Structural Contents (SC) are

$$CQ(k) = \frac{\sum_{i=1,j=1}^{M,N} C(i,j,k) * \hat{C}(i,j,k)}{\sum_{i=1,j=1}^{M,N} C^2(i,j,k)} \tag{6}$$

$$SC(k) = \frac{\sum_{i=1,j=1}^{M,N} C^2(i,j,k)}{\sum_{i=1,j=1}^{M,N} \hat{C}^2(i,j,k)} \tag{7}$$

C. Quality Metrics in Frequency Domain. Median Block Weighted metrics are calculated by first doing discrete Fourier transform, extracting their weighted metrics for each block, and then use the max, mean, median value as potential feature[15].

Finally, to get Human Visual System (HVS) based metrics, users need first transform image data into HVS mdoel, details could be found in [26].

0.2. Why white balance is idempotence

Idempotence refers to the function property that applying it multiple times, the result stays the same.

$$F(F(x)) = F(x) \tag{1}$$

This property holds for the white balance. To be more specific, we explain these algorithms separately in the following.

6.2.0.1. *Max-RGB algorithm.* This type of algorithm all follows an assumption that the white points inside an image should follow a constant way, either number of whites, the percentage, or even more complex. But no matter what these algorithms varies, once the image have satisfied this assumption, there would no need for the algorithm to change.

6.2.0.2. *Gray-World assumption.* This is a big category of white balance algorithms. This type of algorithms assume the average color of an image that is recorded under a white light source is achromatic. For all images that have already been balanced, we can easily seen that those average color is already been set as achromatic.

Due to the fact that the WB setting may differs from each other, we know that all cameras would treat the lighting environments separately, say, daylight, cloudy, flash, shade, *etc*. These would first transfer the lighting into the white light source and then do the white balance. Therefore, we can simply treat all images before white balance inside the camera as taken under white light source. Therefore, idempotent property holds this this assumption.

6.2.0.3. *Gamut-based method.* First, camera uses a training set of images or patches to learn this limited set of colors, called *canonical gamut*. Then it computes a set of possible mappings that transforms the input gamut completely inside the canonical gamut.

By applying this WB method the second time, the canonical gamut would stay the same. Therefore, we do not need to do the mapping at all, *i.e*. the color adaptation would compute the transformation matrix as an identity one.

0.3. Support Vector Machine

A Support Vector Machine (SVM) is a discriminative classifier. It use labeled training data to calculate an optimal hyperplane which optimally classifies the existing training data.

Common cost function of classifiers are

$$\max_{\theta} \quad \frac{1}{m}\Sigma_{\imath=1} cost_1(\theta^T)x^{(i)}) + (1-y^{(i)})cost_0(\theta^T x^{(i)}) + \frac{\lambda}{2m}\Sigma_{i=0}^{n}\theta_j^2 \tag{1}$$

After the transformation, we have the cost function for SVM as

$$\max_{\theta} \quad \frac{1}{m}C * \Sigma_{i=1}^{m}\left(y^i cost_1(\theta^T x^{(i)} + (1-y^{(i)})cost_0(\theta^T x^{(i)}))\right) + \frac{\lambda}{2m}\Sigma_{i=0}^{n}\theta_j^2 \tag{2}$$

There are various kernel function, $cost_i(\theta, x)$, such as Linear function, polynomial , Gaussian radial basis and hyperbolic tangent [9].

0.4. Error Metrics for Skewed Classes

In prediction, there are some basic metric to measure a classifier. For 0-1 classifier, assume that we have the following table. Where TP is the true positive, FP is the

TABLE 6.1. Multi-column and multi-row table

	Actual Class	
Predicted Class	True Positive	False Positive
	False Positive	True negative

false positive, FP is the false positive and TN is the true negative.

Some metrics that used in this thesis are described as follows.

$$accuracy(ACC) = (TP + TN)/(TP + TF + FP + FN)$$

$$sensitivity(Recall) = TP/(TP + FN)$$

$$precision = TP/(TP + FP)$$

$$F1score = 2 * TP/(2TP + FP + FN)$$

0.5. A: EM algorithm pseudo-code

/* initialize parameters */

choose { $\alpha^0_{u,v}$ randomly

choose N and θ_0

set p_0 as 1 over the size of range of possible values of $f(x, y)$

n=0

while ($\sum_{u,v} |\alpha^n_{u,v} - \alpha^(_{u,v} n - 1)|$)

{

/* Expectation step */

foreach location (x, y) /* update its residual error */

$r(x, y) = |f(x, y) - \sum^N_{u,v=N} \alpha^n_{u,v} f(x - u, y - v)|$

foreach location (x, y)

$P(x, y) = \frac{1}{\sigma_n \sqrt{2\pi}} exp[-r^2(x, y)/2\sigma^2_n]$

/* conditional probability */

$w(x, y) = \frac{P(x,y)}{P(x,y)+p_0}$ /* posterior probability */

/* Maximization step */

update the probability that $\{\alpha^{n+1}_{u,v}\}$ belongs to class 1

$\sigma_{n+1} = \left(\frac{\sum_{x,y} w(x,y) r^2(x,y)}{\sum_{x,y} w(x,y)}\right)^{0.5}$ /* new vairance estimate */

n=n+1

} /* end of while loop*/

0.6. B: Independent Component Analysis

Independent Component Analysis (ICA) aims to extract original signals from the mixture of their independent sources, without any priori information on the mixture process [40]. ICA solely uses the statistical "latent vairables" model, to better define it, assume a set of observed signal $x_1, x_2, ..., x_n$, each is mixed by another n independent components.

$$x_j = a_{j1}s_1 + a_{j2}s_2 + ... + a_{jn}s_n \quad for\ all\ j \tag{1}$$

Note that when the number of observed signals could be larger than that of potential signals, but in that way, we have the redundant signals, which would theoretically give the same result. Tough due to the existence of noise, in reality, we can combine them to better estimate the potential signal.

If we use bold lower case letters to represent the column vector, and bold upper case letter for the matrix. We can denote $\mathbf{x}$ as the mixtures $x_1, x_2, ..., x_n$, and likewise $\mathbf{s}$ as $s_1, x_2, ..., x_n$, we have

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{2}$$

If we need the columns of matrix $\mathbf{A}$, we use a_j to represent it.

Note that we assume that the independent component must have *nongaussian* distribution. To get the original signal $\mathbf{s}$, we would only need to estimate $\mathbf{A}$ by this nongaussian property. After that, we can simply get the original independent components by its inverse,

$$\mathbf{s} = \mathbf{A}^{-1}\mathbf{x} \tag{3}$$

To use nongaussianity in ICA estimation, we need set a quantitative measure of nongaussianity. Without loss of generality, we assume variable $\mathbf{x}$ is already zero-mean, and have unit variance, and independent with each other. Most people use

Kurtosis to quantify nongaussinitary, which is defined as :

$$kurt(x) = E\{x^4\} - 3(E\{x^2\})^2 \tag{4}$$

The successful of Kurtosis come from its cumulant:

$$kurt(s_1 + s_2) = kurt(s_1) + kurt(s_2) \tag{5}$$

$$kurt(\alpha s) = \alpha^4 kurt(s) \tag{6}$$

Another popular measure of nongaussianity is Negentropy

$$J(y) \sim \frac{1}{12} Ey^{3^2} + \frac{1}{48} kurt(y)^2 \tag{7}$$

ICA estimation is variation of the maximum likelihood estimation, its computational procedure is as follows:

(1) Whiten the data to give x /* zero-mean, unit vairance */

(2) Set iteration count $i = 1$

(3) Take a random vector w_i

(4) Maximize nongaussinity of $w_i^T x$, under constrains $||w_i|| = 1$ and $w_i^T w_j = 0, j < i$

(5) increase iteration count by 1, jump to step 3

In step 4, updating the w_i could be done by traditional gradient method, though most people are using FastICA fixed-point algorithm for speed consideration.