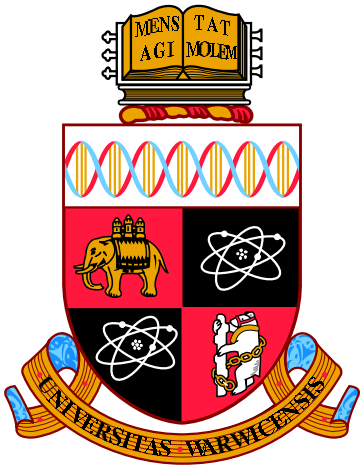



# Digital Image Forensics Based on Sensor Pattern Noise

by

**Xufeng Lin**

**Thesis**

Submitted to The University of Warwick

for the degree of

**Doctor of Philosophy**

**Department of Computer Science**

September 2016

THE UNIVERSITY OF WARWICK

# Contents

# Acknowledgements

Foremost, I would like to express my sincere gratitude and utmost respect to my supervisor, Prof. Chang-Tsun Li, for his tremendous academic support and valuable career advice. I would like to thank him for providing me with so many opportunities to shape me into a research scientist. Without his guidance and encouragement, this PhD would not have been achievable. I am very grateful to my advisors, Dr. Victor Sanchez and Dr. Nathan Griffiths, for their inspiring guidance and valuable advice on my PhD progress.

I would like to thank my lab mates: Dr. Xingjie Wei, Dr. Yi Yao, Dr. Yu Guan, Ning Jia, Ruizhe Li, Alaa Khadidos, Roberto Leyva, Xin Guan, Qiang Zhang, Justin Chang, Shan Lin, Bo Wang, Bo Gao, Chao Chen, Huanzhou Zhu, Zhuoer Gu, and Portos Portis. Completing this work would have been all the more difficult were it not for the support and friendship provided by them.

Finally, I would like to express my deepest gratitude to my parents, sister, grandaunt and her husband for their constant support and unconditional love. Regardless of the ups and downs in life, they were always there to help me and stood by my side.

# Declarations

I hereby declare that the work presented in this thesis entitled *Digital Image Forensics Based on Sensor Pattern Noise* is my own work and has not been submitted to any college, university or any other academic institution for the purpose of obtaining an academic degree.

Xufeng Lin

Signature:

Date:

# Publications

[1] **Xufeng Lin** and Chang-Tsun Li, **Large-Scale Image Clustering Based on Camera Fingerprints**, accepted for publication in *IEEE Transactions on Information Forensics and Security*, 2016.

[2] **Xufeng Lin** and Chang-Tsun Li, **Refining PRNU-Based Detection of Image Forgeries**, in *Proceedings of IEEE Digital Media & Academic Forum*, Santorini, Greece, 4-6 July, 2016.

[3] **Xufeng Lin** and Chang-Tsun Li, **Enhancing Sensor Pattern Noise via Filtering Distortion Removal**, *IEEE Signal Processing Letters*, 23(3):381-385, 2016.

[4] **Xufeng Lin** and Chang-Tsun Li, **Preprocessing Sensor Pattern Noise via Spectrum Equalization**, *IEEE Transactions on Information Forensics and Security*, 11(1):126-140, 2016.

[5] **Xufeng Lin** and Chang-Tsun Li, **Two Improved Forensic Methods of Detecting Contrast Ehancement in Digital Images**, in *Proceedings of SPIE International Conference on Media Watermarking, Security, and Forensics*, February 3-5, 2014, San Francisco, California, US.

[6] **Xufeng Lin**, Chang-Tsun Li, and Yongjian Hu, **Exposing Image Forgery through the Detection of Contrast Enhancement**, in *Proceedings of IEEE International Conference on Image Processing*, September 15-18, 2013, Melbourne, Australia.

# Abstract

With the advent of low-cost and high-quality digital imaging devices and the availability of user-friendly and powerful image-editing software, digital images can be easily manipulated without leaving obvious traces. The credibility of digital images if often challenged when they are presented as crucial evidence for news photography, scientific discovery, law enforcement, etc. In this context, digital image forensics emerges as an essential approach for ensuring the credibility of digital images.

Sensor pattern noise mainly consists of the photo response non-uniformity noise arising primarily from the manufacturing imperfections and the inhomogeneity of silicon wafers during the manufacturing process. It has been proven to be an effective and robust device fingerprint that can be used for a variety of important digital image forensic tasks, such as source device identification, device linking, and image forgery detection. The objective of this thesis is to design effective and robust algorithms for better fulfilling the forensic tasks based on sensor pattern noise.

We found that the non-unique periodic artifacts, typically shared amongst cameras subjected to the same or similar in-camera processing procedures, often give rise to false positives. These periodic artifacts manifest themselves as salient peaks in the magnitude spectrum of reference sensor pattern noise. We propose a spectrum equalization algorithm to detect and suppress the salient peaks in the magnitude spectrum of reference sensor pattern noise, aiming to improve the accuracy and reliability of source camera identification based on sensor pattern noise. We also propose a framework for large-scale image clustering based on device fingerprints (sensor pattern noises). The proposed clustering framework deals with large-scale and high-dimensional device fingerprint databases and is capable of overcoming the $NC \gg SC$ problem, i.e., the number of cameras is much higher than the average number of images acquired by each camera. Additionally, for the task of image

forgery detection based on sensor pattern noise, we propose a refining algorithm to solve the missing detection problem along the boundary area between the forged and non-forged regions.

The proposed algorithms are evaluated on either a public benchmarking database and our own image databases. Experimental results, as well as the comparisons with state-of-the-art algorithms, confirm their effectiveness and robustness.

# Abbreviations

| | |
|---|---|
| **AWGN** | Additive White Gaussian Noise |
| **AUC** | Area Under ROC Curve |
| **BM3D** | Block Matching and 3D filtering |
| **CCN** | Circular Cross-correlation Norm |
| **CD-PRNU** | Color-Decoupled Photo Response Non-Uniformity |
| **CFA** | Color Filter Array |
| **CFAR** | Constant False Acceptance Rate |
| **CLT** | Central Limit Theorem |
| **CRF** | Camera Response Function |
| **DFT** | Digital Fouerior Transform |
| **DWT** | Digital Wavelet Transform |
| **DSNU** | Dark Signal Non-Uniformity |
| **FAR** | False Acceptance Rate |
| **FRR** | False Rejection Rate |
| **FPR** | False Positive Rate |
| **HHD** | Hard Disk Drive |
| **IDFT** | Inverse Discrete Fourier Transform |
| **KPCA** | Kernel Principal Component Analysis |
| **LDA** | Linear Discrimination Analysis |
| **LSE** | Lease Square Estimator |
| **MAP** | Maximum A Posteriori |
| **MLE** | Maximum Likelihood Estimation |
| **MRF** | Markov Random Field |
| **NCC** | Normalized Correlation Coefficient |
| **PAA** | Piecewise Aggregate Approximation |

| | |
|---|---|
| **PCA** | Principal Components Analysis |
| **PCAI** | Predictor based on Context Adaptive Interpolation |
| **PCAI8** | 8-neighborhood Predictor based on Context Adaptive Interpolation |
| **PCE** | Peak-to-Correlation Energy |
| **PRNU** | Photo Response Non-Uniformity |
| **QMF** | Quadrature Mirror Filter |
| **ROC** | Receiver Operating Characteristic |
| **ROI** | Region of Interest |
| **SCI** | Source Camera Identification |
| **SEA** | Spectrum Equalization Algorithm |
| **SPCE** | Signed Peak-to-Correlation Energy |
| **SPN** | Sensor Pattern Noise |
| **SSD** | Solid State Drive |
| **SVD** | Singular Value Decomposition |
| **SVM** | Support Vector Machine |
| **TPR** | True Positive Rate |
| **WF** | Wiener Filtering in DFT domain |
| **ZM** | Zero-Meaning |

# List of Figures

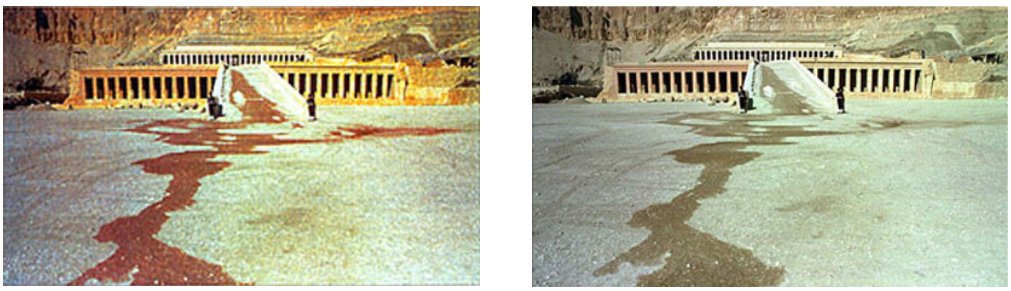
# List of Tables

# CHAPTER 1

# Introduction

## 1.1 Digital Image Forensics

Advances in digital imaging technologies have led to the development of low-cost and high-quality digital imaging devices, such as camcorders, digital cameras, scanners and built-in cameras of smartphones. The ever-increasing convenience of image acquisition has facilitated the distribution and sharing of digital images, and bred the pervasiveness of powerful image editing tools, allowing even unskilled persons to easily manipulate digital images for malicious or criminal ends. Under the circumstances where digital images serve as critical evidence for news photography, scientific discovery, law enforcement, etc., maliciously manipulated images may lead to serious consequences.



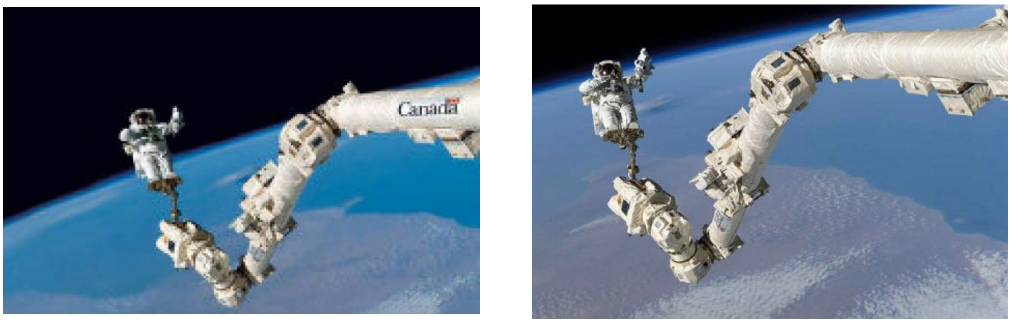
(a) (b)

Figure 1.1: Misleading report on Luxor massacre by the Swiss tabloid Blick in 1997. (a) Altered image. (b) Original image.

Fig. 1.1(a) shows a image posted by the Swiss tabloid Blick after 62 tourists (including 36 Swiss tourists) were killed in a terrorist attack at the temple of Hatshepsut in Luxor Egypt[1]. A puddle of water in the original image (Fig. 1.1(b)) was

[1] http://pth.izitru.com/1997_11_00.html

digitally altered to appear as blood flowing from the temple. Fig. 1.2(a) shows the American astronaut Stephen Robinson on a spacewalk in front of the Canadarm2[2]. Unfortunately, The Economist pointed out that the Canadian logo visible on the arm in the image was crudely composited, because the background color and texture of the logo clearly does not match the arm. In fact, it was discovered that the original image (Fig. 1.2(b)) without any Canadian logo was captured by NASA. The composite image was used by several Canadian government websites to promote Canada's involvement with the International Space Station. Tampered image presented as evidence in the law enforcement system can be used to frame innocent people unjustly. Kanhaiya Kumar, a student leader at India's Jawaharlal Nehru University, was arrested on sedition charges in the midst of a variety of contested images and video clips which claim to show him advocating for the disintegration of India[3]. One image (Fig. 1.3(a)) that circulated widely on social media shows Kumar speaking in front of a map showing a divided India, in which portions of Kashmir and Gujarat were annexed to Pakistan. However, in the original, undoctored image (Fig. 1.3(b)), Kumar is actually speaking in front of a blank background.




(a) (b)

Figure 1.2: An example of forged images used in the scientific discovery field. (a) A composite image used by several Canadian government websites to promote Canada's involvement with the International Space Station and (b) the original image.

As can be seen in the above examples, we are now living in a world of "seeing

[2] http://pth.izitru.com/2014_10_00.html
[3] http://pth.izitru.com/2016_02_01.html


(a) (b)

Figure 1.3: An example of forged images used by law enforcement. (a) A doctored image showing Kumar speaking in front of a map of a divided India and (b) the original image.

is not believing". Digital images can be easily forged with increasingly powerful and user-friendly image editing software without leaving visible traces. The increasing appearance of digitally altered forgeries on the Internet and in mainstream media is casting doubt on the integrity of digital images. Untrustworthy images have challenged the public credibility, judicial impartiality, academic morality, journalistic integrity, etc., therefore, it is important and necessary to carry out research on forensic technologies that can verify the originality, integrity and authenticity of digital images.

Digital image forensics research aims at revealing underlying facts about a suspicious image [10]. It is not limited to exposing image forgeries, as shown in the above examples, but involves a wide range of investigations. It is intended to answer the questions including but not limited to:

- Which device among the candidates was used to capture the image in question?
- Which images among the candidates were captured by the same source device X?
- Is this image an original image or has it been tampered with for ulterior motives?

- What parts of the image has been forged and up to what extent?
- What is the processing history of the image?

### 1.1.1 Active Digital Image Forensics

To determine the originality, integrity and authenticity of a digital image, a straightforward way is to insert extra digital watermark [11–17] or signature [18–21] into the image when it is created. The changes in the image are then detected by recovering and checking the embedded digital signature or watermark. For example, Fridrich and Goljan [13] proposed two techniques to embed an image in itself both for protecting the image content and for authentication. In [14], a semi-fragile watermarking method was proposed for the automatic authentication and restoration of the image content using irregular sampling. In spite of the effectiveness of these active techniques, they can only be applied when the image is protected at the origin. Nowadays, the majority of images do not contain a digital watermark/signature mainly due to the following reasons:

- Camera manufacturers have to devise extra digital watermark/signature embedding components in the camera, so only some high-end cameras have watermark embedding features.
- The embedded watermark/signature may degrade the image quality and significantly reduce the market value of cameras featuring watermark/signature embedding capability.
- The successful implementation of watermark-based protection requires close collaborations among publishers/manufaturers, investigators and potentially trusted third-party organizations. This limits the adoption of digital watermark/signature based techniques.

### 1.1.2 Passive Digital Image Forensics

Compared to the above active forensic technology, passive image forensic technologies require no collaboration of users and is based on the inherent patterns in the image itself. Most image manipulations do not leave noticeable visual artifacts, but they do inevitably change the statistical characteristics of the manipulated image. Passive forensic technologies verify the originality, integrity and authenticity of images through detecting the changes of statistical image characteristics. It does not need to insert extra information in the image and there is no special requirement on the imaging hardware. Therefore, there has been growing interest in passive forensic techniques. Over the past few years, the research on passive image forensics mainly has focused on the following three directions [10]:

1. Image source identification to determine the data acquisition device that generates one image. It identifies the source device at four different accuracy levels:

    - Device type: For example, is the given image generated by a camera or a scanner?
    - Device brand: For example, is the given image taken by a Canon Camera or a Nikon camera?
    - Device model: For example, is the given image taken by Canon 450D or Canon IXUS70?
    - Individual device: For example, is the give image taken by this specific camera or another camera?

2. Discrimination of synthetic images images from real images to identify computer-generated images which do not depict a real-life occurrence. As computer graphics technology develops, it becomes increasingly difficult to distinguish photographic images from computer-generated ones by naked eyes. There has

been a trend of presenting synthetic images as real images or augmenting the computer-generated virtual scenes with real scenes to deceive the viewer.

3. Image forgery detection to determine whether or not an image has undergone any form of modification or processing after it was originally captured. It also uncovers the parts of the given image that have been altered and up to what extent.

As shown in Fig. 1.4, passive image forensics can be divided into three categories based on the characteristics of the used images:

1. Passive image forensics based on natural image statistics: For example, Lyn and Farid [22] decomposed each color channel of a color image using the separable quadrature mirror filters (QMFs) [23, 24], then calculated the mean, variance, skewness, and kurtosis of the subband coefficient histograms at each orientation and scale. They also considered the linear prediction errors of coefficient magnitudes between adjacent pixels, scales, directions and color channels. These features were input into a linear discrimination analysis (LDA) or a nonlinear support vector machine (SVM) classifier to differentiate between photorealistic (computer-generated) and photographic images. In [25], Kharrazi et al. proposed 34-dimensional image features, including the mean value of the RGB channels (3 features), the correlation pairs between RGB channels (3 features), neighbor distribution center of mass for each of the 3 channels (3 features), RGB pairs energy ratio (3 features), wavelet domain statistics for 3 sub-bands in 3 color channels (9 features) as well as 13 image quality metrics features, to distinguish the images taken by different cameras.

2. Passive image forensics based on traces left by specific image manipulations: A counterfeiter has to apply one or more image manipulations to forge an image. A specific image manipulation will leave some specific patterns in the forged image. The techniques of this category reveal the potential image

forgeries by looking for the traces left by specific image manipulations such as double JPEG compression [26, 27], resampling [28, 29], contrast enhancement [30–34], etc. It also includes the methods that detect the duplicated image regions [35–37] or the inconsistent lighting conditions [38, 39] introduced by copy-and-paste image forgery.

3. Passive image forensics based on consistent characteristics introduced by various imaging devices or different hardware or software processing components in the imaging pipeline: Shown in Fig. 1.5 is the simplified imaging pipeline in digital camera. Each processing component in the pipeline will leave specific and consistent characteristics in the resulting image. Therefore, by detecting the presence of the consistent characteristics introduced by imaging device, we can trace back to the image source. Likewise, we can also uncover the image forgery by detecting the absence of those consistent characteristics. The consistent characteristics include the characteristics left by lens distortion [40], color filter array (CFA) interpolation (or demosaicing) [28, 41], camera response function (CRF) [41–43], JPEG artifacts [44] and sensor pattern noise (SPN) [1, 3, 9, 45].

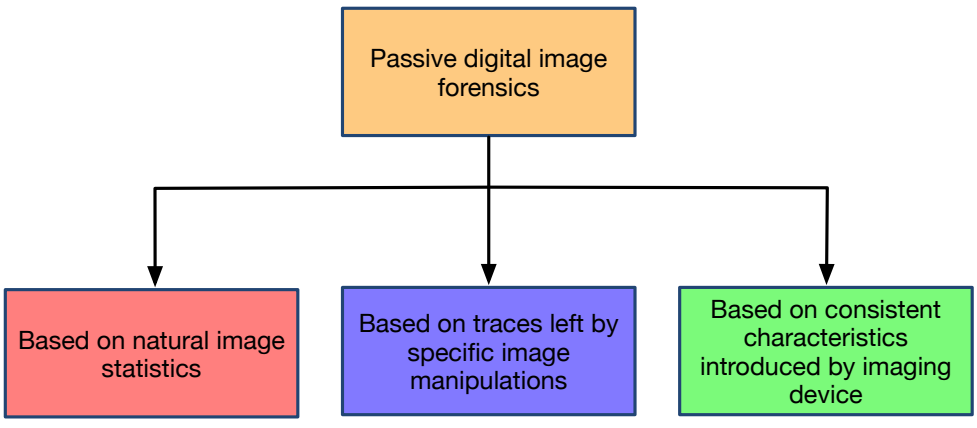



Figure 1.4: Passive image forensics categories.

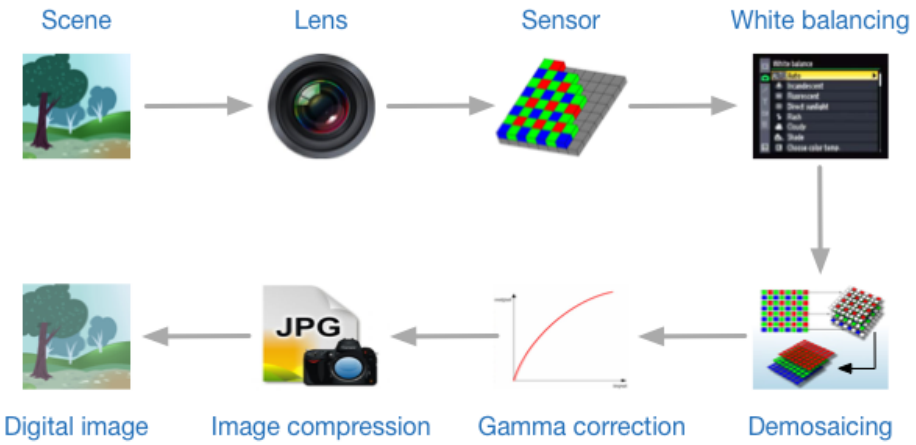



Figure 1.5: Simplified imaging pipeline in a typical digital camera.

## 1.2 Sensor Pattern Noise

As one of the most promising passive image forensic approaches, methods based on sensor pattern noise (SPN) have attracted increasing attention in recent years. As shown in Fig. 1.6, sensor pattern noise consists of two main components. One is the dark signal non-uniformity (DSNU) [4] (or dark current noise as it is more commonly referred to as), which is the pixel-to-pixel differences when the sensor array is not exposed to light. The dominant component in SPN is the photo response non-uniformity (PRNU) noise, which arises primarily from the manufacturing imperfections and the inhomogeneity of silicon wafers during the sensor manufacturing process. It is a powerful forensic tool due to its 1) uniqueness to individual device, 2) stability against environmental conditions and 3) robustness to some common image processing operations. Therefore, it can be considered as the fingerprint of imaging devices and has been widely and successfully applied in the filed of digital image forensics. In the following subsections, source camera identification, image clustering and image forgery detection based on SPN will be introduced.

[4] https://en.wikipedia.org/wiki/Fixed-pattern_noise#cite_note-1

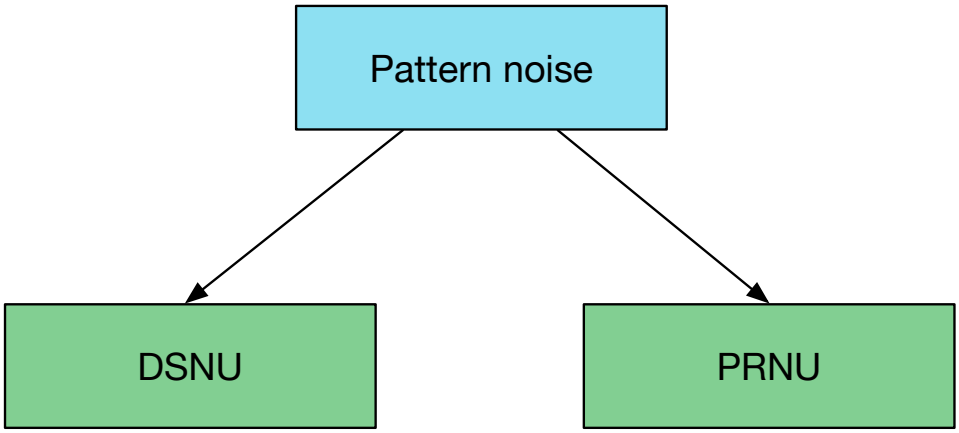
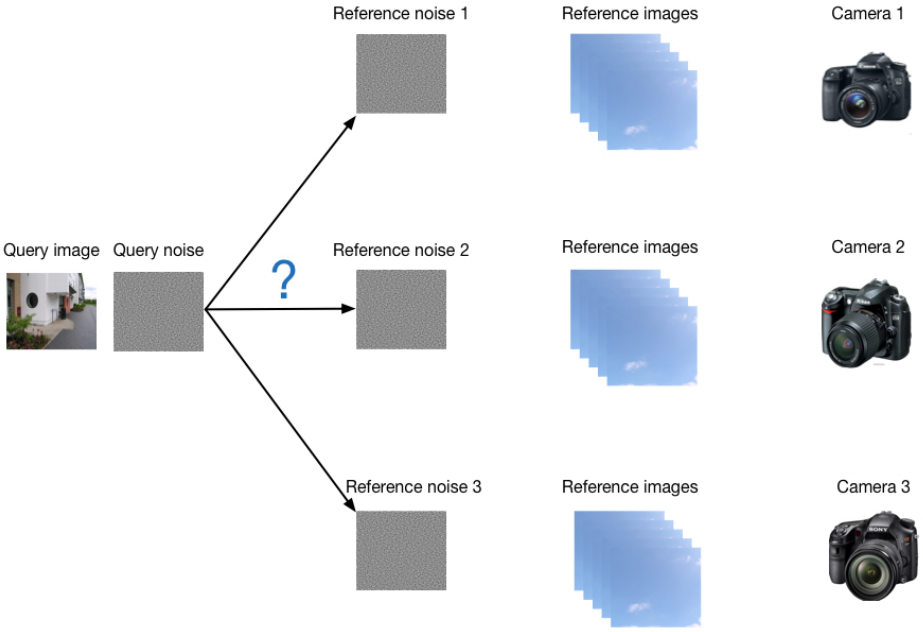


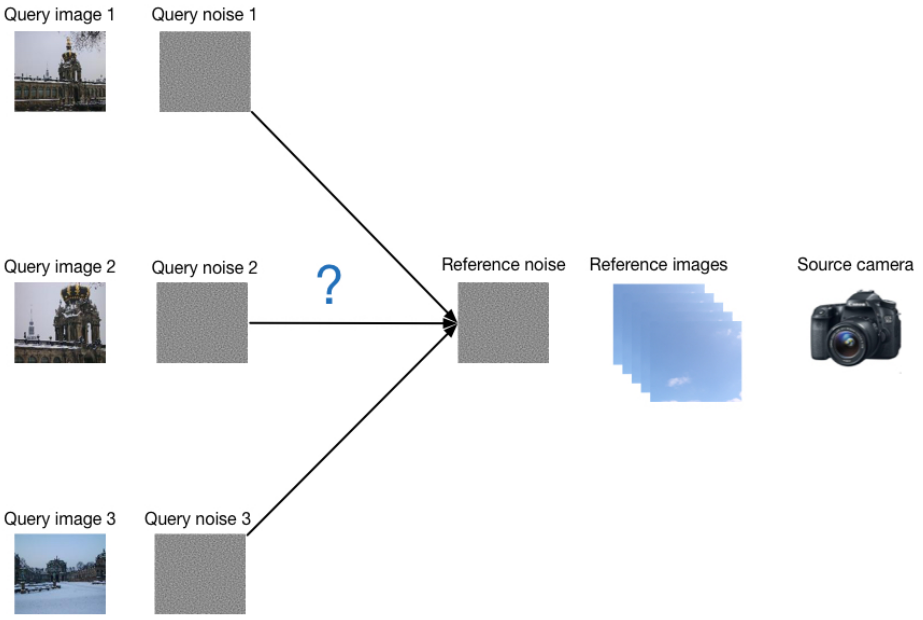

Figure 1.6: Pattern noise of imaging sensors [1].

### 1.2.1 Source Camera Identification Based on SPN

One challenging problem of multimedia forensics is source camera identification (SCI). The task of SCI is to reliably match a particular digital image with its source device. SPN is considered as the fingerprint of source camera left in every image taken by the source camera, so it can be used for tracing back to the source device of an image. Shown in Fig. 1.7(a) is the process of SPN-based source camera identification. Firstly, a reference SPN is constructed for every candidate camera by averaging the noise residues extracted from a collection of images captured by the candidate camera. The images used for constructing the reference SPN are usually those with high intensity and low texture, e.g., blue sky images, because the SPN is better preserved in them. To determine the source camera that has taken a query image among all candidate cameras, the noise residue of the query image (or query SPN) is extracted and compared with the reference SPN. If the highest similarity is higher than a predefined threshold, the query image is deemed to be taken by the camera with the highest similarity. Another forensic scenario is shown in Fig. 1.7(b), where SPN is used for identifying the image among candidate images taken by a camera. Compared to the first scenario, it is the same in essence while differing




(a)




(b)

Figure 1.7: Use of SPN in source camera identification. (a) Identifying the source camera of a given image among candidate cameras and (b) Identifying the images taken by a camera among candidate images.

in minor points. In this scenario, as long as the similarity between the query SPN and the reference SPN is high enough, the query image is considered to be captured by the camera.

### 1.2.2 Image Clustering Based on SPN

In the above application of SPN in source camera identification, a set of images taken by the same camera are required for the construction of the reference SPN. This requirement can be easily fulfilled when the camera is available to the investigators. However, in many real-world scenarios, only a set of images are available, but without any information about the source cameras. In such circumstances, sometimes it is still desirable to be able to cluster the images into a number of groups, each including the images acquired by the same camera. Take Internet child pornography for example, lots of illegal images can be confiscated from pornographic website, but the cameras which have taken these images are not available to the forensic investigators. If we can cluster these images into a number of groups, each including the images acquired by the same camera, we are able to associate different crime scenes and would be in a better position to link the evidence to the seized hardwares that are owned by the suspects in the future. As shown in Fig. 1.8, this task can be accomplished by resorting to the use of SPNs extracted from the images. By clustering the SPNs, the corresponding images taken by the same camera are grouped into the same group.

### 1.2.3 Image Forgery Detection Based on SPN

Sensor pattern noise (SPN) can be considered as a spread-spectrum watermark embedded in every image taken by the source imaging device. Therefore, it can also be used for localizing the forgeries in digital images. However, since SPN, by its very nature, is a very weak noise-like signal, its reliable detection requires jointly processing a large number of pixel samples, e.g., in a block-wise manner. As shown

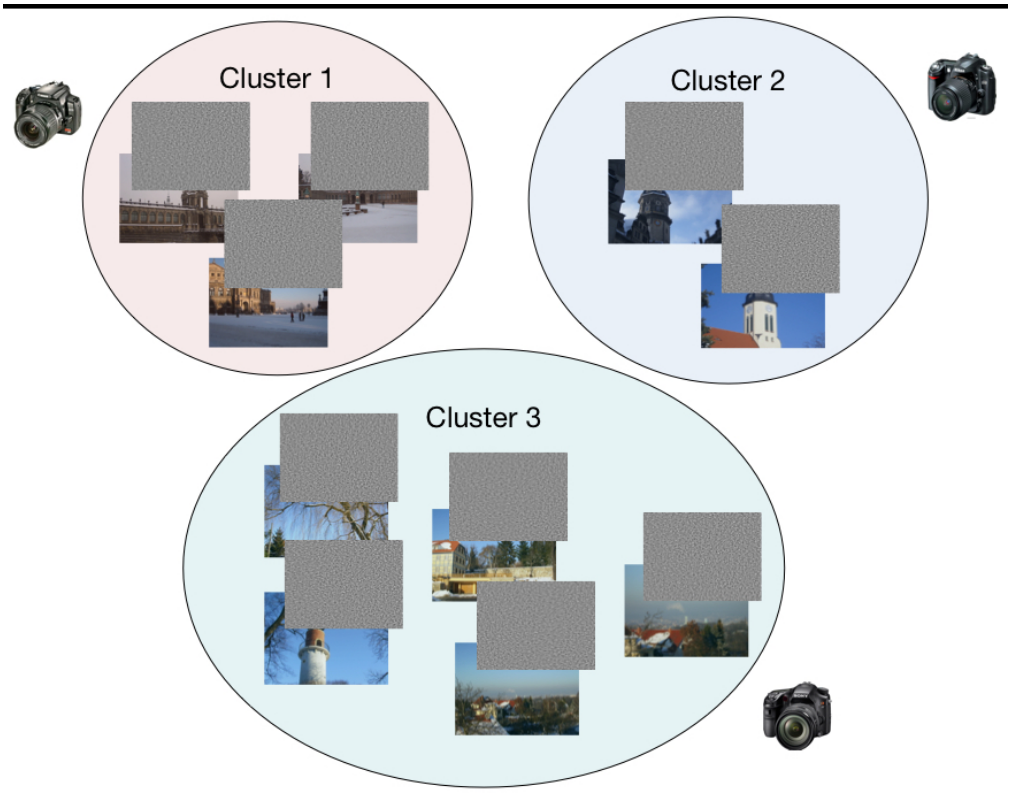



Figure 1.8: Use of SPN in image clustering.

in Fig. 1.9, the SPN extracted from the image in question is compared with the reference SPN in a block-wise manner. If their similarity (usually normalized cross correlation), which serves as a decision statistic, is below a pre-determined threshold (e.g., by Neyman-Pearson criterion), the center pixel in the block is declared as forged. This method exposes the image forgery by detecting the absence of SPN irrespective of the specific type of forgery, it therefore arouses wide attention of the researchers in the field of digital forensics.

## 1.3 Main Contributions

Digital image forensics is a broad area and includes research spanning a range of directions. In this thesis, our research on digital image forensics is narrowed down to the applications of sensor pattern noise in source camera identification, image

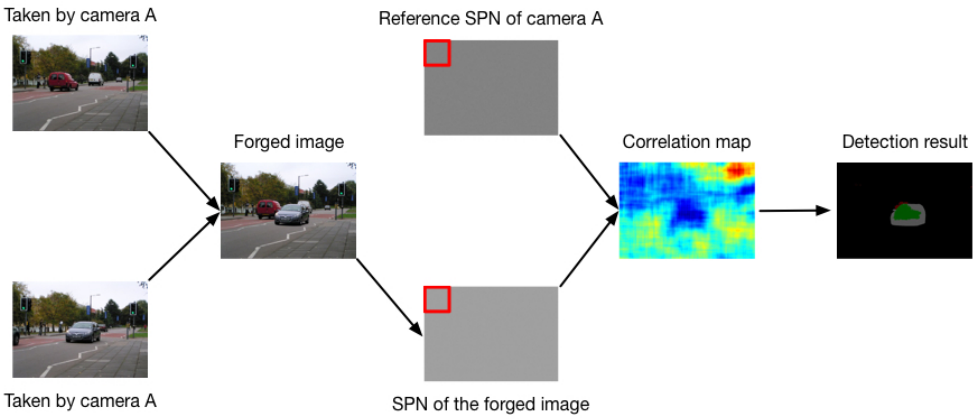



Figure 1.9: Use of SPN in digital image forgery detection.

clustering and image forgery detection. The major contributions we have made are summarized in detail as follows.

1. Although SPN has been proven to be an effective means to uniquely identify digital cameras, some non-unique artifacts, shared amongst cameras subjected to the same or similar in-camera processing procedures, often give rise to false identifications. We propose a novel preprocessing approach, i.e., spectrum equalization algorithm (SEA) in Chapter 3, that equalizes the magnitude spectrum of the reference SPN through detecting and suppressing the peaks according to the local characteristics. It aims at removing the interfering periodic artifacts and therefore reduces the false identification rate.

2. The challenges of clustering large-scale camera fingerprints come from the large-scale and high-dimensional nature of the problem. The difficulties can be further aggravated when the number of classes is much higher than the averaged class size, which is not uncommon in many practical scenarios. We propose an efficient clustering framework for large-scale device fingerprint database in Chapter 4. The framework works in an iterative way and utilizes both the dimension reduction technology and divide&conquer strategy to reduce the time and space complexity. The proposed framework can be

easily parallelized and adapted to very large-scale camera fingerprints clustering tasks.

3. As demonstrated in Fig. 1.9, the reliable detection of SPN for exposing image forgery requires the algorithm to work in block-wise manner. However, when the detection block falls near the boundary of the tampered and the non-tampered regions, the decision statistic becomes a weighted average of two different contributions and may lead to a high false acceptance rate (FAR) (i.e., misidentifying a tampered block as non-tampered). We propose a refining algorithm to alleviate the missing detection problem in Chapter 5. We model how the decision statistic changes as the detection block is placed across the boundary of two different regions (i.e., tampered and non-tampered) and adjust the decision threshold accordingly to achieve a more satisfactory detection.

## 1.4 Outline of Thesis

This chapter briefly introduces the background of digital image forensics and how sensor pattern noise can be used for identifying source camera, clustering images and exposing image forgeries. The rest of this thesis is organized as follows.

Chapter 2 first categories the interferences introduced in the imaging pipeline that affect the quality of the estimated SPN and introduces the approaches that aim at alleviating or eliminating the effect of the interferences to improve the performance of SPN based source camera identification. It then revisits the related work on image clustering based on camera fingerprint and points out the challenges of clustering large-scale image databases that have not yet been addressed by the existing methods. Literature on SPN based image forgery detection will also be reviewed in the last section of this chapter.

Chapter 3 starts with a case study to investigate the limitations of existing

SPN preprocessing schemes. It then presents the details of a novel preprocessing scheme, spectrum equalization algorithm (SEA), that overcomes the limitations of existing approaches. Combined with 6 state-of-the-art SPN extraction methods, the proposed preprocessing scheme is evaluated with comprehensive experimental results and analysis on the Dresden image database for both the general and special cases. Finally, it also gives the comparison of running times among different preprocessing schemes and similarity measurements.

Chapter 4 introduces our proposed clustering framework for large-scale camera fingerprint databases step by step, including data preparation, coarse clustering, fine clustering, centroid attraction and post processing. It also discusses the parameter selection and time complexity. Finally, it validates the capability of clustering large-scale databases and the advantage of the proposed framework over other state-of-the-art clustering algorithms with extensive experimental results.

Chapter 5 first introduces the missing detection problem of SPN based image forgery detection along the boundary between the forged and non-forged areas. It then models the correlation distribution in the problematic area and proposes to adjust the threshold accordingly to alleviate the missing detection problem. In this chapter, the effectiveness of the algorithm is verified via three realistic image forgery detection examples.

Chapter 6 summarizes this thesis. Several key challenges we are confronted with and some possible future research directions will be also given in this chapter.

# CHAPTER 2

# Literature Review

## 2.1 Source Camera Identification

Sensor Pattern Noise (SPN) has attracted much attention from researchers in the area of digital forensics since it was proven to be an effective and robust device fingerprint in [1]. One of the important applications of SPN is source camera identification (SCI), which is about identifying the source camera of a given image from candidate cameras. The typical process of using SPN for SCI is as follows. A reference SPN $\mathbf{R}$ is first constructed by averaging the noise residual $\mathbf{W}_i$ extracted from the $i$th image of the $N$ images taken by the same camera:

$$\mathbf{R} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{W}_i. \tag{2.1}$$

The similarity between the reference SPN $\mathbf{R}$ and the query noise residue $\mathbf{W}$ is measured by the normalized correlation coefficient (NCC) $\rho$:

$$\rho(\mathbf{R},\mathbf{W}) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}(W[i,j]-\overline{W})(R[i,j]-\overline{R})}{\|\mathbf{W}-\overline{W}\|\cdot\|\mathbf{R}-\overline{R}\|}, \tag{2.2}$$

where $\|\cdot\|$ is the $L_2$ norm, $\mathbf{R}$ and $\mathbf{W}$ are of the same size $m \times n$, and $\overline{R}$ and $\overline{W}$ are the arithmetic mean of $\mathbf{R}$ and $\mathbf{W}$, respectively. Suppose the reference SPN of camera $c$ is $\mathbf{R}_c$, the task of SCI is then achieved by identifying camera $c^*$ with the maximal NCC value that is greater than a predefined threshold $\tau_\rho$ as the source device of the query image, i.e.,

$$c^* = \underset{c\in\mathcal{C}}{\operatorname{argmax}}\,\rho(\mathbf{R}_c,\mathbf{W}), \rho(\mathbf{R}_{c^*},\mathbf{W}) > \tau_\rho, \tag{2.3}$$

where $\mathcal{C}$ is the set of candidate cameras.

However, the correlation-based detection of SPN heavily relies upon the quality of the extracted SPN, which can be severely contaminated by different interferences. In the following subsections, the interferences that may degrade the quality of SPN and the approaches used to alleviate or eliminate the interferences will be discussed.

### 2.1.1 Scene Details

SPN is a weak noise-like signal, so its estimation is usually based on the the noise residual $\mathbf{W}$, where the image content is significantly suppressed. The noise residual $\mathbf{W}$ of an image $\mathbf{I}$ is formulated as

$$\mathbf{W} = \mathbf{I} - F(\mathbf{I}), \tag{2.4}$$

where $F(\cdot)$ is a denoising filter. The widely used denoising filter for SPN extraction is the wavelet-based technique proposed by Mihcak et al. [2]. As described in [1], it works as follows

1. Calculate the four-level wavelet decomposition of the noisy image with the 8-tap Daubechies QMF. Denote the horizontal, vertical, and diagonal subbands as $h[i, j]$, $v[i, j]$ and $d[i, j]$, where $[i, j]$ runs through an index set $\mathbb{J}$ that depends on the decomposition level. We only show the operations for $h[i, j]$ in the following two steps, because the operations on $v[i, j]$ and $d[i, j]$ are exactly the same.

2. Estimate the local variance of the original noise-free image for each wavelet coefficient using the maximum a posteriori (MAP) estimation for four sizes of

a square $w \times w$ neighborhood $N_w$, i.e.,

$$\hat{\sigma}^2[i,j] = \min_{w \in \{3,5,7,9\}} \left[ \max \left( 0, \frac{1}{w^2} \sum_{[k,l] \in N_w} h^2[k,l] - \sigma_0^2 \right) \right], \tag{2.5}$$

where $\sigma_0^2$ is the overall variance of additive Gaussian white noise to be removed and is usually set to 4 or 9 for high-quality images.

3. The denoised wavelet coefficients are obtained suing the Wiener filter

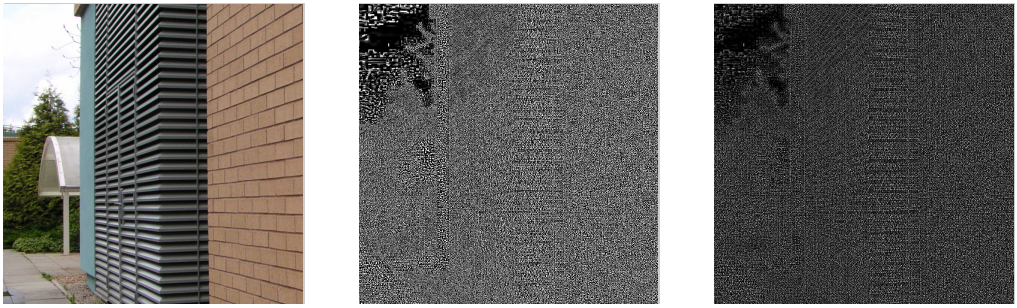
$$\hat{h}[i,j] = h[i,j] \frac{\hat{\sigma}^2[i,j]}{\hat{\sigma}^2[i,j] + \sigma_0^2}. \tag{2.6}$$

4. Repeat above 3 steps for each level and apply the inverse wavelet transform to the denoised wavelet coefficients to obtain the denoised image.

Shown in Fig. 2.1(a) and Fig. 2.1(b) are an image of natural scene and the noise residual extracted with the Mihcak denoising filter [1, 2]. As can be seen, the noise residual contains strong scene details, especially in the edge areas, because scene details account for part of the components of **I** and their magnitude is far greater than that of SPN [3].

Many efforts have been devoted to suppressing scene details in the noise residual to obtain SPN of higher quality. In [3], Li proposed 5 models to attenuate scene detail by assigning less significant weighting factors to the strong components of SPN in the Digital Wavelet Transform (DWT) domain. The underlying rationale is based on the hypothesis that *the stronger a signal component in noise residual* **W** *is, the more likely that it is associated with strong scene details, and thus the less trustworthy the component should be.* The enhanced noise residual of the image (Fig. 2.1(a)) is shown in Fig. 2.1(c), where the magnitudes of strong edges have been suppressed.

As mentioned in Section 1.2 of Chapter 1, the dominant component in SPN



(a) (b) (c)

Figure 2.1: (a) A natural image, (b) noise residual obtained with the Mihcak denoising filter [1, 2], and (c) noise residual enhanced using Model 3 with parameter 6 proposed in [3]. Note that the noise residual is extracted only from the green channel.

is the PRNU noise. So in [9], noise residual $\mathbf{W}$ is modeled as

$$\mathbf{W} = \mathbf{IK} + \mathbf{\Xi}, \tag{2.7}$$

where $\mathbf{I}$ is the observed image, $\mathbf{K}$ is the zero-mean like multiplicative factor responsible for PRNU, and $\mathbf{\Xi}$ stands for other noises including the interferences from image content, dark current noise, shot noise, and the errors introduced by the denoising filter. Equation (2.7) indicates that to obtain better estimation of SPN, the luminance $\mathbf{I}$ should be as high as possible but not saturated because saturated pixels carry no information about the PRNU factor [9]. Another observation in Equation (2.7) is that high variation in $\mathbf{\Xi}$ introduces strong interference to $\mathbf{IK}$. Therefore, SPN is better preserved in smooth areas (i.e., with lower variance). Based on the two insights that stems from Equation (2.7), some works improve the performance of source camera identification by taking into consideration pixel intensity and spatial variation. For example, a weighting scheme based on the image gradient was proposed in [46]. The weight $w(p)$ for each pixel $p$ is calculated as

$$w(p) = G(\sigma) * \frac{1}{1 + \|\nabla I_p\|}, \tag{2.8}$$

where $G(\sigma)$ is a Gaussian kernel and $\nabla$ denotes the gradient operator. While in [47], both the image intensities and image textures were considered. The authors classified the image pixels into four classes:

- Class 1: Low brightness, Low texture
- Class 2: Medium brightness, High texture
- Class 3: High brightness, Low texture
- Class 4: Saturated brightness, Medium texture.

They found that Class 3 is the best class in the sense of increasing the intra-camera similarities and decreasing the inter-camera similarities. A more sophisticated weighting scheme that formulates the relationship between the image features (related with image intensity and texture) and the correlation values was proposed in [48]. The image features are defined as

$$\begin{cases} f_{bi} = \dfrac{1}{|B_b|} \displaystyle\sum_{i \in B_b} I_p \\ f_{bt} = \dfrac{1}{|B_b|} \displaystyle\sum_{i \in B_b} \dfrac{1}{1 + \mathrm{var}_3(I_p)}, \end{cases} \tag{2.9}$$

where $B_b$ is the image block, $|B_b|$ is the cardinality of $B_b$, $I_p$ is the intensity of the image at pixel $p$, and $\mathrm{var}_3(I_p)$ is the variance of $I_p$ in a $3 \times 3$ neighborhood. The 2-dimensional features are extracted from each of the $M$ 128×128-pixel image blocks cropped from a set of training images (100 training images for each camera were used in [48]). Then the kernel principal component analysis (KPCA) [49] was used to predict the correlations $\boldsymbol{\rho}$ between the reference SPN and the SPN extracted from training blocks:

$$\hat{\boldsymbol{\rho}} = \mathbf{H}\boldsymbol{\theta}, \tag{2.10}$$

where $\mathbf{H}$ consists of the principal components of $\Phi(\mathbf{X})$, which defines the non-linear transform from the original feature space $\mathbf{X}$ to a higher dimensional space. $\mathbf{H}$ can

be considered as the projected data of $\mathbf{X}$ via KPCA, and $\theta$ can be estimated by applying the lease square estimator (LSE)

$$\boldsymbol{\theta} = \left(\mathbf{H}^{\mathrm{T}}\mathbf{H}\right)^{-1}\mathbf{H}^{\mathrm{T}}\boldsymbol{\rho}. \tag{2.11}$$

A low predicted correlation $\hat{\rho}$ means a large amount of image content is left in the noise residue. On the other hand, a large predicted correlation means that the region does not suffer from image content effect. Hence, the predicted correlation can be used to weight the center pixel of the extracted SPN [48]. While all the afore-mentioned schemes aim at weighting the query SPN, a weighted averaging scheme [50] was proposed for the estimation of the reference SPN. Specifically, the equal weighting factor $1/N$ in Equation (2.1) is replaced with

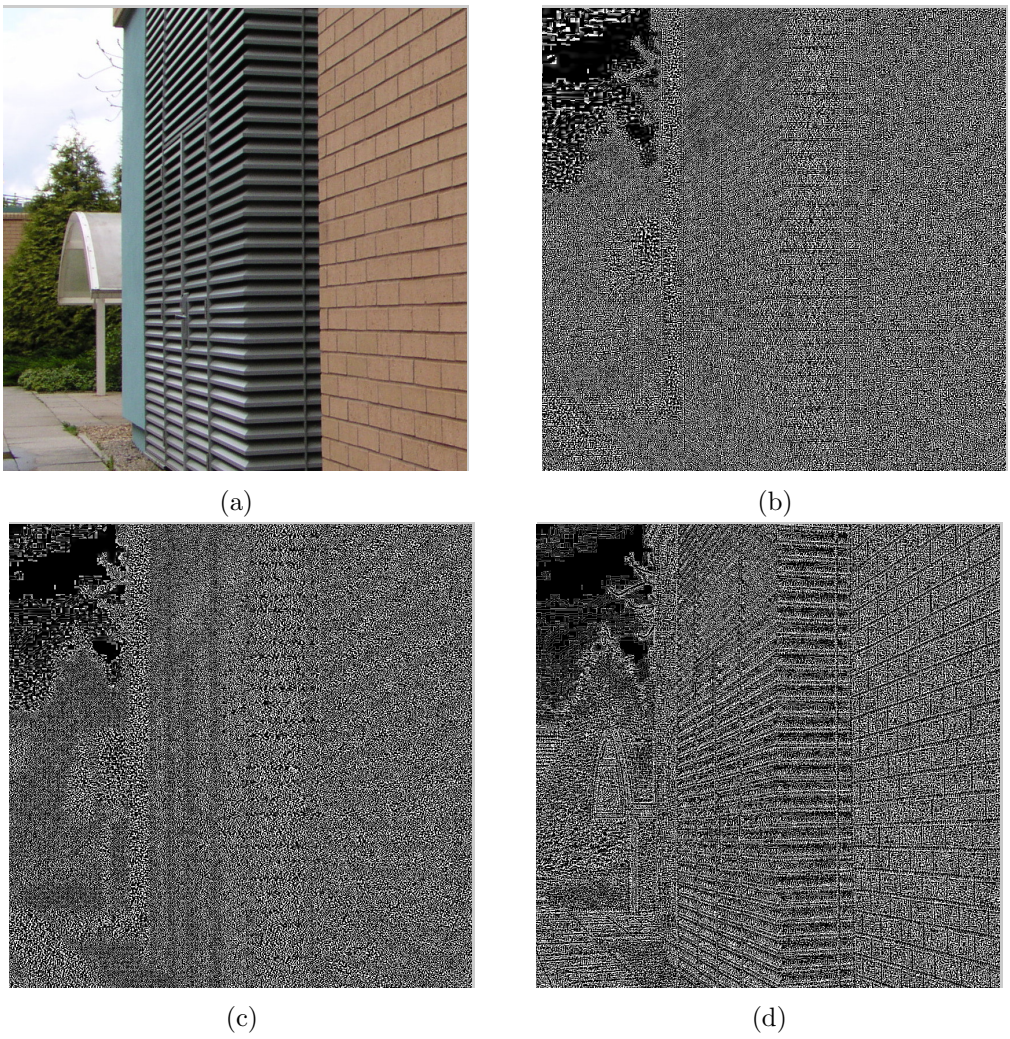
$$w_i = \frac{1}{\sigma_i^2}\Big(\sum_{m=1}^{N}\frac{1}{\sigma_m^2}\Big)^{-1}, i = 1, ..., N, \tag{2.12}$$

where $\sigma_i^2$ is the variance of undesirable noise, i.e.,

$$\begin{cases} \sigma_i^2 = \dfrac{\sum_{j=1}^{L}\left(\mathbf{n}_i[j] - \bar{\mathbf{n}}_i\right)^2}{L} \\ \mathbf{n}_i = \mathbf{W}_i - \overline{\mathbf{W}} \\ \bar{\mathbf{n}}_i = \dfrac{\sum_{j=1}^{L}\mathbf{n}_i[j]}{L} \\ \overline{\mathbf{W}} = \dfrac{\sum_{i=1}^{N}\mathbf{W}_i}{N}. \end{cases} \tag{2.13}$$

Note that $L$ is the length of samples (i.e., the image size). As reported in [50], significant performance improvements can be achieved by adopting this new weighting scheme.

Since the denoising filter $F(\cdot)$ in Equation (2.4) plays a key role in preventing the scene details from propagating to $\mathbf{W}$, a straightforward idea is to use more advanced denoising filters. Chierchia et al. used a more effective denoising filter,



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

Figure 2.2: (a) A natural image, (b) noise residual obtained with the Mihcak denoising filter [1, 2], (c) noise residual obtained with PCAI8 [4], and noise residual extracted with the BM3D denoising filter [5]. Note that we only show the noise residual extracted from the green channel, because there is no much difference for the red and blue channels.

i.e., block matching and 3D filtering (BM3D), to obtained the noise residual from images [5] BM3D works by grouping 2D image patches with similar structures into 3D arrays and collectively filtering the grouped image blocks. The sparseness of the representation due to the similarity between the grouped blocks makes it capable of better separating the true signal and noise. Compared to the noise residual obtained with the Mihcak filter [2] (Fig. 2.2(b)), most of the image content-related structures

are removed in the noise residual obtained with BM3D [5]. Wu et al. [51] proposed another SPN extractor, edge adaptive SPN predictor based on context adaptive interpolation (PCAI), to suppress the effect of scenes and edges. Suppose $I_p$ is the pixel value to be predicted at pixel $p$, then the predicted pixel value is formulated as

$$\hat{\mathbf{I}}(p) = \begin{cases} \text{mean}(\mathbf{t}), & (\max(\mathbf{t}) - \min(\mathbf{t}) \leq 20) \\ (n+s)/2, & (|e-w| - |n-s| > 20) \\ (e+w)/2, & (|n-s| - |e-w| > 20) \\ \text{median}(\mathbf{t}), & (\text{otherwise}), \end{cases} \tag{2.14}$$

where $\mathbf{t} = [n, s, e, w]^{\mathrm{T}}$ is the four-neighboring pixels of $p$, as shown in Fig. 2.3. An extension of this method is to make use of all the eight neighboring pixels $\mathbf{t}' = [n, s, e, w, en, es, wn, ws]^{\mathrm{T}}$, as proposed in [4], where $\hat{\mathbf{I}}(p)$ is formulated as

$$\hat{\mathbf{I}}(p) = \begin{cases} \text{mean}(\mathbf{t}'), & (\max(\mathbf{t}') - \min(\mathbf{t}') \leq 20) \\ (n+s)/2, & (|e-w| - |n-s| > 20) \\ (e+w)/2, & (|n-s| - |e-w| > 20) \\ (es+wn)/2, & (|en-ws| - |es-wn| > 20) \\ (en+ws)/2, & (|es-wn| - |en-ws| > 20) \\ \text{median}(\mathbf{t}'), & (\text{otherwise}). \end{cases} \tag{2.15}$$

Finally, a pixel-wise Wiener filter will be performed on the difference between the original and the predicted image, $\mathbf{D} = \mathbf{I} - \hat{\mathbf{I}}$, to obtain the noise residual

$$W[i,j] = D[i,j] \frac{\sigma_0^2}{\hat{\sigma}^2[i,j] + \sigma_0^2}, \tag{2.16}$$

where

$$\hat{\sigma}^2[i,j] = \max\Big(0, \frac{1}{w^2} \sum_{[k,l] \in N_w} D^2[k,l] - \sigma_0^2\Big). \tag{2.17}$$

| | | |
|---|---|---|
| wn | n | en |
| w | p | e |
| ws | s | es |

Figure 2.3: Neighborhood of the center pixel to be predicted. This figure is excerpted from [4].

$\sigma_0^2$ and $w$ have the same meanings as in Eq. 2.5. In both [51] and [4], the size $w$ of the local neighborhood and $\sigma_0^2$ are set to 3 and 9, respectively. Throughout this thesis, the method in [4] will be referred to as PCAI8 and used to represent this line of work. The noise residual extracted with PCAI8 is shown in Fig. 2.2(d). Surprisingly, although PCAI8 was declared to have better performance than the Mihcak filter and BM3D in [4], the scene details are more pronounced in Fig. 2.2(d).

### 2.1.2 Demosaicing

In digital cameras, the imaging sensor is probably the most expensive component. Therefore, to reduce the costs and increase the sales, most consumer digital cameras are equipped with only one sensor and an associated color filter array (CFA) to capture all necessary colors at the same time. Fig. 2.4 shows the simplified imaging pipeline, where the most widely used Bayer CFA is placed on top of a CCD sensor. For each pixel, only one color component passes through the CFA and consequently forms a monochrome image. To recover the full-color image from the mosaic-like monochrome image captured by the sensor, a demosaicing, or CFA interpolation, process needs to be devised to estimate the missing color components. However, the estimated color components are not directly acquired by physical hardware and carry interpolation noises. We expect that the PRNU extracted from the physical components, which are free from interpolation noise, should be more reliable than

that from the artificial channels, which carry interpolation noise. Therefore, it is expected that the SPN extracted from the "physical" components, which are free from interpolation noise, should be more reliable than that from the "artificial" components.

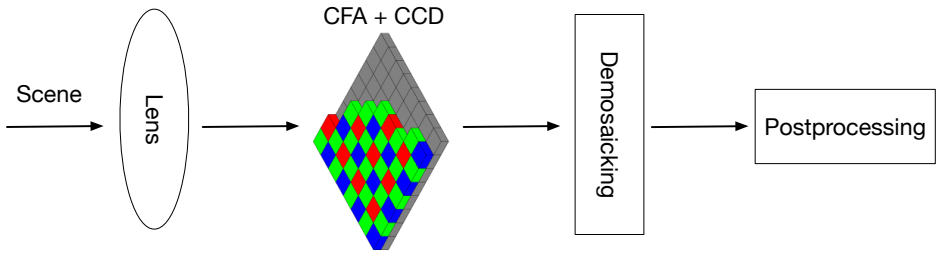



Figure 2.4: Simplified imaging pipeline of a typical digital camera.

To address this issue, Li and Li proposed a Color-Decoupled PRNU (CD-PRNU) extraction method in [6], which first decomposes each color channel into 4 sub-images and then extracts the PRNU noise from each sub-image. The PRNU noise patterns of the sub-images are then assembled to get the CD-PRNU. Fig. 2.5 shows the noise residual extraction process for the red color channel, similarly for the green and the blue channels. This method can prevent the interpolation noise from propagating into the physical components, thus improving the accuracy of device identification and image content integrity verification [6]. In [7], Hu et al. proposed to exploit the CFA structure to make better use of the information from all the 3 color channels. The process is depicted in Fig. 2.6: First, the large components in the noise residual of each color channel are selected based on their work in [52] (the first column of Fig. 2.6). Second, only the physical components (i.e., the components that are not in the interpolated positions) are kept, as shown in the second column of Fig. 2.6. Finally, the remaining components in 3 color channels are combined to acquire the camera fingerprint (the last column of Fig. 2.6). However, the CFA structure is usually unknown in practice, so they compared each component in the second column of Fig. 2.6 with its counterparts at the same

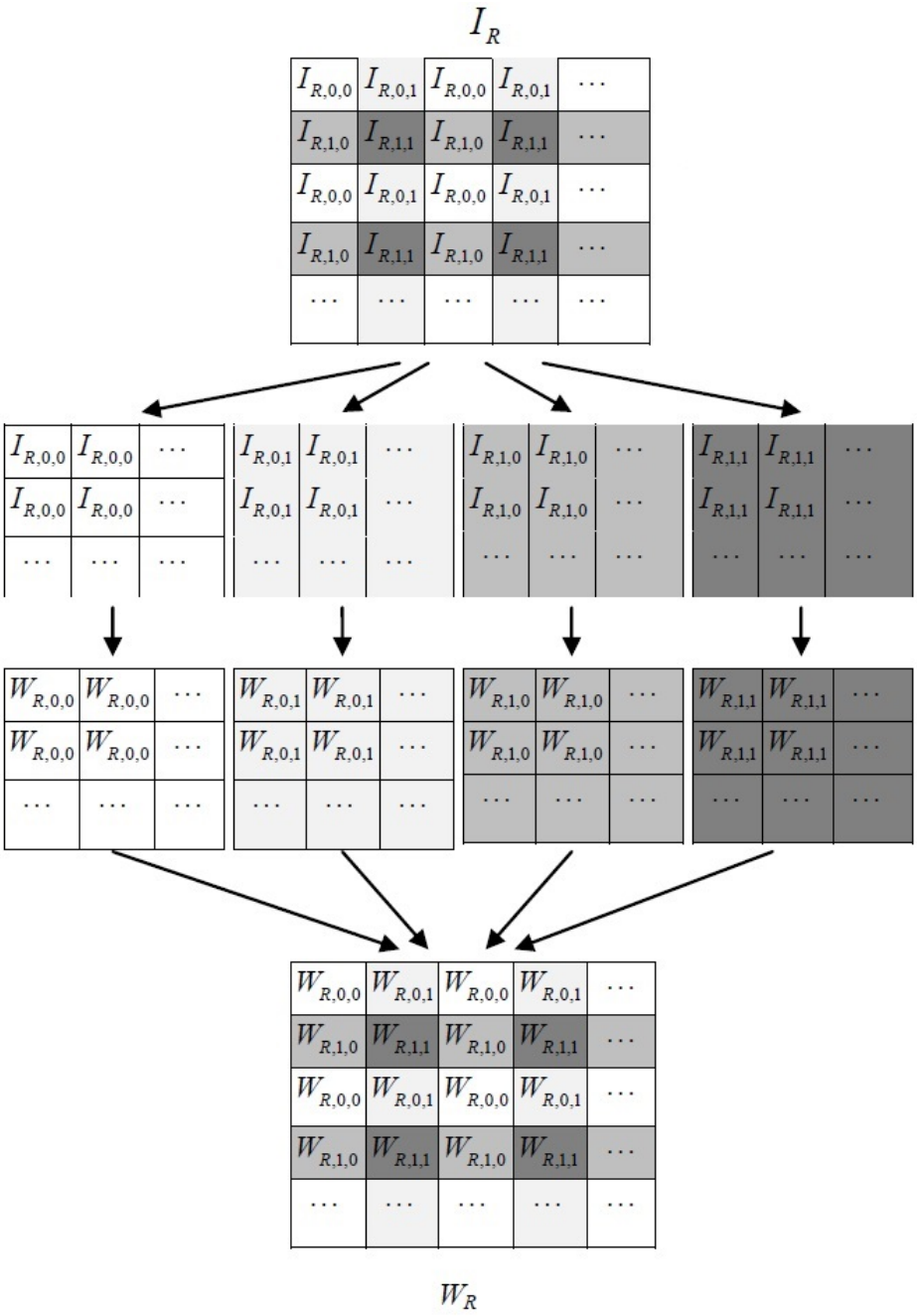



Figure 2.5: The color-decoupled noise residual extraction process for the red channel. This figure is excerpted from [6].

position on other color channels. If it is the largest component, then it is kept, otherwise it is set to 0. Taking the green channel as an example, for all pixels on

$\tilde{\mathbf{W}}_g$,

$$\tilde{\mathbf{W}}_g = \begin{cases} \tilde{\mathbf{W}}_g, & \text{if } g = \underset{x \in \{r,g,b\}}{\operatorname{argmax}} |\tilde{\mathbf{W}}_x| \\ 0, & \text{otherwise.} \end{cases} \tag{2.18}$$

In this way, the impact of demosaicing on the extracted noise residual has been reduced.

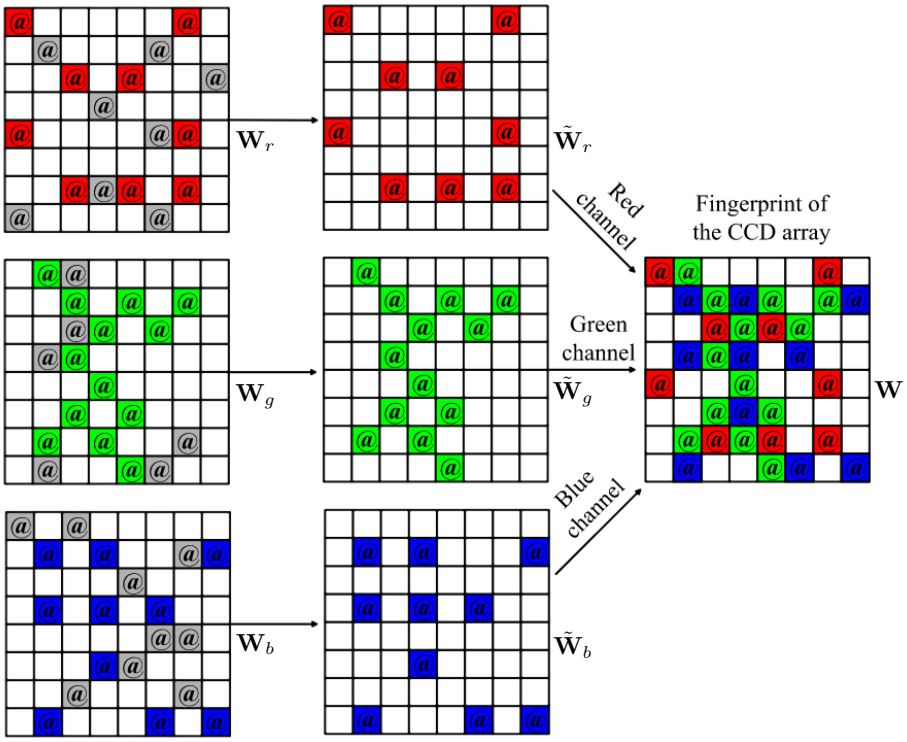



Figure 2.6: The construction of SPN using information from 3 color channels. Block with @ refers to the pixel with large magnitude. This figure is excerpted from [7].

### 2.1.3 Periodic Artifacts

For the task of source camera identification, some periodic artifacts, shared amongst cameras subjected to the same or similar periodic in-camera processing procedures, are not unique to the sensor. The SPN estimated from two different cameras may thus be slightly correlated, which would give rise to false positives and decrease the

reliability of identification system. Non-unique periodic artifacts can be summarized as follows:

- **CFA interpolation artifacts:** A typical CFA interpolation is accomplished by estimating the missing components from spatially adjacent pixels according to the component-location information indicated by a specific CFA pattern. As CFA patterns form a periodic structure, measurable offset gains will result in periodic biases in the interpolated image [9]. The periodic biases manifest themselves as peaks in the DFT spectrum, and the locations of the peaks depend on the configuration of the CFA pattern.

- **JPEG blocky artifacts:** In JPEG compression, non-overlapping $8 \times 8$-pixel blocks are coded with DCT independently. So high JPEG compression causes blocky artifacts, which manifest themselves in the DFT spectrum as peaks in the positions $(\frac{U}{8}u, \frac{V}{8}v)$, where $U$ and $V$ are the sizes of the spectrum, and $u, v \in \{0, 1, ..., 7\}$.

- **Diagonal artifacts:** As reported in [53], unexpected diagonal artifacts were observed for the reference SPN of Nikon CoolPixS710. Although the cause is yet to be investigated, the artifacts are reflected in the spectrum as peaks in the positions corresponding to the row and column period introduced by the diagonal artifacts.

Some works have been proposed to suppress these periodic artifacts. Chen et al. preprocessed the reference SPN $\mathbf{R}$ by zero-meaning (ZM) operation and Wiener filtering (WF) in the DFT domain in [9]. In the zero-meaning operation, column averages of $\mathbf{R}$ are subtracted from each pixel in each column and then row averages are subtracted from every pixel in the row to obtain the zero-meaned signal $\mathbf{R}_{\text{zm}}$. In the subsequent WF operation, $\mathbf{R}_{\text{zm}}$ is further processed in the DFT domain using a Wiener filter:

$$\mathbf{R}_{\text{wf}} = \text{Real}\Big(\mathcal{F}^{-1}\big(\mathbf{D} \cdot \frac{|\mathbf{D}| - \mathcal{W}(|\mathbf{D}|, \sigma^2)}{|\mathbf{D}|}\big)\Big), \tag{2.19}$$

where Real$(\cdot)$ returns the real part of input, $\mathcal{F}(\cdot)$ is the DFT, $\mathcal{W}(\cdot)$ is the Wiener Filter, $\mathbf{D} = \mathcal{F}(\mathbf{R}_{\text{zm}})$, $|\mathbf{D}|$ is the magnitude of $\mathbf{D}$, and $\sigma^2 = \frac{1}{MN}\sum_{i,j} R_{\text{zm}}[i,j]$. In [45], Kang et al. only kept the phase component of the noise residual and constructed a reference phase SPN to enhance the performance of SCI:

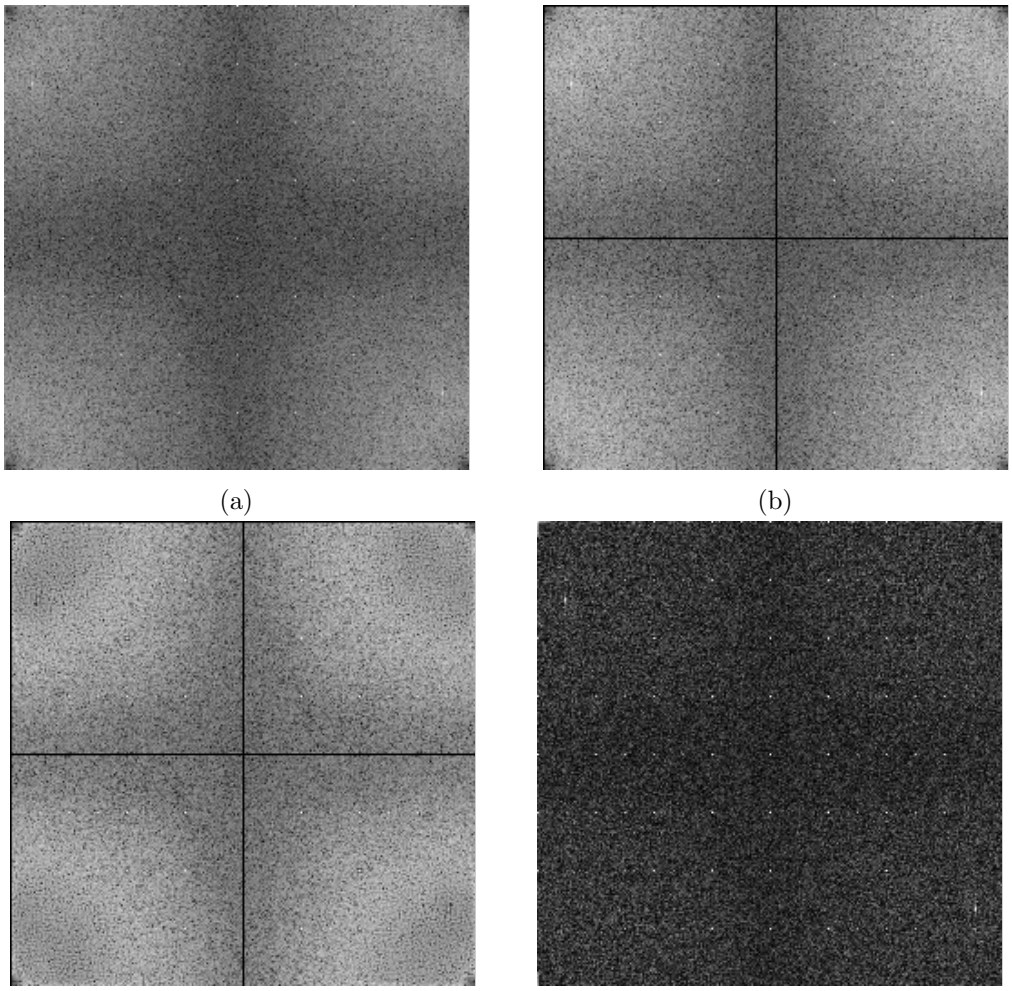
$$\mathbf{R}_{\text{ph}} = \text{Real}\Big(\mathcal{F}^{-1}\big(\frac{\sum_{i=1}^{N}\mathbf{W}_{\phi i}}{N}\big)\Big), \tag{2.20}$$

where $\mathcal{F}^{-1}(\cdot)$ is the inverse DFT and $\mathbf{W}_{\phi i}$ is phase-only component of noise residual $\mathbf{W}_i$, i.e.,

$$\mathbf{W}_{\phi i} = \frac{\mathcal{F}(\mathbf{W}_i)}{|\mathcal{F}(\mathbf{W}_i)|}, i = 1, 2, ..., N. \tag{2.21}$$

The underlying rationale is that the SPN (noise residue) is usually modeled as an additive white Gaussian noise (AWGN) in its extraction process, it is reasonable to assume the camera reference SPN to be a white noise signal with flat frequency spectrum to remove the impact of the contamination in the frequency domain [45]. Only keeping the phase components whitens the noise residual in the frequency domain and helps to remove the periodic artifacts.

The above-mentioned methods are claimed to be effective in suppressing the periodic artifacts. We conducted experiments on one Olympus Mju 1050SW, which suffers severely from the periodic artifacts, to see their performances on suppressing the unwanted periodic artifacts. 50 flat field (i.e., intensities are approximately constant) images were used for estimating the reference SPN. The size of reference SPN is $256 \times 256$ pixels and each noise residual used for reference estimation was extracted from the green channel using the Mihcak denoising filter [1]. The logarithmic magnitude spectrum of $\mathbf{R}$, $\mathbf{R}_{\text{zm}}$, $\mathbf{R}_{\text{wf}}$, and $\mathbf{R}_{\text{ph}}$ are shown in Fig. 2.7(a)-2.7(d). As can be seen, compared to $\mathbf{R}$, the DC components of $\mathbf{R}_{\text{zm}}$ are completely removed, and $\mathbf{R}_{\text{wf}}$ and $\mathbf{R}_{\text{ph}}$ are significantly whitened. However, the "white" points (peaks) in the spectrum, which are related with the periodic artifacts, are clearly shown in the processed reference SPN. This leaves space for improvement and will be discussed



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

Figure 2.7: The logarithmic magnitude spectra of (a) $\mathbf{R}$, (b) $\mathbf{R}_{\text{zm}}$, (c) $\mathbf{R}_{\text{wf}}$, and $\mathbf{R}_{\text{ph}}$.

in depth in Chapter 3.

Another set of approaches attempts to suppress the periodic artifacts through the use of more sophisticated detection statistics or similarity measurements. Goljan [54] proposed the peak-to-correlation energy (PCE) measure to attenuate the influence of periodic noise contaminations

$$\text{PCE}(\mathbf{R}, \mathbf{W}) = \frac{C^2_{\mathbf{RW}}[0,0]}{\frac{1}{MN-|\mathcal{A}|}\sum_{[k,l]\notin\mathcal{A}} C^2_{\mathbf{RW}}[k,l]}, \tag{2.22}$$

where $\mathbf{R}$ and $\mathbf{W}$ are of the same size $M \times N$, $\mathbf{C}_{\mathbf{RW}}$ is the 2D circular cross correlation

between $\mathbf{R}$ and $\mathbf{W}$, $\mathcal{A}$ is a small area around $(0, 0)$, and $|\mathcal{A}|$ is the cardinality of the area. Later in [45], Kang *et al.* proposed to use the correlation over circular cross-correlation norm (CCN) to further decrease the false-positive rate:

$$\mathrm{CCN}(\mathbf{R}, \mathbf{W}) = \frac{C_{\mathbf{RW}}[0, 0]}{\sqrt{\frac{1}{MN-|\mathcal{A}|}\sum_{[k,l]\notin\mathcal{A}} C^2_{\mathbf{RW}}[k, l]}}, \tag{2.23}$$

where all the symbols have the same meanings as in Equation (2.22). Actually CCN shares the same essence as the signed PCE (SPCE) [55, 56]

$$\mathrm{SPCE}(\mathbf{R}, \mathbf{W}) = \frac{\mathrm{sign}(C_{\mathbf{RW}}[0, 0])C^2_{\mathbf{RW}}[0, 0]}{\frac{1}{MN-|\mathcal{A}|}\sum_{[k,l]\notin\mathcal{A}} C^2_{\mathbf{RW}}[k, l]}, \tag{2.24}$$

where $\mathrm{sign}(\cdot)$ is the sign function, and all the other symbols have the same meanings as in Equation (2.22). Note that the above-mentioned approaches can be combined for additional performance gains. For instance, one can apply ZM and WF operations on the reference SPN extracted with BM3D or PCAI algorithm, and enhance the query noise residual with the help of Li's models [3], and finally choose SPCE or CCN as the similarity measurement to identify the source camera.

## 2.2 Image Clustering Based on SPN

As described in the last section, to determine the origin of a given image among candidate cameras, a reference SPN is constructed for each candidate camera by averaging the noise residues extracted from a set of images taken by the camera. The given image is deemed to be taken by the camera associated with the detected SPN in the image. In this case, a set of images taken by the same camera are required for the construction of the reference SPN. This requirement can be easily fulfilled when the camera is available to the investigators. However, in many real-world scenarios, only a set of images are available, but without any information about the source cameras. Taking child pornography as an example, illegal images

are confiscated from pornographic websites, but the cameras which have been used to take these images are not available to the forensic investigators. In some forensic circumstances, it is necessary to cluster the images into a number of groups, each including the images acquired by the same camera, so that the forensic investigators are able to associate different crime scenes and would be in a better position to link the evidence to the seized hardwares that are owned by the suspects. This task can be accomplished by clustering the images based on the SPNs extracted from the images. While several methods have been proposed for clustering the camera fingerprints, their applicability is restricted to small datasets mainly due to the large-scale and high-dimensional nature of the problem.

### 2.2.1 Markov Random Field Based Method

One of the first works dedicated to clustering camera fingerprints into a number of classes was reported in [57], where each enhanced fingerprint is treated as a random variable and Markov random field (MRF) is used to iteratively update the class label of each fingerprint. A subset of images are randomly chosen from the entire dataset to set up a training set and a pairwise similarity matrix is calculated for the training set, based on which a reference similarity is determined using the $k$-means ($k = 2$) clustering algorithm and a membership committee consisting of a certain number of the most similar fingerprints is formed for each fingerprint. The similarity values and class labels within the membership committee are used to estimate the likelihood probability of assigning each class label to the corresponding fingerprint. Then the class label with the highest probability in its membership committee is assigned to the fingerprint in question. The clustering stops when there are no label changes after two consecutive iterations. Finally, each fingerprint in the rest of the dataset is classified to its closest cluster. This algorithm performs well on small datasets, but it is very slow due to the calculation of the likelihood probability, which involves all the members in the membership committee and needs

to be calculated for every class label and every fingerprint. The time complexity is nearly $\mathcal{O}(n^3)$ in the first iteration, where $n$ is the number fingerprints, so it becomes obviously computationally prohibitive for large-scale datasets. Another limitation is that when there are many classes in the dataset, the size of the training set has to be large enough to ensure that all of the classes are present in the training set. These two limitations make it computationally infeasible even for medium-sized datasets.

### 2.2.2 Graph Clustering Based Method

In [58], camera fingerprints clustering is formulated as a weighted undirected graph partitioning problem. Each fingerprint is considered as a vertex in the graph, while the weight of each edge is represented by the similarity between the two fingerprints linked by the edge. To avoid the time-consuming pairwise similarity calculation, a sparse graph is constructed instead. A vertex is randomly selected as the initial center of the graph and the weights of its edges with all other vertices are calculated. The ($\kappa$+1)th closest vertex to the first center is then selected as the second center and its edge weights with all other vertices except the first center are calculated, where $\kappa$ is a parameter controlling the sparsity of the graph. The construction procedure stops when the number of vertices that have not been considered as a center is not larger than $\kappa$. A multi-class spectral clustering algorithm [59] is then employed on the constructed graph to partition the vertices (fingerprints) into a number of clusters. For every vertex being investigated, its similarities with all the other vertices have to be calculated when constructing the sparse graph. It is fine if all fingerprints can be fit into the memory at a time, but when it comes to large-scale datasets, the algorithm becomes inefficient due to the high I/O overhead because one fingerprint may need to be read from the disk many times for calculating its similarities with the centers. The time complexity of the algorithm in [59] is $\mathcal{O}(n^{\frac{3}{2}}m + nm^2)$, where $m$ is the number of partitions. It is thus faster than Li's algorithm [57]. However, to determine the optimal number of partitions, the algorithm works in an iterative

manner until the size of the smallest cluster equals 1. Putting aside the speed, the aptness of the manner of finding the optimal partitions number is still an issue for large-scale camera fingerprint datasets.

### 2.2.3 Hierarchical Clustering Based Method

Another algorithm based on the hierarchical clustering was proposed in [60]. Similar to [57], the fingerprint is enhanced beforehand and only a random subset (training set) of the whole dataset is used for clustering, followed by a classification stage for the remaining fingerprints. Initially considering each fingerprint as one cluster, the algorithm first calculates the pairwise similarity matrix of the training set. The two most similar clusters are merged into one and the similarity matrix is updated by replacing the corresponding two rows and columns with the similarities between the merged cluster and all other clusters. After the update, a silhouette coefficient, which measures the separation among clusters and the cohesion within each cluster, is calculated for each fingerprint. A global measure of the silhouette coefficients is calculated by taking the average value of the silhouette coefficients, and stored with the current partition. When all fingerprints have been merged into a cluster, the partition corresponding to the lowest silhouette coefficient is the optimal clustering. In the end, the classification stage is the same as what has been described in [57]. Another similar hierarchical clustering based algorithm was proposed in [61], where the only difference is that the calculation of the silhouette coefficient is performed for each cluster rather than for each fingerprint and only the separation to the nearest neighboring cluster is measured. As reported in [60], with comparable accuracy, the hierarchical clustering based algorithm is faster than [57]. But the computational cost of the hierarchical clustering is still very high and requires at least $\mathcal{O}(n^2 \log n)$ operations. This therefore limits its applicability to large-scale databases.

### 2.2.4 Other Clustering Methods

There are many other algorithms that have been proposed for clustering various types of data. However, the challenges of clustering large-scale camera fingerprints come from the large-scale and high-dimensional nature of the problem. The difficulties can be further aggravated when the Number of Classes (i.e., number of cameras) is much higher than the average Size of Class (i.e., number of images acquired by each camera). We refer to this as the $NC \gg SC$ problem, which is not uncommon in many practical scenarios. The $NC \gg SC$ problem makes it difficult, if not impossible, to form a training set at random that can sufficiently represent the entire population. The characteristics of the camera fingerprints clustering problem also make it difficult to employ most of the classic clustering algorithms. For example, the partition clustering algorithms, as typified by $K$-means [62] and CLARANS [63], require users to provide the desired partition number $K$, the determination of which can be tricky in practice. Moreover, they may require several passes over the database and thus do not scale well to large-scale camera fingerprint databases. The density based approaches, such as DBSCAN [64], are directly performed on the entire database. As a result, for large databases that cannot fit in the main memory, it could incur substantial I/O cost [65]. Furthermore, its sensitiveness to parameters and its inability to handle clusters with various densities make it hard to produce satisfactory results on camera fingerprints, whose noise-like characteristics can easily result in clusters with various densities. Some hierarchical clustering algorithms using random sampling to reduce the input size for large databases, such as [65] and [66], will suffer from the $NC \gg SC$ problem. Other hierarchical clustering algorithms designed for large-scale databases, as typified by BIRCH [67] and CHAMELEON [68], do not perform well on camera fingerprint databases because of either the sensitivity to outliers or the high I/O cost when constructing the $\kappa$-nearest graph.

## 2.3 Forgery Detection

Detecting image forgeries is an interesting while very challenging task due to the variety of image manipulations a user can perform with increasingly powerful image editing software. Active techniques, such as digital watermarking, are effective in verifying the content of an image, but the requirement of originally embedding into the protected image limits their widespread use in practice. Therefore, there has been growing interest in passive techniques for image forgery detection. Many passive image forgery detection techniques are designed for detecting one specific form of image altering operations, such as contrast enhancement [31, 33, 34, 69] and copy-move forgery [35, 70, 71], however, in most scenarios, the forensic investigators do not have any prior information of the suspicious images and have to apply a large set of forensic methods on the suspect images before making the final decisions. Sensor pattern noise (SPN) can be considered as an intrinsic watermark embedded in every image taken by the source device and therefore appears as a promising tool for detecting image forgeries. When the camera reference SPN is available, image forgeries can be exposed by detecting the absence of the SPN in targeted regions irrespective of the specific type of forgery. Another merit of SPN is that it is robust to some common image processing operations, such as JPEG compression, filtering, or gamma correction [1, 9]. These two merits make SPN very attractive in many practical applications.

### 2.3.1 Preliminary Method

To the best of our knowledge, the first work that uses SPN for image forgery detection was proposed in [8], where two different approaches were devised for verifying the integrity of a selected Region of Interest (ROI) and for automatically identifying forged areas. Specifically, the first approach aims to verify the integrity of a selected area $\Omega$ in an image $\mathbf{I}$. Firstly, a reference SPN $\mathbf{R}$ is constructed as in Equation

(2.1) for the source camera $C$ that has taken $\mathbf{I}$. To calculate the statistical evidence that $\Omega$ has been tampered with, a large set of image regions $\mathbf{Q}_k, k = 1, ..., N$ of the same size and shape is collected either from the images taken by the same camera $C$ (but a different location within the images), or from the images taken by other cameras. These image regions can be considered as "tampered" regions, and the correlations $\rho(\mathbf{R}_\Omega, \mathbf{W}_k), k = 1, ..., N$ are used to estimate a generalized Gaussian distribution, where $\mathbf{R}_\Omega$ is the reference SPN in the position of $\Omega$ and $\mathbf{W}_k$ is the SPN (i.e., noise residual) extracted from $\mathbf{Q}_k$. Using the correlation distribution estimated from "tampered" regions, the probability that a generalized Gaussian random variable will attain the value $\rho(\mathbf{R}_\Omega, \mathbf{W}_\Omega)$ or larger is

$$p = 1 - G(\rho(\mathbf{R}_\Omega, \mathbf{W}_\Omega)), \tag{2.25}$$

where $\mathbf{W}_\Omega$ is the SPN in $\Omega$ and $G(\cdot)$ is the cumulative distribution function of the estimated generalized Gaussian distribution. $\Omega$ has been forged if $p > \alpha = 10^{-3}$, and not forged otherwise.

The second approach is capable of automatically identifying the forged area. To detect forgeries of different shapes, twelve sliding blocks of different shapes and sizes are prepared, as illustrated in Fig. 2.8. These blocks will be moved across the entire image and the correlation at each location will be calculated for decision-making. The details of the algorithm are given as follows [8]:

1. For each block type $i \in \{1, ..., N\}$ illustrated in Fig. 2.8, compute the correlation of the noise residual with the reference SPN in the sliding block moving across the entire image (overlapping approximately $50\% - 75\%$). This will generate a number of correlations $\rho_j, j = 1, ..., n_i$, where $n_i$ is the number of correlations calculated for each block type $i$.

2. For each block type $i$, select $m$ blocks $\{i_1, ..., i_m\}$ with the smallest correlations $\rho_k, k = 1, ..., m$. There will be a total of $m \times N$ blocks $\mathfrak{B}_k$.

3. Construct the mask $\mathfrak{B} = \bigcup_{k=1}^{m \times N} \mathfrak{B}_k$.

4. For each pixel $q \in \mathfrak{B}$, let $t(q) = |\{\mathfrak{B}_k | q \in \mathfrak{B}_k\}|$ be the number of blocks selected in Step 2 covering $q$.

5. The potentially forged region is $\mathfrak{R} = \{q | t(q) > t\}$, where $t$ is the median value of $t(q)$ for $q \in \mathfrak{B}$.

The detected area output by the second approach can be further verified as in the first approach for detection forgeries in user-selected ROIs.

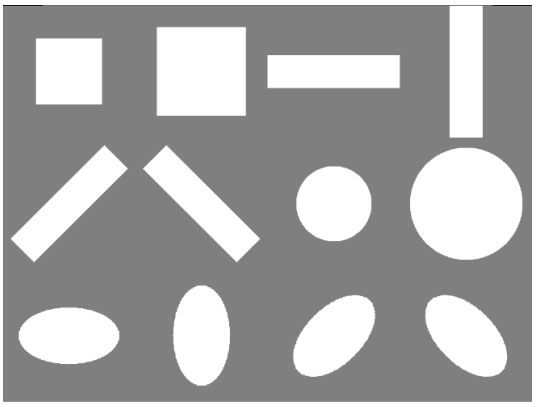


Figure 2.8: Sliding block shapes used for automatic ROI detection. This figure is excerpted from [8].

### 2.3.2 Constant False Acceptance Rate Method

The work in [8] was formally modeled and improved by Chen et al. in [9]. They modeled the SPN detection problem as a binary hypothesis testing problem:

$$\begin{cases} H_0 : \mathbf{W} = \mathbf{\Xi}, \\ H_1 : \mathbf{W} = \mathbf{R} + \mathbf{\Xi}, \end{cases} \tag{2.26}$$

where $\mathbf{W}$ is the noise residual extracted from the image region in question, $\mathbf{R}$ is the reference SPN (camera fingerprint), and $\mathbf{\Xi}$ is the combination of other independent interferences. The forgery detection at each pixel is therefore formulated as

a hypothesis testing problem applied to a block surrounding the pixel. But before doing that both the correlation distribution under hypothesis $H_0$, $p(x|H_0)$, and the correlation distribution under hypothesis $H_1$, $p(x|H_1)$ need to be estimated.

$p(x|H_0)$ can be estimated using the method described in Section 2.3.1, while the estimation of $p(x|H_1)$ is difficult, because the correlation heavily depends on image content and is most likely to be over-fitting to the available images. Therefore, instead of explicitly estimating $p(x|H_1)$, a correlation predictor is constructed as a mapping from the image feature space to the predicted correlation value. $K$ image blocks of $128 \times 128$ pixels are cropped from several images taken by the source camera. These blocks may be overlapped with each other, so that one can extract sufficient image blocks from one image. Four image features are extracted from each image block: Image intensity feature $f_I$, texture feature $f_T$, signal flattening feature $f_S$, and texture-intensity feature $f_E$. Let $\boldsymbol{\rho}$ be the correlations between the noise residuals of the $K$ image blocks and the reference SPN in the corresponding positions, and $\boldsymbol{f}_I$, $\boldsymbol{f}_T$, $\boldsymbol{f}_S$, and $\boldsymbol{f}_E$ be the corresponding $K$-dimensional feature vectors. $\boldsymbol{\rho}$ is modeled as a linear combination of the features and their second-order terms

$$\begin{aligned}\rho[k] = \theta_0 + \theta_1 f_I[k] + \theta_2 f_T[k] + \theta_3 f_S[k] + \theta_4 f_E[k] + \\ \theta_5 f_I[k] f_I[k] + ... + \theta_{14} f_E[k] f_E[k] + \Psi[k],\end{aligned} \tag{2.27}$$

where $\Psi[k]$ is the modeling noise and $\boldsymbol{\theta}$ is the coefficients to be determined. Equation (2.27) can be rewritten into a matrix form $\boldsymbol{\theta} = \mathbf{H}\boldsymbol{\theta} + \boldsymbol{\Psi}$, where $\mathbf{H}$ is a $K \times 15$ matrix of features and their multiplications. By applying the least square estimator (LSE), we can obtain the estimated parameters

$$\hat{\boldsymbol{\theta}} = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\boldsymbol{\rho}. \tag{2.28}$$

Given an image block, its expected correlation $\hat{\rho}$ can be predicted based on the

image features extracted from it:

$$\hat{\rho} = [1, f_I, f_T, f_S, f_E, ..., f_E f_E]\hat{\boldsymbol{\theta}}. \tag{2.29}$$

With the correlation predictor, $p(x|H_1)$ is modeled as the generalized Gaussian distribution $GG(\hat{\rho}, \sigma_1, \alpha_1)$, where the predicted correlation $\hat{\rho}$ is the mean, while the scale parameter $\sigma_1$ and the shape parameter $\alpha_1$ can be estimated from the difference between the real and predicted correlations, i.e., $\boldsymbol{\rho} - \hat{\boldsymbol{\rho}}$.

To detect forgeries in an image, the algorithm proceeds by sliding a 128×128-pixel detection block across the image and evaluates the test statistic $\rho_i = \rho(\mathbf{R}_i, \mathbf{W}_i)$ for each block, where $\mathbf{W}_i$ and $\mathbf{R}_i$ are the noise residual and the reference SPN in the block centered at pixel $q_i$. If $\rho_i < t$, pixel $q_i$ is deemed as forged. Here the threshold $t$ is related with a constant false acceptance rate (CFAR) $10^{-5}$:

$$\int_t^{\infty} p(x|H_0)\mathrm{d}x = 10^{-5}. \tag{2.30}$$

That is why this method is usually referred to as the constant false acceptance rate (CFAR) method. However, for a highly textured or saturated block, even it is not forged, its correlation still tends to be low due to the absence of SPN. So to avoid labeling non-tampered pixels as tampered, pixel $q_i$ will be labeled as non-tampered if

$$\int_{-\infty}^{t} p(x|H_1)\mathrm{d}x > \beta, \tag{2.31}$$

where $\beta$ was set to 0.01 in [9]. The resulting binary map $\hat{\boldsymbol{u}} \in \{0, 1\}^{M \times N}$, which indicates the forged pixels (1 for forgery and 0 for genuine pixel), is further dilated with a square $20 \times 20$ kernel to obtain the final result.

### 2.3.3 Bayesian-MRF Based Method

The constant false acceptance rate (CFAR) method makes decisions independently for each pixel, but it does not take into account the spatial dependencies exhibited by natural images, which could generate fragmented and inconsistent binary map. To penalize the isolated points or the small disjoint regions and provide smooth output, Chierchia et al. adopted the Bayesian rule and Markov random field (MRF) model to improve the detection results [72]. This Bayesian-MRF method is based on the CFAR method but differs from it in both the formulation and the solution of the problem. The forgery detection is formulated as a problem of finding the label map $\hat{\mathbf{u}} \in \{0,1\}^{M\times N}$ that has the maximum probability to occur given the known information:

$$\hat{\mathbf{u}} = \underset{\mathbf{u}\in\{0,1\}^{M\times N}}{\operatorname{argmax}} p(\boldsymbol{\rho}|\mathbf{u},\hat{\boldsymbol{\rho}})p(\mathbf{u}), \tag{2.32}$$

where $M\times N$ is the image size, $\boldsymbol{\rho}$ is the actual correlations calculated in a block-wise manner, and $\hat{\boldsymbol{\rho}}$ is the predicted (or expected) correlations given by the correlation predictor described in last section. In the above equation, $p(\boldsymbol{\rho}|\mathbf{u},\hat{\boldsymbol{\rho}})$ is the conditional likelihood of observing $\boldsymbol{\rho}$, and $p(\mathbf{u})$ is the prior probability that takes into account the spatial dependencies of the pixels by resorting to the Markov random field model:

$$p(\mathbf{u}) = \frac{1}{Z} e^{-\sum_{c\in\mathcal{C}} V_c(\mathbf{u})}, \tag{2.33}$$

where $Z$ is a normalizing constant, and $V_c(\mathbf{u})$ is the potential defined on cliques $c$ (i.e., small groups of neighboring pixels). Only the single-site cliques $\{c'\}$ and 4-connected two-site cliques $\{c''\}$ are considered:

$$V_{c'}(u_i) = \begin{cases} \dfrac{-\alpha}{2} & \text{if } u_i = 0 \\ \dfrac{\alpha}{2} & \text{if } u_i = 1 \end{cases} \tag{2.34}$$

$$V_{c''}(u_i, u_j) = \begin{cases} \beta & \text{if } u_i \neq u_j \\ 0 & \text{otherwise} \end{cases} \tag{2.35}$$

where $\alpha = \log(p_0/p_1)$ is related to the prior probability of forged $p_0$ and non-forged $p_1$, and $\beta$ is the edge-penalty parameter indicating how strong the interaction between pixels is. After replacing $p(\mathbf{u})$ with Equation (2.33) and taking the negative log, Equation (2.32) can be rewritten as

$$\hat{\mathbf{u}} = \underset{\mathbf{u}\in\{0,1\}^{M\times N}}{\operatorname{argmin}} \left\{ -\sum_{i=1}^{M\times N} \log p(\rho_i|u_i, \hat{\rho}_i) + \alpha \sum_{i=1}^{M\times N} u_i + \beta R(\mathbf{u}) \right\}, \tag{2.36}$$

where the regularization term $R(\mathbf{u}) = \sum_{i=1}^{M\times N} \sum_{j\in\mathcal{N}_i} |u_j - u_i|$, with $\mathcal{N}_i$ the set of 4-connected neighbors of $i$, is the sum of all class transitions over all 4-connected cliques of the image. By assuming the likelihood probability to be Gaussian under both hypotheses, with zero mean and variance $\sigma_0^2$ under hypothesis $H_0$, and mean $\hat{\rho}_i$ and variance $\sigma_1^2$ under hypothesis $H_1$, Equation (2.36) becomes

$$\hat{\mathbf{u}} = \underset{\mathbf{u}\in\{0,1\}^{M\times N}}{\operatorname{argmin}} \left\{ \sum_{i=1}^{M\times N} u_i \left[ \frac{(\rho_i - \hat{\rho}_i)^2}{2\sigma_1^2} - \frac{\rho_i^2}{2\sigma_0^2} - \log\frac{\sigma_0}{\sigma_1} - \log\frac{p_1}{p_0} \right] + \beta R(\mathbf{u}) \right\}. \tag{2.37}$$

By resorting to the convex-optimization algorithm proposed in [73], the $\hat{\boldsymbol{u}}$ that gives the maximal probability can be obtained. This method incorporates the prior information and spatial dependencies between pixels, and therefore produces a more consistent and smooth binary map.

### 2.3.4 Image Segmentation Based Methods

Despite the good results on detecting large tampered regions (e.g., larger than $128 \times 128$ pixels) given by the above CFAR and Bayesian-MRF method, there is still a need to improve the spatial resolution, allowing for detecting smaller forgeries. As can be observed in the detection results presented in [9], the falsely identified areas

are generally located around the boundary of the forged area. The reason is that, when the detection block falls near the boundary between two different regions (i.e., forged and non-forged regions), the test statistic $\rho$ is a weighted average of two different contributions and more likely to exceed the decision threshold [74]. As a result, miss detection occurs along the boundary. In [74], this problem is solved by first segmenting the image under investigation. However, this method heavily depends on the performance of image segmentation, which itself is a very challenging and unreliable process. In [75], an algorithm based on the guided filtering [76] was proposed to avoid the ill-posed image segmentation problem. The basic idea is to post-process the calculated correlation map $\boldsymbol{\rho}$ by resorting to a pilot image, which can be a combination of the color bands of the original image or its denoised version, or any suitable field of features extracted from images [75]. The pilot image bears some valuable information, such as geometrical structures, of the image content and can be viewed as the soft-segmented version of the original image. By incorporating the structure information of the image, the guided filtering is helpful in making a better decision. However, either the hard segmentation based method [74] or the soft segmentation based method [76] will fail on the so called *occlusive* forgeries, where part of the background is copied and pasted to hide objects of the original scene, e.g., an airplane is covered by a patch of blue sky. Actually, as pointed out in [74], the use of segmentation-based decision is at great risks in such scenarios, because the forged region would be a relatively small part of a large region and the statistics calculated on the overall large region would become irrelevant in the average.

## 2.4 Summary

In this chapter, we first introduced the typical process of SPN-based source camera identification and reviewed the approaches that aim at improving the performance

from the perspective of suppressing interfering sources, namely scene details, demosaicing artifacts and periodic artifacts. We then discussed the image clusterings algorithms based on camera fingerprints (SPNs) and the difficulties of directly applying other classic clustering methods on large-scale camera fingerprint databases. Finally, we revisited the SPN-based image forgery detection algorithms in detail. After introducing the SPN-based algorithms for each of the three image forensic tasks (i.e., source camera identification, image clustering, and image forgery detection), we also pointed out the limitations of existing methods. These limitations will be addressed in the following three chapters, each corresponding to one of the three image forensic tasks.

# CHAPTER 3

## Spectrum Equalization Algorithm for Preprocessing Reference Sensor Pattern Noise

As mentioned in Section 2.1 of Chapter 3, although pattern noise (SPN) has been proven to be an effective means to uniquely identify digital cameras, some non-unique artifacts, shared amongst cameras subjected to the same or similar in-camera processing procedures, often give rise to false identifications. Therefore, it is desirable and necessary to suppress these unwanted artifacts so as to improve the accuracy and reliability.

In this chapter, we propose a novel preprocessing approach for attenuating the influence of the non-unique artifacts on the reference SPN to reduce the false identification rate. Specifically, we equalize the magnitude spectrum of the reference SPN through detecting and suppressing the peaks according to the local characteristics, aiming at removing the interfering periodic artifacts. Combined with six SPN extraction or enhancement methods, our proposed Spectrum Equalization Algorithm (SEA) is evaluated on the Dresden image database as well as our own database, and compared with the state-of-the-art preprocessing schemes. Experimental results indicate that the proposed procedure outperforms, or at least performs comparably to the existing methods in terms of the overall ROC curve and kappa statistic computed from a confusion matrix, and tends to be more resistant to JPEG compression for medium and small image blocks.

The reminder of this chapter is organized as follows. The next section gives a brief overview of the background. In Section 3.2, we revisit the previous works through a case study and point out the limitations of existing preprocessing approaches. The details of the proposed preprocessing scheme, SEA, are presented in

Section 3.3. Comprehensive experimental results and analysis for both the general and special cases are given in Section 3.4. Finally, Section 3.5 concludes the chapter.

## 3.1 Introduction

One challenging problem of multimedia forensics is source camera identification (SCI), the task of which is to reliably match a particular digital image with its source device. Despite the methods based on metadata, or watermarking embedded in the image, are effective in proving the source of an image, unfortunately they are infeasible under many circumstances. For example, the metadata might not be available, and legacy images might not be watermarked at the time when they were created. In view of the limitations, researchers have switched their attention to the methods that search for the intrinsic characteristics of digital cameras left in the image. Generally speaking, any inherent traces left in the image by the processing components, either hardware or software, of the image acquisition pipeline, such as defective pixels [77, 78], color filter array (CFA) interpolation artifacts [41, 79], JPEG compression artifacts [80, 81], lens aberration [82, 83] or the combination of several image intrinsic characteristics [25, 84], can be utilized to link the images to the source camera. Apart from the above-mentioned techniques, the methods that attract the most attention may be those based on SPN [1, 3, 9, 45, 51, 55, 85], which mainly consists of the photo-response non-uniformity (PRNU) noise [1] arising primarily from the manufacturing imperfections and the inhomogeneity of silicon wafers. The uniqueness to individual camera and stability against environmental conditions make SPN a feasible fingerprint for identifying and linking source cameras.

However, the correlation-based detection of SPN heavily relies upon the quality of the extracted SPN, which can be severely contaminated by image content, color interpolation, JPEG compression and other non-unique artifacts. To achieve

high accuracy and reliability of identification, the size of SPN has to be very large, for example, $512 \times 512$ pixels or above. But the large size of SPN limits its applicability in some scenarios. One example is image or video forgery localization [6, 9, 72, 74, 86–88], where there exists a trade-off between localization and accuracy. Another scenario is digital camcorder identification [89], where the spatial resolution of video frames is usually much smaller than that of typical still images. One more example is camera fingerprints (SPNs) clustering [57, 58, 60]. The complexity of clustering is usually very high and the high dimension of SPNs will further bring difficulties to computation and storage. The clustering algorithm is expected to use SPNs of small size but still able to deliver good performance. Therefore, exploring the ways of improving the quality of SPNs extracted from small-sized image blocks becomes of great significance for the above-mentioned SPN-based applications.

In this chapter, we propose a new preprocessing scheme, namely Spectrum Equalization Algorithm (SEA), for the reference SPN to enhance the performance of SCI. If the reference SPN is modeled as white Gaussian noise (WGN), the theoretical analysis of WGN would indicate that the reference SPN should have a flat magnitude spectrum. Peaks appearing in the spectrum are probably originated from the periodic artifacts and unlikely to be associated with the true SPN. Therefore, by detecting and suppressing the peaks in the spectrum, we can obtain more clean (noise-like) signals. We will start by studying the limitations of existing preprocessing schemes, and then propose our SEA in detail to overcome the limitations.

## 3.2 Reference SPN Preprocessing: A Case Study

As can be found in Section 2.1 of Chapter 2, most works [9, 45, 52] focus on the processing of the reference SPN, only Li's enhancers [3] are applied on the query noise residual. The reason is that, the noise residual extracted from a single image can be severely contaminated by interfering artifacts arisen from scene details, CFA

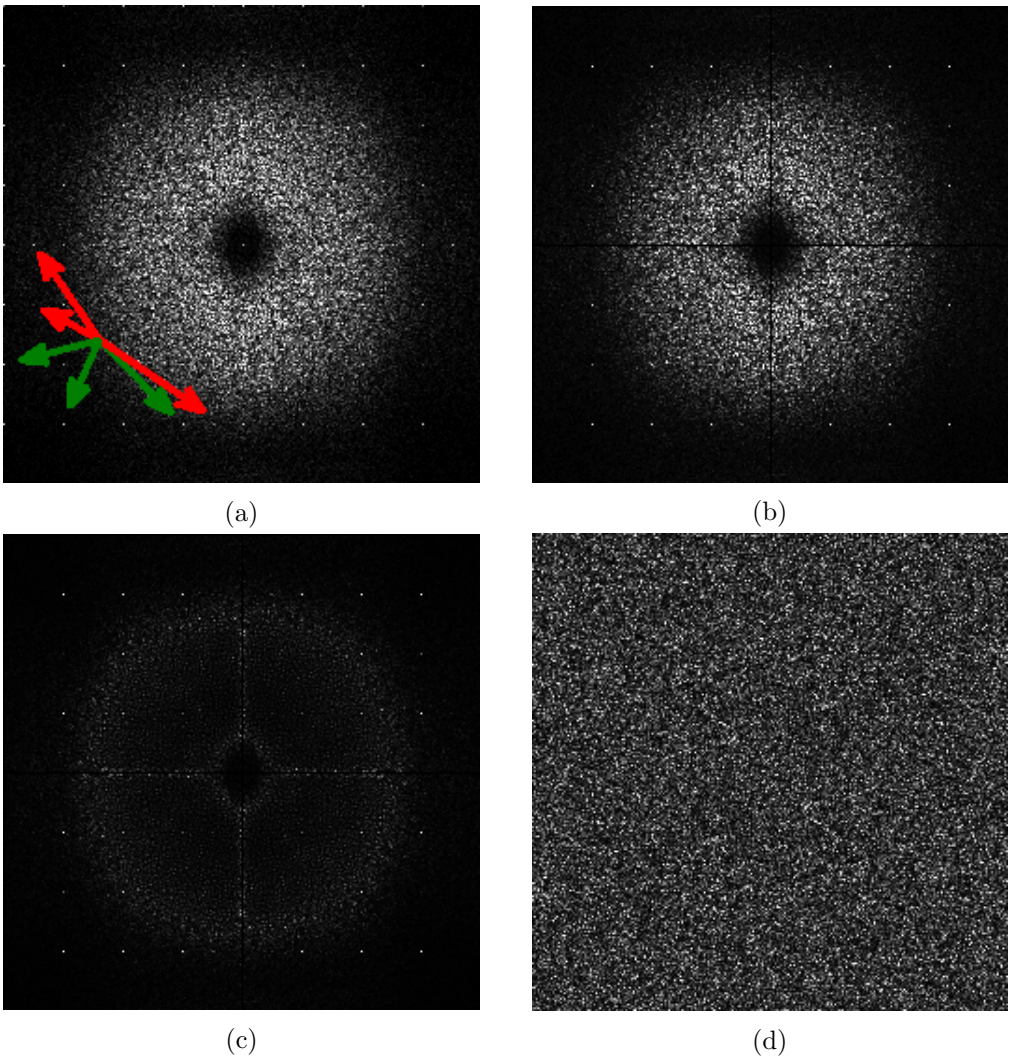


Figure 3.1: Filtering for the reference SPN of Canon PowerShot A400. (a) Spectrum of the original reference SPN, (b) spectrum of the ZM filtered reference SPN, (c) spectrum of the ZM + WF filtered reference SPN, (d) spectrum of white noise. For visualization purposes, gamma correction with an exponent of 1.5 was applied to the spectra.

interpolation, on-sensor signal transfer [90], JPEG compression and other image processing operations. Therefore, it is extremely difficult to distinguish the true SPN from the estimated SPN. While in the reference SPN, the random artifacts, such as the shot noise, read-out noise and quantization noise have been averaged out. Moreover, if the camera is available, we can take high-quality images, such as

blue sky or flat field images, to better estimate the reference SPN. As a result, we can easily incorporate our prior knowledge of SPN to refine the estimated reference SPN. In [9], for example, Chen *et al.* proposed the zero-meaning (ZM) procedure to remove the artifacts introduced by CFA interpolation, row-wise and column-wise operations of sensors or processing circuits, as well as the Wiener filtering (WF) procedure to suppress the the visually identifiable patterns in the ZM processed signal. Specifically, row and column averages are deducted from every pixel in the corresponding row and column of the reference SPN $\mathbf{R}$ to form the zero-meaned reference SPN $\mathbf{R}_{\text{zm}}$. The WF procedure is carried out by transforming $\mathbf{R}_{\text{zm}}$ into the discrete Fourier transform (DFT) domain, $\mathbf{D}_{\text{zm}}$, and applying the Wiener filter on each frequency index $[u, v]$

$$D_{\text{wf}}[u, v] = D_{\text{zm}}[u, v]\frac{\sigma_0^2}{\hat{\sigma}^2[u, v] + \sigma_0^2}, \tag{3.1}$$

where $\sigma_0^2$ represents the overall variance of $\mathbf{R}_{\text{zm}}$, and $\hat{\sigma}^2[u, v]$ is the maximum a posterior estimation of the local variance

$$\hat{\sigma}^2[u, v] = \min_{w\in\{3,5,7,9\}} \left[\max\left(0, \frac{1}{w^2}\sum_{[k,l]\in N_w} D_{\text{zm}}^2[k, l] - \sigma_0^2\right)\right], \tag{3.2}$$

where $N_w$ is a $w \times w$ local neighborhood centered at $[u, v]$. The final reference SPN $\mathbf{R}_{\text{wf}}$ is obtained by applying the inverse discrete Fourier transform (IDFT) on $\mathbf{D}_{\text{wf}}$.

We conducted experiments on a Canon PowerShot A400 to see how well ZM and WF suppress the visually identifiable patterns in the DFT spectrum. The reference SPN was estimated from 50 blue sky images captured by the Canon PowerShot A400 using BM3D [5]. Note that although we used the noise residues extracted using BM3D [5], similar results can be observed for the SPNs extracted using other extraction methods [1, 4, 9]. The white noise is drawn from the normal distribution with zero mean and the same variance as the original SPN. The DFT magnitude

spectra of the original reference SPN and its counterparts processed by ZM and ZM+WF are shown in Fig. 3.1(a), 3.1(b) and 3.1(c), respectively. For the purpose of visualization, the zero-frequency component has been shifted to the center of the spectrum. Unless otherwise specified in this chapter, we use the term "spectrum" to refer to the DFT magnitude spectrum hereinafter. For comparison, the spectrum of random white noise is also illustrated in Fig. 3.1(d). As can be seen, the peaks resulted from the periodic artifacts can be easily identified in the DFT domain (Fig. 3.1(a)). As we know that the peaks associated with one signal with period $T$ will appear in the locations $(\frac{U}{T}u, \frac{V}{T}v)$, where $U$ and $V$ are the dimensions of the spectrum, and $u, v \in \{0, 1, ..., T-1\}$. Thus, what is striking in Fig. 3.1(a) is that most of the peaks are resulted from the artifacts with period 8 (as indicated by the green arrows), but some of them are brought about by the artifacts with period 16 (as indicated by the red arrows). Due to the symmetry of the spectrum, only the peaks in one quadrant are illustrated in Fig. 3.1(a). But as shown in Fig. 3.1(b), after applying the ZM operation, the horizontal and vertical DC components are completely removed as hinted by the two "dark" intersecting lines passing through the center of the spectrum. Though the magnitudes of other frequency components remain quantitatively unchanged, the remaining peaks become visually more distinct as the peaks with dominating values have been removed. ZM removes all the DC components in the spectrum, so any artifacts lying in the two central "dark" lines will be also removed.

However, when comparing with the spectrum in Fig. 3.1(d), we found that ZM seems overly aggressive in modifying the DC components. If the WF operation is further applied to the ZM filtered SPN in the DFT domain, we can get the resultant spectrum as shown in 3.1(c). According to Equation (3.1), a magnitude spectrum coefficient, $D_{\text{zm}}[u, v]$, with a larger local variance, $\hat{\sigma}^2[u, v]$, will be suppressed more substantially than the one with a smaller local variance. Normally, the variance in the low-frequency region is relatively larger than that in the high-

frequency region. As shown in Fig. 3.1(c), most of the spectral energy concentrates in the low-frequency region, and therefore is suppressed more significantly. As a consequence, the spectrum in Fig. 3.1(c) looks "flatter" than that in Fig. 3.1(b). So with the help of ZM and WF, three improvements have been made: 1) any artifacts appearing in DC components are completely removed, 2) the spectrum is more noise-like (flat) and 3) the peaks arisen from periodic artifacts are significantly suppressed. Despite the peaks in the low-frequency region have been suppressed effectively, those in the high-frequency region are less affected, which can be clearly seen from the "white" points in Fig. 3.1(c). Therefore, in view of the effects of ZM and WF, ZM is overly aggressive in modifying the DC components and WF appears to be too conservative in suppressing the peaks in the high-frequency band. These shortcomings leave room for improving the quality of reference SPN.

## 3.3 Spectrum Equalization Algorithm (SEA)

As mentioned at the start of Section 3.2, the "purity" of the reference SPN makes it more suitable to be modified by incorporating prior knowledge of SPN, such as the fact that the true SPN signal is unlikely to be periodic and should have a flat spectrum. But the key problem is how to incorporate the prior knowledge appropriately and modify the reference SPN accordingly. When comparing the spectra of the ZM and WF filtered SPN with that of white noise, we can see that the horizontal and vertical DC components are completely removed and the peaks in the high-frequency band are still visible. Besides, the low-frequency components in the spectrum probably have been severely contaminated by scene details, so it is difficult to distinguish the true SPN signal from other low-frequency artifacts. So without enough information to ensure the global "flatness" (i.e., flatness across the entire spectrum), can we just ensure the local "flatness" of the spectrum by simply removing the salient peaks within a local neighborhood? Identifying the

periodic artifacts responsible for the prominent peaks in the spectrum can help us better understand the problem and find an appropriate solution. We summarize the periodic artifacts as follows:

- **CFA interpolation artifacts.** A typical CFA interpolation is accomplished by estimating the missing components from spatially adjacent pixels according to the component-location information indicated by a specific CFA pattern. As CFA patterns form a periodic structure, measurable offset gains will result in periodic biases in the interpolated image [9]. The periodic biases manifest themselves as peaks in the DFT spectrum, and the locations of the peaks depend on the configuration of the CFA pattern.

- **JPEG blockiness artifacts.** In JPEG compression, non-overlapping $8 \times 8$ pixel blocks are coded with DCT independently. So aggressive JPEG compression causes blockiness artifacts, which manifest themselves in the $U \times V$-point DFT spectrum as peaks in the positions $(\frac{U}{8}u, \frac{V}{8}v)$, where $u, v \in \{0, 1, ..., 7\}$.

- **Diagonal artifacts.** As reported in [53], unexpected diagonal artifacts were observed for the reference SPN of Nikon CoolPix. Although the cause is yet to be investigated, the artifacts manifest themselves in the spectrum as peaks in the frequency positions corresponding to the row and column period introduced by the diagonal artifacts.

Given that the noise-like SPN should have a flat spectrum without salient peaks, the rationale, which forms the basis of the proposed SEA for preprocessing the reference SPN, is that the peaks present in the DFT spectrum are unlikely to be associated with the true SPN, and the unnatural traces usually appear in the form of periodic patterns, such as the $2 \times 2$ or $4 \times 4$ CFA patterns, $8 \times 8$ JPEG blockiness and so on, which correspond to the peaks in fixed positions of spectrum. By suppressing these peaks, SPN of better quality can be obtained.

**Procedure 1** Spectrum Peak Detection

**R**: original reference SPN of $U \times V$ pixels;
$w$: size of a local neighborhood;
$\tau_1$, $\tau_2$: two thresholds for peak detection, $\tau_1 < \tau_2$;
**P**: $U \times V$ binary map of detected peak locations;

1: Calculate the magnitude spectrum $\mathbf{D} = \mathrm{DFT}(\mathbf{R})$;
2: Initialize two $U \times V$ binary maps $\mathbf{P}_1 = \mathbf{P}_2 = \mathbf{0}$;
3: Initialize two $U \times V$ mean matrices $\mathbf{M}_1 = \mathbf{M}_2 = \mathbf{0}$;
4: $count = 0$; $u = 1$ to $U$ $v = 1$ to $V$
5: $M_1[u,v] = \frac{\sum_{[k,l]\in N_w} |D[k,l]| \tilde{P}_1[k,l]}{\sum_{[k,l]\in N_w} \tilde{P}_1[k,l]}$;
6: $M_2[u,v] = \frac{\sum_{[k,l]\in N_w} |D[k,l]| \tilde{P}_2[k,l]}{\sum_{[k,l]\in N_w} \tilde{P}_2[k,l]}$;
7: $\mathbf{P}_1 = \mathcal{L}\big(\frac{|\mathbf{D}|}{\mathbf{M}_1} \geq \tau_1\big)$;
8: $\mathbf{P}_2 = \mathcal{L}\big(\frac{|\mathbf{D}|}{\mathbf{M}_2} \geq \tau_2\big)$;
9: $count = count + 1$; $count > 2$
10: Create a $U \times V$ binary matrix **B**, with 1s only at indices $(\frac{U}{16}u, \frac{V}{16}v), u, v = 0, 1, \ldots, 15$;
11: Screen out spurious peaks $\mathbf{P} = \mathbf{P}_1 \& \mathbf{B} | \mathbf{P}_2$;
12: **P**;

**Procedure 2** Spectrum Peak Suppression

**R**: original reference SPN of $U \times V$ pixels;
**P**: $U \times V$ detected peak locations;
$w$: size of a local neighborhood;
$\mathbf{R}_{SEA}$: $U \times V$ spectrum equalized reference SPN;

1: $\mathbf{D} = \mathrm{DFT}(\mathbf{R})$; $u = 1$ to $U$ $v = 1$ to $V$
2: $L[u,v] = \frac{\sum_{[k,l]\in N_w} |D[k,l]| \tilde{P}[k,l]}{\sum_{[k,l]\in N_w} \tilde{P}[k,l]}$;
3: $\mathbf{R}_{\mathrm{SEA}} = \mathrm{IDFT}(\frac{\mathbf{LD}}{|\mathbf{D}|})$;
4: $\mathbf{R}_{SEA}$;

SEA consists of *peak detection* and *peak suppression*, as detailed in Procedure 1 and Procedure 2, respectively. The peaks in the spectrum of the reference SPN are detected by comparing the ratio of the spectrum to the local mean with a threshold, as shown in Steps 12 and 13 of Procedure 1. When calculating the local mean within a neighborhood $w$ centered at $[u, v]$ in Steps 8 and 9 of Procedure 1, the tilde sign "$\sim$" over $\mathbf{P}_1$ and $\mathbf{P}_2$ is the logical negation operator, which excludes the spectral components indicated by the logical 0s in $\mathbf{P}_1$ and $\mathbf{P}_2$ from the calculation of the local mean $\mathbf{M}$. $\mathcal{L}(\cdot)$ in Steps 12 and 13 labels any nonzero entry of input to logical '1' and zero to logical '0', so the peaks in the magnitude spectrum $\mathbf{D}$ will be disclosed by the logical 1s stored in $\mathbf{P}_1$ and $\mathbf{P}_2$. The above steps are repeated 3 times to make the peak detection more accurate. Finally in Step 17, '&' and '|' are the logical AND and OR operator, respectively, so the potential spurious peaks not at the indices $(\frac{U}{16}u, \frac{V}{16}v), u = 0, 1, \ldots, 15,$ and $v = 0, 1, \ldots, 15$ are screened out. Having identified the peak locations, the peaks are suppressed by simply replacing them with the local mean intensities in the spectrum, as shown in Step 4 of Procedure 2. There are several remarks need to be made for SEA:

- We use two thresholds $\tau_1$ and $\tau_2$, with $\tau_1 < \tau_2$, for peak detection in Steps 12 and 13 of Procedure 1. The underlying motivation for this particular consideration is to detect peaks more liberally by using a smaller threshold $\tau_1$ in Step 12 to avoid missing the peak positions signified in $\mathbf{B}$ (corresponding to a period of 16 pixels) in Step 16 of Procedure 1, but more conservatively by using a larger threshold $\tau_2$ in Step 13 in other positions to avoid distorting the true SPN. Because spurious peaks are more likely to be detected with a smaller threshold $\tau_1$, suppressing these spurious peaks will probably distort the true SPN. But with a larger threshold $\tau_2$, the prominent peaks can be detected without worrying about being excessively modified.

- We only consider the artifacts with a period of up to 16 pixels. Albeit the fact

that the artifacts sometimes appear with different periods, such as the $3 \times 3$ or $6 \times 6$ CFA pattern, and they may also contain components with period larger than 16. The justification is twofold: 1) the $2 \times 2$ and $4 \times 4$ are the most common CFA patterns; 2) peaks caused by the artifacts with other periods may overlap in the peak locations hinted in $\mathbf{B}$. For example, half of the peaks caused by the $32 \times 32$ periodic signal will appear in the peak locations of $\mathbf{B}$. As a consequence, peaks in positions indicated by $\mathbf{B}$ account for the dominant components caused by the majority of periodic artifacts, which is consistent with our observation in Fig. 3.1 and the following experiments in Section 3.4.5. Even if the prominent peaks are missed out by $\mathbf{B}$, they will still be caught out by $\mathbf{P}_2$.

- Although some similarity measurements, such as PCE, CCN or SPCE, also aim at suppressing the periodic noise contamination, there are two fundamental differences between our SEA and the aforementioned similarity measurements:

    1. The calculation of the similarity measure involves the reference SPN and the query noise residual. However, the query noise residual is likely to be severely contaminated by other interfering artifacts. As a consequence, even for the query noise residual extracted from a high-quality image, the periodic patterns are very inconspicuous, making the peaks discovered by the circular cross correlation in PCE, CCN and SPCE not so remarkable as those found in the spectrum of the reference SPN.

    2. The similarity measurements require extra computation for every SPN pair, which will largely increase the computational complexity. However, with the proposed SEA, we only need to apply it on the reference SPN once for all, which will save a considerable amount of time for large databases.

## 3.4 Experiments

### 3.4.1 Experimental Setup

We first evaluated the performance of the proposed preprocessing scheme on the Dresden Image Database [91]. The basic information of the used cameras can be found in Table 3.1. 49 cameras, covering 15 models and 10 brands, that have contributed 50 flatfield images were chosen. The 50 flatfield images were used to estimate the reference SPN for each camera, and another 150 natural images captured by the same camera served as query images. As mentioned in [53], unexpected artifacts were observed in the estimated reference SPN of Nikon CoolPix S710, FujiFilm FinePix J50 and Casio EX-Z150, so we will take special care for the 13 cameras of these 3 models after analyzing the remaining 36 cameras as the general cases.

Apart from the effectiveness, the robustness of the proposed scheme against JPEG compression was also investigated on our own uncompressed image database, as detailed in Table 3.2. 600 natural images taken in BMP format by six cameras were used in this experiment. The images contain a wide variety of natural indoor and outdoor scenes taken during holidays, around campus and cities, in offices and sports center, etc. Among the 100 images captured by each camera, 50 were randomly chosen to estimate the reference SPN, and the other 50 were used as query images. For each BMP image, compressed images were produced by libjpeg [92] with quality factor of 100%, 90%, 80%, 70%, 60% and 50%. Therefore, 6 groups JPEG images, i.e., $3,600$ in total, with different quality factors were generated.

We aim to compare the performances of different preprocessing schemes, but with different SPN extraction techniques continue to appear, it would be interesting to see how well the preprocessing schemes work in conjunction with existing SPN extractors. What is more, comparing the performance of different algorithms on real-world databases provides insight into the advantages and disadvantages of each algorithm, and offers a valuable reference for practical applications. Bearing this in

| Camera Model | Resolution | Number of devices |
|---|---|---|
| Canon Ixus55 | $2,592 \times 1,944$ | 1 |
| Canon Ixus70 | $3,072 \times 2,304$ | 3 |
| Casio EX-Z150 | $3,264 \times 2,448$ | 5 |
| FujiFilm FinePix J50 | $3,264 \times 2,448$ | 3 |
| Nikon CoolPix S710 | $4,352 \times 3,264$ | 5 |
| Olympus Mju 1050SW | $3,648 \times 2,736$ | 5 |
| Pentax OptioA40 | $4,000 \times 3,000$ | 4 |
| Pentax OptioW60 | $3,648 \times 2,736$ | 1 |
| Praktica DCZ5.9 | $2,560 \times 1,920$ | 5 |
| Rollei RCP-7325XS | $3,072 \times 2,304$ | 3 |
| Samsung L74wide | $3,072 \times 2,304$ | 3 |
| Samsung NV15 | $3,648 \times 2,736$ | 3 |
| Sony DSC-H50 | $3,456 \times 2,592$ | 2 |
| Sony DSC-T77 | $3,648 \times 2,736$ | 4 |
| Sony DSC-W170 | $3,648 \times 2,736$ | 2 |

Table 3.1: 49 cameras involved in the creation of the images in the Dresden database

mind, we incorporated six SPN extraction or enhance algorithms in the experiments. For the sake of convenience, we refer to the technique in [1] as "Basic", [9] as "MLE", [3] as "Enhancer", [5] as "BM3D", [45] as "Phase", and [4] as "PCAI8". Although "Enhancer" only enhances the query noise residual and has nothing to do with the noise extraction, hereinafter we refer to all these six algorithms as SPN extractors for convenience. For the preprocessing schemes, we refer to zero-mean operation as ZM, the Wiener filter in the DFT domain as WF, the combination of ZM and WF operations as ZM+WF, and the proposed spectrum equalization algorithm as SEA.

### 3.4.2 Parameters Setting

For Basic [1], MLE [9] and Phase [45], we used the source codes published in [55, 56]. For Li's Enhancers [3], we adopted Model 3 with $\alpha = 6$ because it shows better results than his other models. For BM3D [5], we downloaded the source codes from

| Camera Model | Resolution | Number of images |
|---|---|---|
| Canon 450D | $4,272 \times 2,848$ | 100 |
| Canon Ixus870 | $1,600 \times 1,200$ | 100 |
| Nikon D90 | $4,288 \times 2,848$ | 100 |
| Nikon E3200 | $2,048 \times 1,536$ | 100 |
| Olympus C3100Z | $2,048 \times 1,536$ | 100 |
| Panasonic DMC-LX2 | $3,168 \times 2,376$ | 100 |

Table 3.2: 6 cameras involved in the creation of the images in our own database

[93] and simply used the default parameters. To facilitate fair comparison, we set the noise variance $\sigma_0^2 = 4$ for all the algorithms that use Mihcak filter [2], as well as BM3D and PCAI8.

For SEA, the neighborhood size $w$ and the thresholds $\tau_1$ and $\tau_2$ for peaks detection depend on the strength of periodic artifacts and the spectrum distribution. If the neighborhood size or threshold is too small, the peaks detector is too sensitive to local variations. But if the neighborhood size or threshold is too large, it cannot capture the local characteristics of the spectrum. We experimentally set $w$ to 17 and 15 (selected from $[8, 32]$) for Procedures 1 and 2, respectively. The thresholds $\tau_1$ and $\tau_2$ in Procedure 1 are set to 3.0 and 3.4 (selected from $[2.0, 5.0]$), respectively. From the visual observation of the spectrum, the selected parameters work well for the cameras we have tested in the Dresden database. For all the methods involved, only the green channel of images is used. In addition, to better simulate the real-world applications such as image or video forgery localization, the experiments were performed on image blocks with different sizes cropped from the center of the full-resolution images due to the vignetting effects [94]. When needed in the rest of this chapter, we will use the terms "large", "medium" and "small" to refer to the sizes of $1024 \times 1024$, $256 \times 256$ and $64 \times 64$ pixels, respectively. It is worth noting that all preprocessing schemes will only be applied on the reference SPN due to the reason mentioned at the beginning of Section 3.2. In the following experiments, NCC, as

defined in Equation (2.2), will be used as the similarity measurement between the reference SPN and the query noise residual, but the results of SPCE, as defined in Equation (2.24), will also be presented when necessary.

### 3.4.3 Evaluation Statistics

To demonstrate the performance of the proposed preprocessing scheme, we adopted two evaluation statistics, namely the overall receiver operating characteristic (ROC) curve [4, 45] and the kappa statistic [95] computed from a confusion matrix.

To obtain the overall ROC curve, for a given detection threshold, the true positives and false positives are recorded for each camera, then these numbers are summed up and used to calculate the True Positive Rate (TPR) and False Positive Rate (FPR). Specifically, as the numbers of images captured by each camera are exactly the same, we can simply calculate the TPR and FPR for a threshold as follows:

$$\begin{cases} \text{TPR} = \dfrac{\sum_{i=1}^{C} \mathcal{T}_i}{T} \\ \text{FPR} = \dfrac{\sum_{i=1}^{C} \mathcal{F}_i}{(C-1)T}, \end{cases} \tag{3.3}$$

where $C$ is the number of cameras, $T$ is the total number of query images, $\mathcal{T}_i$ and $\mathcal{F}_i$ are the true positives and false positives of camera $i$, respectively. By varying the detection threshold from the minimum to the maximum value as calculated using Equation (2.2), we can obtain the overall ROC curve.

To obtain the confusion matrix $\boldsymbol{\mathcal{M}}$, we calculated the similarity between the noise residue of one query image and the reference SPN of each camera, then this image was deemed to be taken by the camera corresponding to the maximal similarity. The value of each element $\mathcal{M}[i, j]$ in the confusion matrix indicates the number of images taken by camera $i$ that have been linked to camera $j$ as the source device. In other words, the values along the main diagonal indicate the numbers

of correct identifications. Each confusion matrix can be reduced to a single value metric, kappa statistic $\mathcal{K}$ [95]:

$$\mathcal{K} = \frac{o - e}{T - e}, \tag{3.4}$$

where $o$ is the number of observed correct identifications, i.e., the trace of confusion matrix, $T$ is the total number of query images, and $e$ is the number of expected correct identifications:

$$e = \sum_{c=1}^{C} \frac{\sum_{i=1}^{C} \mathcal{M}[c,i] \sum_{j=1}^{C} \mathcal{M}[j,c]}{T}, \tag{3.5}$$

where $C$ is the number of cameras. Kappa statistic measures the disagreement between the observed results and random guess, therefore the larger the value of $\mathcal{K}$, the better the performance, with 1 indicating the perfect performance.

The reason why both the overall ROC curve and the kappa statistic are used is that we want to properly evaluate the performances of two different SPN-based real-world applications, SCI and forgery detection, which are the same in essence while differing in minor points. Normally, in the task of forgery detection, the similarity measurement, between the reference SPN and the noise residue extracted from the image block in question, is directly compared with a threshold suggested by some criterion, such as the Neyman-Pearson criterion, to determine whether the image block has been tampered with or not. This process is equivalent to the generation of one point in the ROC curve, therefore it is more appropriate to evaluate the performance of forgery detection using the overall ROC curve. While in the context of SCI, the query image is believed to be taken by the camera with the maximal similarity which is greater than a predefined threshold at the same time. It is more like the process of creating a confusion matrix. So the kappa statistic computed from a confusion matrix is a preferable evaluation statistic for SCI.

### 3.4.4 General Cases

Before delving into the details, let us first look at a straightforward comparison of the effects of different preprocessing on the spectrum of the reference SPN. Fig. 3.2(a)-3.2(c) are the corresponding 3D spectra of Fig. 3.1(a)-3.1(c), while Fig. 3.2(d) shows the spectrum of the SEA filtered SPN. To reduce the dynamic range of magnitudes and make the peaks more conspicuous, a $3 \times 3$ averaging filter is convolved with the spectrum beforehand. As can be clearly seen in Fig. 3.2(d), when compared with the spectrum processed by the ZM and ZM+WF, the "spiky" interferences have been nicely smoothed out by SEA while the rest of the spectrum still remains untouched. In this manner, the true SPN has been preserved as much as possible.

The advantages of SEA can also be observed in Fig. 3.3, where we show the estimated inter-class (in light blue color) and intra-class (in light green color)

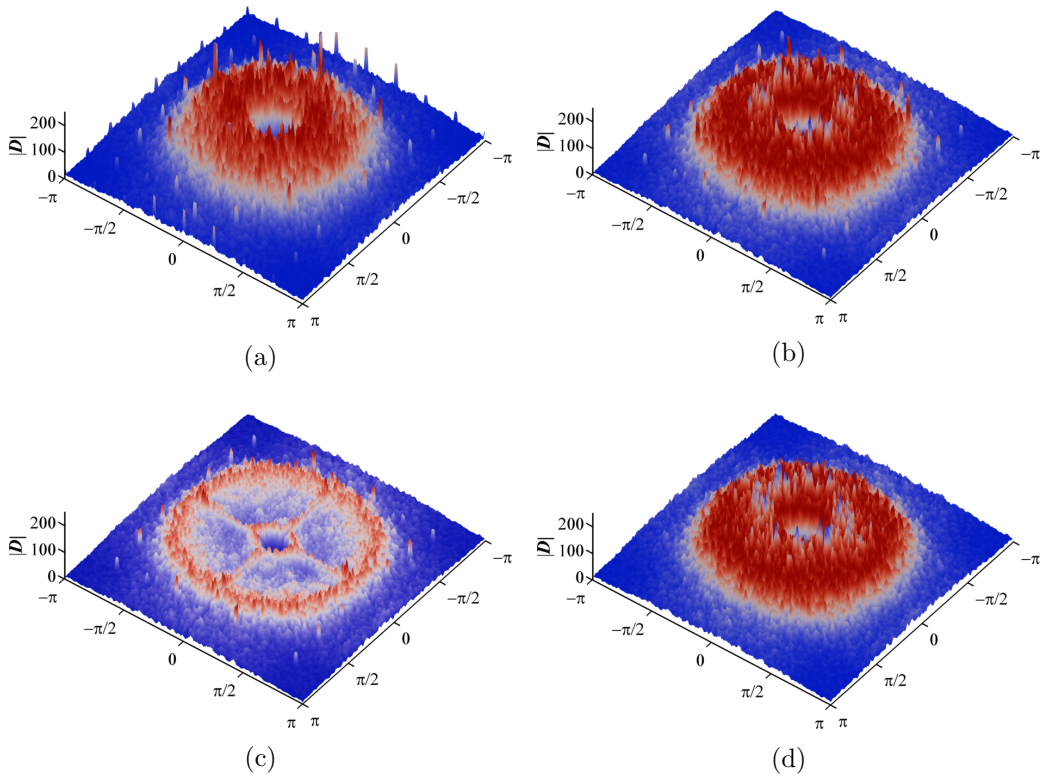



Figure 3.2: (a) Spectrum of the original reference SPN and the ones preprocessed by (b) ZM, (c) ZM+WF and (d) SEA.

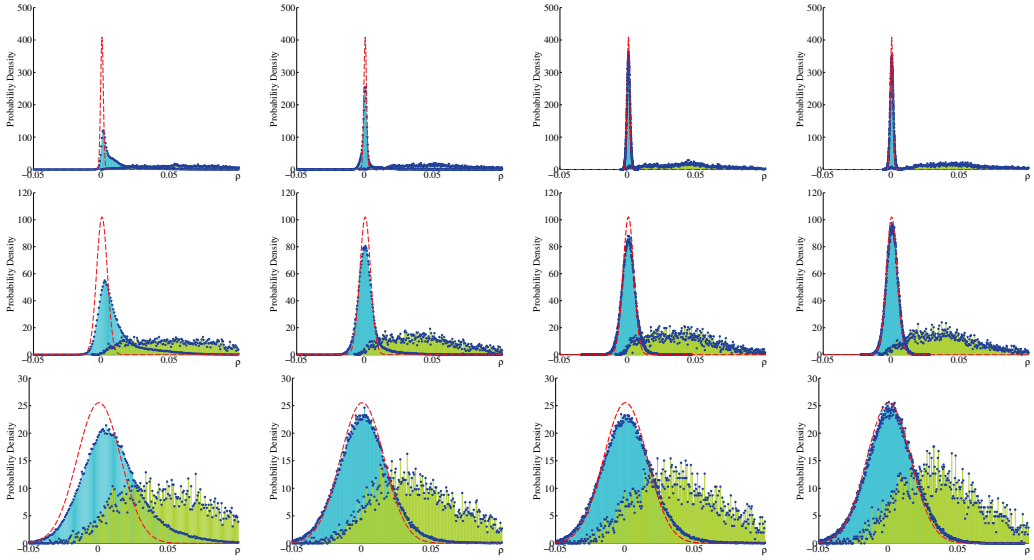



Figure 3.3: Estimated inter-class and intra-class PDFs of $\rho$ calculated from SPNs extracted from 3 different sizes of image blocks using BM3D. From top to bottom, the rows show the distributions for image blocks sized $1024 \times 1024$, $256 \times 256$ and $64 \times 64$ pixels, respectively. From left to right, the distributions are resulting from the original reference SPN and the ones preprocessed by ZM, ZM+WF and SEA, respectively.

probability density functions (PDFs) of the correlation value $\rho$ for the 36 cameras in the Dresden database. The values outside the range of $[-0.05, 0.1]$ are cut off to make the figures look more compact. As shown in the first column, due to the long right-hand tail of the inter-class distribution, there are a considerable amount of overlaps between the inter-class and intra-class distributions if no preprocessing is applied. After ZM and WF are applied sequentially, the long tail on the right-hand side of the inter-class distribution is curtailed and the inter-class variances significantly decrease from $2.46 \times 10^{-4}$, $3.49 \times 10^{-4}$ and $6.04 \times 10^{-4}$ to $1.44 \times 10^{-6}$, $2.65 \times 10^{-5}$ and $3.55 \times 10^{-4}$ for large, medium and small image blocks, respectively. As a result, the overlaps between the inter-class and intra-class correlation distribution are reduced substantially. Compared with the inter-class distribution brought about by ZM+WF, the resulting inter-class distribution of SEA looks even "thinner", with a smaller variance $1.89 \times 10^{-5}$ and $2.95 \times 10^{-4}$ for medium and small image

blocks, respectively. The smaller variance of inter-class distribution makes the two distributions more separable from each other, and therefore boosts the performance. For the large size, $1024 \times 1024$ pixels, the variance of inter-class distribution for SEA is $1.47 \times 10^{-6}$, which is slightly larger than that of ZM+WF, $1.44 \times 10^{-6}$. But the intra-class mean for SEA, 0.05, is slightly larger than the 0.045 for ZM+WF. So considering these two aspects, SEA and ZM+WF are comparable to each other in the case of large image blocks, which will also be quantitatively reflected in Fig. 3.4 and 3.5. In addition, we can see that the inter-class distribution resulting from SEA fits quite well to the theoretical distribution, which is a normal distribution with 0 mean and $1/d$ variance (in red dashed lines), where $d$ is the length of SPNs.

We will reveal more details on the comparison of different combinations of SPN extraction algorithms and preprocessing schemes in terms of the overall ROC curve. Fig. 3.4 shows the overall ROC curves of the combinations of the six SPN extraction algorithms and three preprocessing schemes on image blocks with different sizes. As it is desirable to see the TPR at a low FPR, the ROC curves are plotted in the logarithmic scale to show more details of the area where a FPR is low. The curves for the original reference SPN and the ones filtered by ZM, ZM+WF and SEA are highlighted in red, cyan, green and pink colors, respectively. With respect to different preprocessing schemes, the pink lines keep standing over other lines in Fig. 3.4 for different sizes and different extraction algorithms, indicating SEA stands out at the top of the list. ZM+WF takes the second place, followed sequentially by ZM and without preprocessing. In the case of large image size, the superiority of SEA over ZM+WF is not apparent, but SEA is in a clearly advantageous position especially for medium and small image blocks, i.e., $256 \times 256$ and $64 \times 64$ pixels. In the case with large block size, except for the Phase method, SEA and ZM+WF outperform the other two schemes by a wide TPR margin, more than 0.6 for a FPR smaller than $1 \times 10^{-3}$.

Fig. 3.5 depicts the TPRs at a FPR as small as $1 \times 10^{-3}$. The dark red

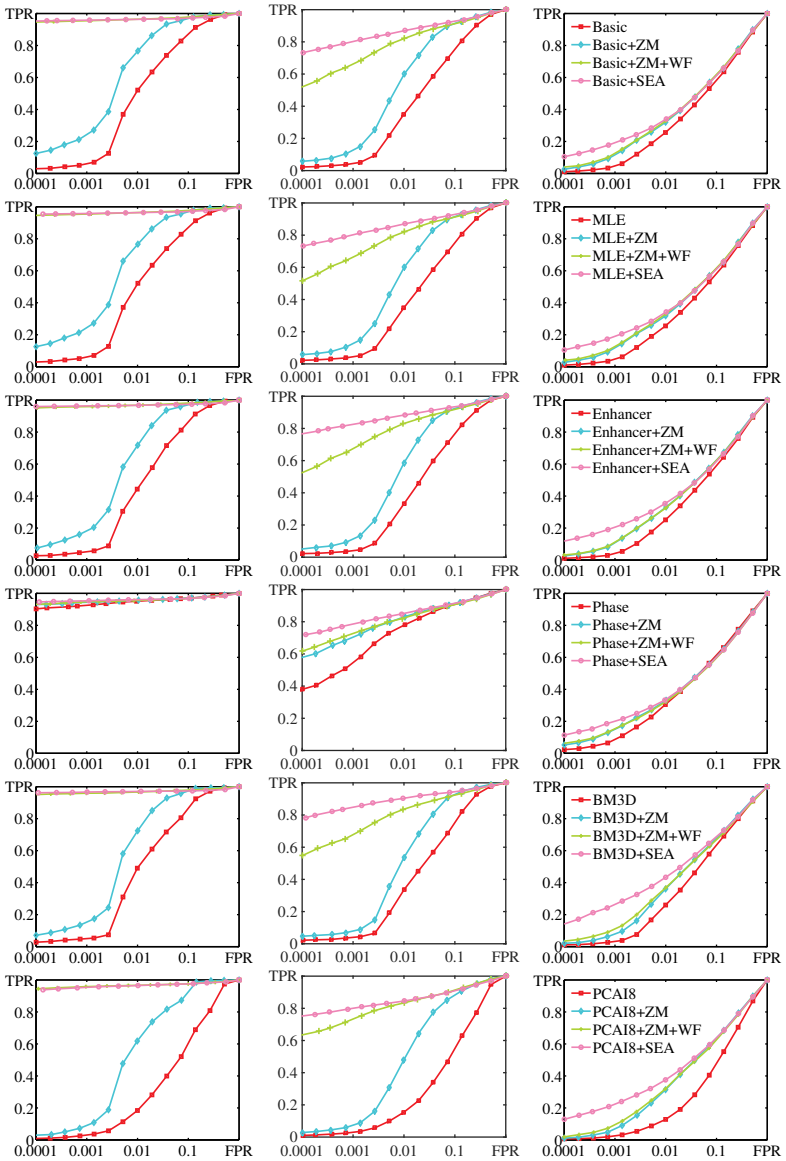
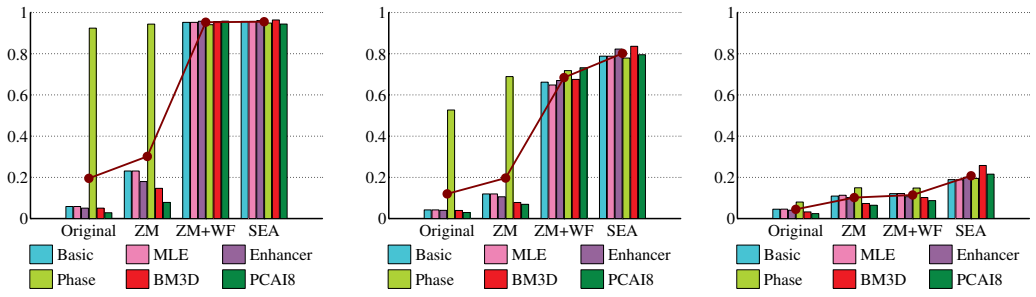



Figure 3.4: Overall ROC curves of the combinations of different SPN extractors and preprocessing schemes (different columns) on different image block sizes (different rows). From left to right, the columns show the ROC curves for image blocks of $1024 \times 1024$, $256 \times 256$ and $64 \times 64$ pixels, respectively. Please refer to the last column for the legend text, which is the same for the figures in the same row.

dotted line shows the average of each group corresponding to one preprocessing scheme. Generally speaking, preprocessing can substantially increase the TPR at a low FPR. Similar with the observation in Fig. 3.3, SEA is equally matched with ZM+WF for large image size, but has higher TPRs than ZM+WF for medium and

small sizes. With regard to different SPN extractors, Phase is stable against various preprocessing, resulting in its outstanding position when preprocessing is not applied. The underlying reason is that Phase only retains the phase component but ignores the magnitude components of each noise residual that are used to estimate the reference SPN. In this way, the periodic artifacts have been suppressed considerably but not completely removed. So preprocessing can further, although slightly, improve the performance of Phase, as can be seen from the green bins in Fig. 3.5. The performance of Basic, MLE and Enhancer are equivalent in many respects, but Enhancer exceeds the other two when combined with SEA. Interestingly, BM3D and PCAI8 perform worse than the other SPN extractors when no or little preprocessing is applied, but they outperform other extractors with the help of ZM+WF or SEA. By using the non-local information to get the better noise estimation [5], it is not surprising that BM3D performs consistently the best among all six extractors in case of being combined with SEA. It is also worth mentioning that PCAI8 performs even worse than Phase when combined with ZM or ZM+WF. This is probably due to insufficient images to create trustworthy reference SPN [4].




(a) (b) (c)

Figure 3.5: TPRs at the FPR of $1 \times 10^{-3}$ for image blocks sized (a) $1024 \times 1024$ (b) $256 \times 256$ and (c) $64 \times 64$ pixels.

Similar tendencies can be observed from the kappa statistics listed in Table 3.3-3.5. To clearly show which extraction methods and which preprocessing schemes perform better, the maximal entry of each column is highlighted with a gray background, and the maximum value of each row is highlighted in bold. That is to say,

a gray background in a column denotes the best extraction method when combined with the preprocessing scheme corresponding to the column, while a bold value in a row signifies the best preprocessing scheme for the corresponding extraction method of the row. So the optimal combination of extraction method and preprocessing scheme is the entry in both gray background and bold font style. As shown in the three tables, most of the bold numbers appear in the last two columns, indicating the effectiveness of ZM+WF and SEA. The most apparent example is PCAI8, of which the kappa statistic increases by approximately 0.2 after preprocessed by ZM+WF or SEA when the image block size is $256 \times 256$ pixels. For the case of $64 \times 64$ blocks, in spite of the slight performance improvements, the reference SPNs extracted with the Michak filter seem to be vulnerable to preprocessing. But for the other two extractors, BM3D and PCAI8, the performance gain is still noticeable. Special attention should be paid to Table 3.3, where ZM+WF performs slightly better than SEA. The average kappa statistic gap 0.0006, between SEA and ZM+WF, is negligible when compared with the average kappa statistic gain 0.0162 and 0.02 in the cases of medium and small image blocks, respectively. To understand it more intuitively, we took a close look at the confusion matrix. We found that, for image blocks sized $1024 \times 1024$ pixels, ZM+WF has only about an average of 3 more correctly classified images than SEA among the 5400 images from 36 cameras. But the average number of correctly classified images by SEA is around 85 and 105 more than ZM+WF for the medium and small image blocks, respectively. As for different extractors, Basic, MLE and Enhancer perform comparably well in all conditions. This is consistent with our observations in Fig. 3.4 and 3.5. As indicated by the gray backgrounds in the 3 tables, BM3D shows a clear superiority over other extractors. Moreover, with the help of SEA, BM3D exhibits the superior (or at least equivalent) performance over other combinations for all block sizes. Therefore, the joint use of BM3D and SEA is preferable for both forgery detection and SCI in practice.

JPEG is probably the most common image format used in digital cameras,

| | Preprocessing | | | |
|---|---|---|---|---|
| | Original | ZM | ZM+WF | SEA |
| Basic | 0.9236 | 0.9644 | **0.9691** | 0.9684 |
| MLE | 0.9236 | 0.9646 | **0.9695** | 0.9682 |
| Enhancer | 0.9059 | 0.9623 | **0.9722** | 0.9701 |
| Phase | **0.9650** | 0.9632 | 0.9610 | 0.9629 |
| BM3D | 0.9192 | 0.9651 | **0.9718** | 0.9714 |
| PCAI8 | 0.7366 | 0.9545 | **0.9701** | 0.9691 |

Table 3.3: Kappa statistics for $1024 \times 1024$ image blocks

| | Preprocessing | | | |
|---|---|---|---|---|
| | Original | ZM | ZM+WF | SEA |
| Basic | 0.7943 | 0.8676 | 0.8697 | **0.8895** |
| MLE | 0.7943 | 0.8674 | 0.8697 | **0.8897** |
| Enhancer | 0.7922 | 0.8714 | 0.8785 | **0.8979** |
| Phase | **0.8770** | 0.8710 | 0.8573 | 0.8691 |
| BM3D | 0.8097 | 0.8912 | 0.8901 | **0.9242** |
| PCAI8 | 0.6634 | 0.8269 | **0.8703** | 0.8621 |

Table 3.4: Kappa statistics for $256 \times 256$ image blocks

so we compared the robustness of ZM+WF and SEA against JPEG compression. The experiments were carried out for different image sizes and different SPN extractors. Sometimes the source devices are available to capture high-quality images for reference SPN estimation. So under this scenario, we can use the reference SPN estimated from images with a high quality factor 100% for each of the six cameras in Table 3.2, and calculate the similarity between the high-quality reference SPN and the query noise residual extracted from images with different quality factors. But the more plausible scenario is that only the images rather than the source devices are available. So we simulated this scenario by estimating the reference SPN using the images with the same JPEG quality as the query images. The ratios of the kappa statistics of SEA to that of ZM+WF for these two scenarios are shown in the first and second column of Fig. 3.6, respectively. The dark red dotted lines show

| | Preprocessing | | | |
|---|---|---|---|---|
| | Original | ZM | ZM+WF | SEA |
| Basic | **0.4116** | 0.4067 | 0.3962 | 0.4070 |
| MLE | **0.4122** | 0.4072 | 0.3952 | 0.4086 |
| Enhancer | **0.4061** | 0.4030 | 0.3907 | 0.4044 |
| Phase | **0.4055** | 0.3735 | 0.3724 | 0.3859 |
| BM3D | 0.4838 | 0.4865 | 0.4625 | **0.5046** |
| PCAI8 | 0.3918 | 0.4166 | 0.4010 | **0.4276** |

Table 3.5: Kappa statistics for $64 \times 64$ image blocks

the average of each group corresponding to one quality factor. A ratio greater than 1 indicates that SEA outplays ZM+WF. We adjusted the y-axis limits to accommodate bins with various heights. As indicated by the dotted lines, the generally higher average ratios in the second column benefit from SEA's high capability of removing the more evident JPEG artifacts in the reference SPN estimated from more aggressively compressed images. For the medium (Fig. 3.6(c) and 3.6(d)) and the small (Fig. 3.6(e) and 3.6(f)) image sizes, most of the average kappa statistics are higher than 1, indicating SEA's superiority over ZM+WF. The growing preponderance of the ratios as the images undergo more aggressive JPEG compression, especially in the second column, indicates that SEA tends to be more robust against JPEG compression for medium and small image blocks. But surprisingly, ZM+WF performs better than SEA in the case of large image blocks. We carefully investigated the spectra of the six cameras and found that unlike the peaks spreading out over the spectrum, as shown in Fig. 3.1(a), all the prominent peaks appear in the borders of the spectrum and the locations indicated by the two "dark" lines in Fig. 3.1(b), which can be completely removed by ZM. But for large images, the components overly modified by ZM are not so considerable as for small images. Therefore, it introduces bias in favor of ZM+WF for large image size. Another cause comes from the fact that by using larger image blocks, it is more likely to have a more enriched and spread-out spectrum, and therefore make some of the peaks fade away into the

background. It is also the reason why SEA limits the further improvement for large blocks in Fig. 3.4 and Table 3.3.

We then measured the average signal-to-noise ratio (SNR) of noise residue (extracted from the uncompressed BMP images) and JPEG quantization noises to the uncompressed images for each of the six cameras. As shown in Fig. 3.7, when the JPEG quality factor drops to 70%, the SNR of quantization noise is even higher than that of noise residual for four of the six cameras. It indicates that the impact of JPEG compression on the quality of SPN and thus the identification performance can be significant. For example, with $256 \times 256$ blocks, when the quality factor of the query images decreases from 100% to 70%, the average kappa statistic over six extractors dramatically declines from 0.9433 to 0.6533 for SEA, and from 0.9460 to 0.6360 for ZM+WF in the first scenario, and even lower in the second scenario, with an average kappa statistic of 0.5767 for SEA and 0.5487 for ZM+WF. But the effects of JPEG compression appear to be much less severe for $1024 \times 1024$ blocks. Even for the 50% quality factor and the second scenario, the average kappa statistics are still considerable, with 0.7680 for SEA and 0.7800 for ZM+WF. So with large enough block size, even if the images undergo high JPEG compression, accurate SCI is still possible.

### 3.4.5 Special Cases

As mentioned in [53], some unexpected artifacts, which might have stemmed from the dependencies between sensor noise and special camera settings or some advanced in-camera post-processing, were observed in the images taken by Nikon CoolPix S710, FujiFilm FinePix J50 and Casio EX-150. More specifically, a diagonal pattern can be clearly seen in the reference SPN of Nikon CoolPix S710 in the spatial domain and manifests itself as peaks in the DFT domain (see Fig. 3.8(a) and 3.8(b)). As the diagonal structures are only observed in images taken by CoolPix S710, it is probably due to the special in-camera post-processing in CoolPix S710. For Fu-

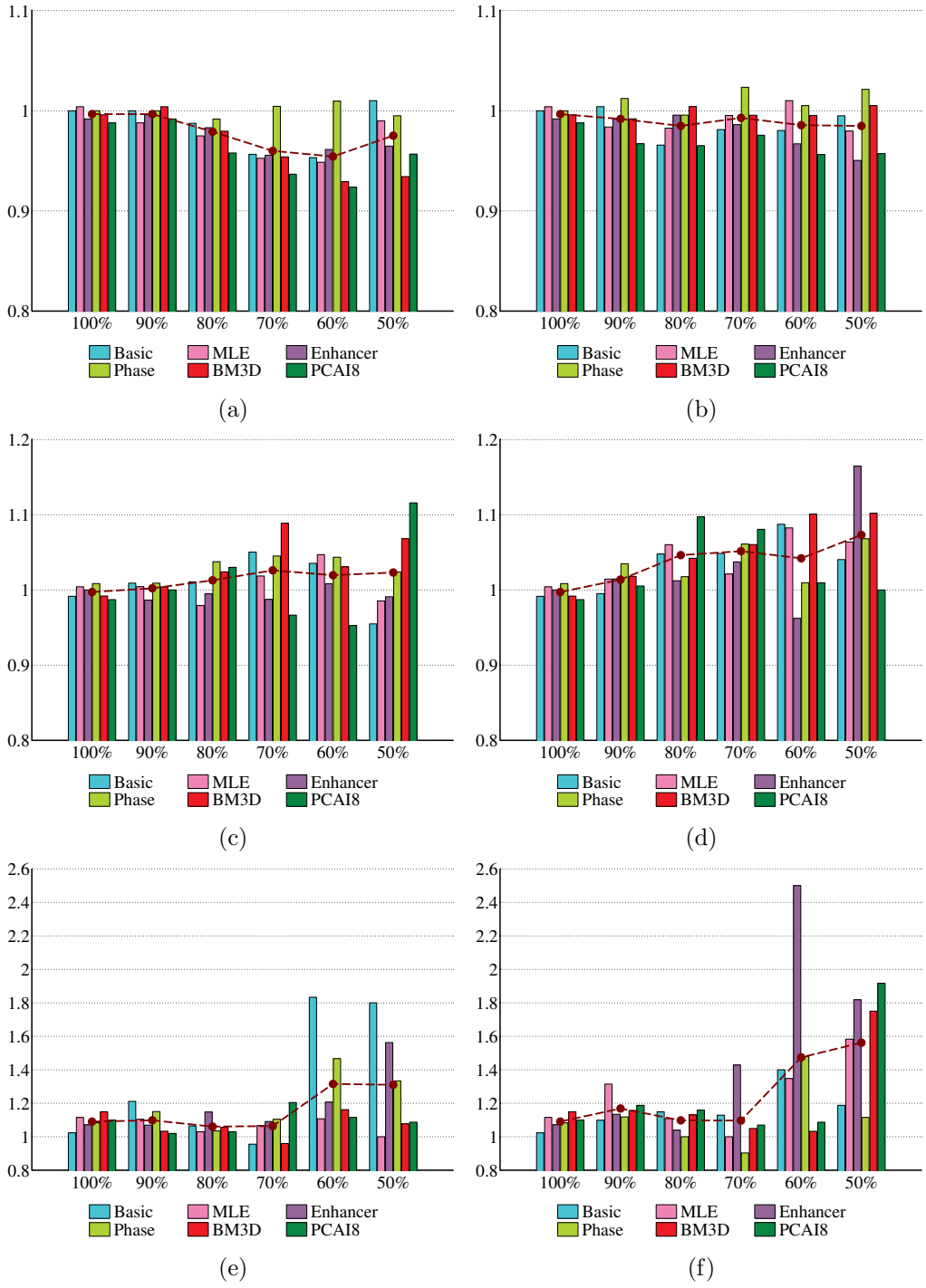



Figure 3.6: Ratios of the kappa statistic of SEA to that of ZM+WF for the cases of estimating the reference SPN from JPEG images with a quality factor 100 (first column) and JPEG images with the same quality factor as the query images (second column). Bins are grouped according to the quality factor of the query images, and each of the six bins in the same group shows the ratio for one of the six SPN extractors. From top to bottom, the rows show the results for image blocks of $1024 \times 1024$, $256 \times 256$ and $64 \times 64$ pixels.

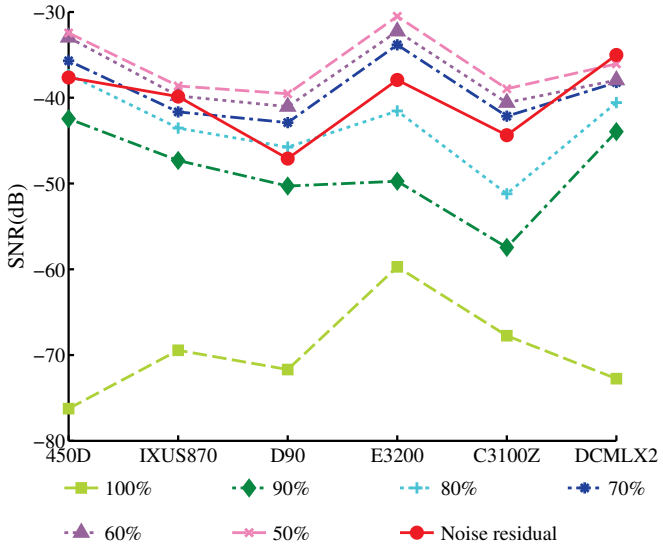



Figure 3.7: SNR for noise residual and the quantization noise introduced by JPEG compression.

jiFilm FinePix J50, the identification results have a relationship with the difference between the exposure times when capturing the images used for estimating the reference and the image used for extracting the query noise residual. It is possibly that some exposure-time-dependent post-processing procedure is employed in FujiFilm FinePix J50, for instance to suppress the noise [53]. The experimental results also confirm that SPNs of FujiFilm FinePix J50 at exposure times $\geq 1/60$s exhibit pixel shifts in horizontal direction. The worst case among the three models is Casio EX-150, the identification performance of which is very poor for images taken at different focal length settings. The image distortions become clear by showing the *p-maps* [29] of images acquired by EX-150. The origin of the artifacts are still unknown to us, but it reminds us to pay particular attention to these 3 models. Thus, separate experiments have been conducted for the 13 cameras of these 3 special models. We only conducted the experiments on blocks of $256 \times 256$ pixels, since similar properties and trends were observed for other sizes. The kappa statistics based on both NCC and SPCE are listed in Table 3.6-3.8 for a more comprehensive comparison. Comparing the kappa statistics of different preprocessing schemes in

Figure 3.8: Spectra of the reference SPNs of the three special camera models. From top to bottom, the rows show the spectra for Nikon CoolPix S710, FujiFilm FinePix J50 and Casio EX-Z150, respectively. From left to right, the columns show the spectra of the original reference SPNs and the ones filtered by ZM, ZM+WF and SEA, respectively.

Table 3.6-3.7, we found that SEA can improve the performance for Nikon CoolPix S710 and FujiFilm FinePix J50. When taking a closer look at Table 3.6 for Nikon CoolPix S710, the performances of all preprocessing methods are comparable in terms of NCC and SPCE. But compared with ZM+WF, the advantage of SEA becomes obvious in Table 3.7 for FujiFilm FinePix J50. For instance, the kappa statistic of Enhancer increases from 0.7633 to 0.8100 in terms of NCC, and from 0.7700 to 0.8033 in terms of SPCE. However, for Casio EX-150, in spite of the slight performance gain brought about by preprocessing, correct and reliable identifications are still impossible for the images captured by this model. Another important observation is that the performances of ZM+WF+SPCE and SEA+NCC are comparable for Nikon CoolPix S710 and Casio EX-150, as shown in Table 3.6 and 3.8, but SEA+NCC is significantly better than ZM+WF+SPCE for FujiFilm FinePix J50, as shown in Table 3.7. Due to the reasons we mentioned in Section 3.3, it is easier to detect the prominent peaks in the spectrum of the reference SPN using SEA than SPCE, attributing to the better performance of SEA+NCC for FujiFilm FinePix J50. Furthermore, we can see from the last column of Table 3.6-3.8 that SPCE can not further improve the performance of the reference SPN filtered by SEA, or only by a limited amount (for PCAI8). This is due to the fact that the periodic artifacts have been mostly and effectively suppressed by SEA.

Further investigations with the spectra of the 3 camera models, as illustrated in Fig. 3.8, may unveil the causes of the difference in performance. Because ZM only deals with the DC components, the two peaks associated with the diagonal artifacts reported in [53] are not removable by ZM, as shown in Fig. 3.8(b). The good news is that the two peaks can be well suppressed by both WF and SEA. However, as can be seen from Table 3.6, the effect of the suppression is not so significant as expected because the energy of the peaks only takes up a small proportion of the overall spectrum energy. For FujiFilm FinePix J50, although WF can effectively suppress the peaks in the areas with a large local variance, it appears to be helpless

in suppressing the peaks in the areas with a small local variance. When zooming in on Fig. 3.8(g), one will find that peaks still exist in the high-frequency band. Despite the much smaller magnitude of the peaks, the overall spectrum has also been substantially reduced at the same time, so the suppression is not so effective as it looks like. This can explain why SEA performs better than ZM+WF for FujiFilm FinePix J50, as shown in Table 3.7. Although we are still unable to provide convincing explanations for the poor performance of Casio EX-Z150, as shown in the last row of Fig. 3.8, the ratio of the energy of low-frequency band to that of high-frequency band seems much higher than those of the other two cameras even in the equalized spectrum, suggesting that the true SPN has been seriously contaminated and making reliable identification difficult. This is probably the reason why the best performance for Casio EX-150 can be achieved by Li's Enhancer [3], which deals with the scene details lying largely in the central area of the spectrum. Actually the performance on these 3 camera models provides a microcosm of the overall performance: SEA and ZM+WF are comparable for the reference SPN with a relatively smooth spectrum, but SEA is better than ZM+WF for the reference SPNs with a spectrum full of peaks, especially in the high-frequency band. Yet, there exist some unexpected artifacts that both ZM+WF and SEA cannot cope with effectively.

### 3.4.6 Running Time

Finally, the running times of different preprocessing schemes and detection statistics for different image sizes are listed in Table 3.9. We ran each configuration 1000 times and calculated the average running time. SEA spends extra time on finding the peaks within local neighborhood in the spectrum, as shown in Procedure 1, so it is reasonable to see that SEA requires more running time. But it takes less than half a second even for $1024 \times 1024$ pixels sized image blocks and only needs to be applied once on the reference SPN. In the scenario where only a few candidate cameras are

| | | Preprocessing | | | |
|---|---|---|---|---|---|
| | | Original | ZM | ZM+WF | SEA |
| NCC | Basic | 0.9517 | 0.9500 | 0.9467 | **0.9583** |
| | MLE | 0.9483 | 0.9483 | 0.9467 | **0.9583** |
| | Enhancer | 0.9517 | 0.9517 | 0.9500 | **0.9583** |
| | Phase | **0.9383** | 0.9317 | 0.9333 | 0.9350 |
| | BM3D | **0.9600** | 0.9550 | 0.9550 | **0.9600** |
| | PCAI8 | 0.9550 | 0.9533 | 0.9600 | **0.9633** |
| SPCE | Basic | 0.9500 | 0.9483 | 0.9483 | **0.9583** |
| | MLE | 0.9500 | 0.9483 | 0.9483 | **0.9583** |
| | Enhancer | 0.9517 | 0.9533 | 0.9500 | **0.9583** |
| | Phase | **0.9383** | 0.9317 | 0.9333 | 0.9350 |
| | BM3D | 0.9550 | 0.9600 | 0.9533 | **0.9600** |
| | PCAI8 | 0.9533 | 0.9617 | 0.9583 | **0.9633** |

Table 3.6: Kappa statistics for Nikon CoolPix on 256 × 256 image blocks

| | | Preprocessing | | | |
|---|---|---|---|---|---|
| | | Original | ZM | ZM+WF | SEA |
| NCC | Basic | 0.7733 | 0.7767 | 0.7367 | **0.7933** |
| | MLE | 0.7700 | 0.7833 | 0.7400 | **0.7967** |
| | Enhancer | 0.7700 | 0.7900 | 0.7633 | **0.8100** |
| | Phase | 0.7467 | **0.7633** | 0.7567 | 0.7533 |
| | BM3D | 0.7267 | 0.7533 | 0.7200 | **0.7567** |
| | PCAI8 | 0.7733 | 0.7600 | 0.7700 | **0.7767** |
| SPCE | Basic | 0.7700 | **0.7900** | 0.7500 | 0.7867 |
| | MLE | 0.7733 | 0.7867 | 0.7433 | **0.7933** |
| | Enhancer | 0.7833 | 0.7933 | 0.7700 | **0.8033** |
| | Phase | 0.7500 | **0.7567** | 0.7533 | 0.7533 |
| | BM3D | 0.7300 | **0.7600** | 0.7200 | 0.7567 |
| | PCAI8 | 0.6867 | 0.7667 | 0.7633 | **0.7833** |

Table 3.7: Kappa statistics for FujiFilm FinePix J50 on 256 × 256 image blocks

involved, the running speed is probably not the biggest concern. In addition, as can be seen in the last two rows of Table 3.9, NCC is faster than SPCE. So for

| | | Preprocessing | | | |
|---|---|---|---|---|---|
| | | Original | ZM | ZM+WF | SEA |
| NCC | Basic | 0.3333 | 0.3367 | **0.3400** | **0.3400** |
| | MLE | 0.3333 | 0.3400 | **0.3450** | 0.3383 |
| | Enhancer | 0.3550 | 0.3517 | 0.3550 | **0.3617** |
| | Phase | 0.3333 | 0.3283 | 0.3217 | **0.3367** |
| | BM3D | 0.3300 | 0.3250 | **0.3367** | 0.3333 |
| | PCAI8 | 0.3350 | 0.3217 | 0.3300 | **0.3367** |
| SPCE | Basic | 0.3333 | 0.3383 | 0.3400 | **0.3417** |
| | MLE | 0.3333 | 0.3433 | **0.3450** | 0.3383 |
| | Enhancer | 0.3567 | 0.3517 | 0.3533 | **0.3617** |
| | Phase | 0.3333 | 0.3283 | 0.3217 | **0.3367** |
| | BM3D | 0.3300 | 0.3267 | **0.3367** | 0.3333 |
| | PCAI8 | 0.3367 | 0.3217 | 0.3283 | **0.3383** |

Table 3.8: Kappa statistics for Casio EX-150 on $256 \times 256$ image blocks

large-scale SCI tasks, the odds of SEA can be evened up by choosing SEA+CNN rather than ZM+WF+SPCE.

| | Image sizes (pixels) | | |
|---|---|---|---|
| | $1024 \times 1024$ | $256 \times 256$ | $64 \times 64$ |
| ZM | 39.7 | 2.5 | 0.6 |
| ZM+WF | 154.2 | 8.4 | 1.6 |
| SEA | 493.0 | 55.2 | 39.5 |
| SPCE | 61.5 | 2.6 | 0.6 |
| NCC | 29.7 | 1.5 | 0.1 |

Table 3.9: Running time comparison (ms)

## 3.5 Conclusion

We have developed a novel SPN preprocessing approach, namely Spectrum Equalization Algorithm (SEA), for the task of SCI to overcome the limitations of existing approaches. The spectrum of the reference SPN is equalized by detecting and sup-

pressing the prominent peaks before calculating the similarity measurement with the query noise residual. Experimental results on the Dresden image database and our own database have confirmed the superiority of SEA in terms of both effectiveness and robustness against JPEG compression for medium and small sized images. We recognize that although only the task of SCI has been considered in this chapter, our work can be extended to the task of SPN-based image forgery detection, which is one of our future lines of investigation. As most existing methods dedicated to improving the performance of SCI only consider the interference coming from one particular source, such as the impact of the denoising filter, the periodic artifacts introduced by CFA interpolation and JPEG compression, the contamination from scene details, etc., an integrated approach for assembling the existing methods to provide superior performance is still lacking. It is not a simple and trivial task due to the possible interference among different methods, so this is another area to be studied in the future.

# CHAPTER 4

## Large-Scale Image Clustering Based on Camera Fingerprint

As discussed in Section 2.2 of Chapter 2, clustering large-scale device fingerprints is very challening due to the large-scale and high-dimentional nature of the problem. The difficulty can be further aggravated by the $NC \gg SC$ problem. In this chapter, we propose a novel clustering framework that is capable of addressing the $NC \gg SC$ issue without a training process. By reducing the dimensionality of camera fingerprints and exploiting the inherent sparseness of the pairwise similarity matrix, the original dataset is partitioned into a number of small batches using a fast graph clustering algorithm. Fine clustering is then applied on each of the small batches to produce a collection of sub-clusters, which will be further merged with the aid of the proposed adaptive thresholding technique. The centroids of the merged clusters serve as the attractors to attract the unclustered fingerprints to their closest centroids. The above procedures are iteratively repeated until no more notable clusters can be found. The proposed clustering framework is evaluated on the Dresden image database and compared with the state-of-the-art camera fingerprints clustering algorithms. Experimental results show that the proposed clustering framework is much faster than the state-of-the-art algorithms while maintaining a high level of clustering quality.

The remainder of this chapter is organized as follows. In the following Section 4.1, we will briefly introduce the limitations of existing device fingerprint clustering algorithms. In Section 4.2, the details of the proposed clustering framework will be given. Section 4.3 discusses the complexity of the algorithm as well as the parameter settings . Comprehensive experimental results and analysis will be presented in Section 4.4. Finally, Section 4.5 concludes the chapter.

## 4.1 Introduction

It can be observed in Chapter 2 that existing methods either initially cluster on a training set randomly sampled from the original dataset [57, 60, 61] or calculate a small portion of the pairwise similarities [58] to generate a number of representative clusters, the centroids of which will be used to classify the remaining fingerprints by assigning each of them to the most similar centroid. The successful classification requires that the whole dataset is well represented by the representative clusters. However, sometimes we are confronted with the $NC \gg SC$ problem, where the number of classes within the training set is probably less than that of the original dataset. The $NC \gg SC$ problem makes it difficult, if not impossible, to form a training set at random that can sufficiently represent the entire population. In consequence, misclassifications happen when some of the remaining fingerprints do not belong to any of the representative clusters. Li proposed in [57] that if the similarity between one fingerprint and the most similar centroid is less than a threshold set by the user, the fingerprint is treated as a new representative cluster. But not only the reliability of the new singleton representative cluster is doubtful, but also there is a probability that one fingerprint does not belong to any of the representative clusters while its similarity with the closest representative cluster is higher than the preset threshold. This usually happens when some fingerprints within one representative cluster are not purely from the same camera. What is worse, such kind of misclassification can be propagated in the succeeding classification process. Therefore, an effective way of determining an appropriate threshold is urgently needed.

In this chapter, we propose a novel clustering framework capable of dealing with large-scale camera fingerprint databases. It takes advantage of the dimension reduction technique and the inherent sparseness of the pairwise similarity matrix to reduce the computational cost. Based on the analysis of correlation distribution, we also derive an adaptive threshold with regard to the size and the quality of

clusters. It is the adaptive threshold that allows the clustering algorithm to work in a divide-and-conquer manner and makes the clustering on large-scale datasets much more efficient. Because the noise residue extracted from one image can be viewed as the noisy version of SPN, unless otherwise stated, we will refer to the true SPN components in the noise residue as the "true SPN" to distinguish it from the noise residue, which will be simply referred to as the "camera fingerprint" or "fingerprint" in the rest of this chapter. Additionally, to differentiate the ground truth and the clustering results, we refer to the fingerprints of the same camera as a *class*, while refer to those clustered into the same group by the clustering algorithm as a *cluster*.

## 4.2 Proposed Clustering Framework

Given the limitations of the existing methods reviewed in the previous section, we propose a new clustering framework as shown in Fig. 4.1. The proposed framework mainly consists of preparation, coarse clustering, fine clustering, attraction and post-processing. In what follows, we will look into each of the five steps to provide a rough picture of the proposed framework.

**Step 1:** Preparation

- **Image preprocessing:** Eliminate the images with saturation or low intensities. Rotate the remaining images to ensure they are all horizontally oriented.
- **Fingerprint extraction and standardization:** Suppose there are $n$ images left in the database after removing the dark or saturated images. A $d$-dimensional fingerprint is extracted from the center of the green channel of each image and standardized to zero mean and unit variance.
- **Dimension reduction:** Project each of the $d$-dimensional fingerprints onto a $k$-dimensional subspace ($k < d$) using the *very sparse random pro-*

*jection* proposed in [96].

- **Fingerprint data storage:** Store the full-length fingerprint, the corresponding dimension-reduced fingerprint as well as the image name into a single file for each image.

**Step 2:** Coarse clustering

- **Correlation matrix approximation:** To avoid calculating the $n \times n$ pairwise correlation matrix $\boldsymbol{M}$, the dimension-reduced fingerprints and the potential-based eviction (to be discussed in Section 4.2.2) are used to reduce the computational cost. In this way, $\boldsymbol{M}$ is replaced with a set of more RAM-efficient lists $\boldsymbol{L}$, of which each element $\boldsymbol{L}_i$ records the indices of the fingerprints that are similar enough to the $i$th fingerprint.
- **Coarse partition:** Based on the graph information recorded in $\boldsymbol{L}$, the $n$ fingerprints are coarsely partitioned into $n_c$ coarse clusters using the fast graph clustering algorithm in [97].

**Step 3:** Fine clustering

- **Cluster splitting:** For each of the $n_c$ coarse clusters, a correlation matrix is calculated and binarized as in the coarse clustering stage, but using the $d$-dimensional full-length fingerprints. Because the size of each coarse cluster is small, the calculation of the correlation matrix is fast and efficient. Based on the binary correlation matrix corresponding to one coarse cluster, the cluster is naturally split into $n_s$ sub-clusters with variable sizes using the flow simulation based graph clustering algorithm [98].
- **Cluster merging:** Each of the $n_s$ sub-clusters is represented by the centroid averaged over the full-length member fingerprints. Based on the $n_s$ centroids, a $n_s \times n_s$ pairwise correlation matrix is calculated and binarized using an adaptive threshold matrix $\boldsymbol{\tau}$ (the element $\tau_{ij}$ depends on the sizes

and the qualities of sub-cluster $S_i$ and sub-cluster $S_j$). Larger clusters are therefore obtained by merging two or several sub-clusters. Meanwhile, the centroids and the qualities of the merged clusters are updated accordingly.

**Step 4:** Attraction

- **Centroid attraction:** The centroids are used as "attractors" to attract the unclustered fingerprints left in the database. Consequently, most of the fingerprints belonging to the discovered clusters will be absorbed to form larger clusters.

**Step 5:** Post-processing

- **Cluster screening and storage:** Place the fingerprints in the clusters with a size smaller than a threshold $\eta$ back into the database and store the remaining clusters as the final clusters. The stored information includes the centroid, the names of the corresponding images, and the quality of the cluster.
- **Termination or continuation:** The algorithm ends if no more notable clusters can be found. Otherwise, place the unclustered fingerprints, as well as those in the clusters with a size smaller than $\eta$, back into the database for the next round of clustering.

In the following subsections, we will provide more details on each of the above steps.

### 4.2.1 Preparation

Since the fingerprint information is not present in the dark or saturated images [9], involving these images into the clustering process not only entails high computational burden, but also makes the clustering results unreliable and unpredictable.

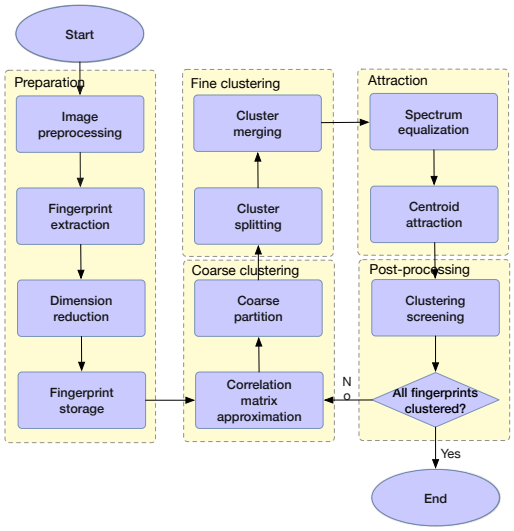



Figure 4.1: Flow chart of the proposed framework.

Hence, those images with insufficient fingerprint information will be ignored. Another concern comes from the orientation of images. As we know, SPN is pixel-dependent and images taken by the same camera can be in either the horizontal or vertical orientation depending on how the camera is held. Inconsistent orientation will desynchronize the pixel-to-pixel correspondence between images taken by the same camera. For JPEG/TIFF images, the orientation information may have been stored in the exchangeable image file format (EXIF) header of the image file, but the EXIF information is untrustworthy and sometimes unavailable. We therefore rotate the vertically oriented images clockwise by 90° to ensure all images are horizontally oriented. Although this cannot completely solve the rotation problem, it successfully limits the freedom of rotation.

After the above preprocessing, a camera fingerprint $\boldsymbol{F}$ is extracted from the central block of the green channel of each image $\boldsymbol{I}$, i.e.,

$$\boldsymbol{F} = \boldsymbol{I} - \mathcal{G}(\boldsymbol{I}), \tag{4.1}$$

where $\mathcal{G}$ is the denoising filter. The normalized cross correlation (NCC) $\rho$ is used as

the similarity measurement between two fingerprints, $\boldsymbol{X}$ and $\boldsymbol{Y}$:

$$\rho(\boldsymbol{X},\boldsymbol{Y}) = \frac{\sum_{i=1}^{d}(X[i]-\overline{X})(Y[i]-\overline{Y})}{\sqrt{\sum_{i=1}^{d}(X[i]-\overline{X})^2}\sqrt{\sum_{i=1}^{d}(Y[i]-\overline{Y})^2}}, \tag{4.2}$$

where $d$ is the length of the fingerprint, $\overline{X}$ and $\overline{Y}$ are the arithmetic mean of $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively. NCC is notorious for its high computational cost due to the calculation of the variances in the denominator of Equation (4.2). To alleviate this, each fingerprint is standardized to have zero mean and unit variance. After standardization, Equation (4.2) is simplified to the more efficient element-wise product:

$$\rho(\boldsymbol{X},\boldsymbol{Y}) = \frac{1}{d}\sum_{i=1}^{d}\hat{X}[i]\hat{Y}[i], \tag{4.3}$$

where $\hat{\boldsymbol{X}}$ and $\hat{\boldsymbol{Y}}$ are the standardized version of $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively.

Standardization also builds up the equivalence between NCC and other commonly-used similarity measures such as the inner product[1] and the Euclidean distance [99]. It reminds us that many similarity-preserving dimension reduction methods are potentially suited for use in reducing the computational cost. Unfortunately, dimension reduction techniques based on the discrete Fourier transform (DFT) [100], singular value decomposition (SVD) [101], discrete wavelets transform (DWT) [102] and piecewise aggregate approximation (PAA) [103] are not applicable to our application due to the nature of noise-like camera fingerprints. More advanced methods, such as principal components analysis (PCA) [104], isometry mapping [105], maximal variance unfolding [106] and locally linear embedding [107] are computationally infeasible for large-scale camera fingerprints due to the mutual dependencies of data. We therefore resort to another more efficient solution — *very sparse random projection* [96, 108] due to its appealing properties of Euclidean distance-preserving, computational efficiency and data independence. Let $d \times n$ matrix $\boldsymbol{D}$ be

[1] $\rho(\boldsymbol{X},\boldsymbol{Y}) = \langle \frac{\hat{X}}{\sqrt{d}}, \frac{\hat{Y}}{\sqrt{d}} \rangle$

the set of $n$ camera fingerprints in the $d$-dimensional space $\mathbb{R}^d$, and $\boldsymbol{R}$ be the $k \times d$ random projection matrix with $\boldsymbol{R}(i,j)=r_{ij}$, where $\{r_{ij}|1 \leq i \leq k, 1 \leq j \leq d\}$ are independent and identically distributed (i.i.d.) random variables drawn from the following probability distribution:

$$r_{ij} = \sqrt{\frac{s}{k}} \times \begin{cases} +1 & \text{with probability} \quad 1/2s, \\ 0 & \text{with probability} \quad 1 - 1/s, \\ -1 & \text{with probability} \quad 1/2s, \end{cases} \tag{4.4}$$

where $s=\sqrt{d}$ according to [96]. The dimension reduction of the $n$ fingerprints from $\mathbb{R}^d$ to $\mathbb{R}^k$ is achieved by a matrix multiplication:

$$\boldsymbol{E} = \boldsymbol{RD}, \tag{4.5}$$

where $\boldsymbol{E}$ is a $k \times n$ matrix, with its columns representing the dimension-reduced camera fingerprints.

Finally, the standardized fingerprint, the dimension-reduced counterpart, along with the image name are stored in a file with an unique filename. The dimension-reduced and the full-length fingerprint will be used for the following coarse clustering and fine clustering, respectively.

### 4.2.2 Coarse clustering

One of the key challenges of clustering large-scale and high-dimensional camera fingerprints is to discover the potentially correlated fingerprint pairs. Calculating the complete, exhaustive set of similarities in the $n \times n$ pairwise correlation matrix $\boldsymbol{M}$ requires at least $n(n-1)/2$ comparisons, which becomes clearly prohibitive for large databases. Even if we have the oracle to know all of the entries of $\boldsymbol{M}$, it is still an issue to cache them in RAM. Practically speaking, the correlation matrix is very

likely to be sparse because only the fingerprints of the same camera are correlated with each other. So instead of keeping all the correlations in the RAM, we use a threshold $t_b$ to binarize the similarities (correlations). Since the intrinsic quality of fingerprints depends on many complex factors, the average similarities for different cameras may vary substantially. But in the sense of clustering, one class with a higher average intra-class similarity should be equivalent to another class with a lower average similarity. In this regard, binarization is an effective way to eliminate the divergences in different classes and make the clustering more resilient against the interferences in the fingerprints.

Another optimization is to calculate a small portion of the $n(n-1)/2$ matrix entries, then the key issue becomes what heuristics or criteria can be used to select the matrix entries for calculation. One such work was proposed in [58], but, as mentioned in Section 2.2.2, the expensive I/O cost makes it unaffordable for large-scale databases. Given $n$ dimension-reduced fingerprints, we randomly partition them into batches of equal size $q$, where $q$ can be customized so that the RAM is sufficient for simultaneously accommodating two batches and the extra space for calculating the inter-batch correlations. Two batches are firstly brought into the RAM, and the correlations of the fingerprints in the two batches are calculated and binarized for updating a set of lists $\boldsymbol{L}$, of which each element $\boldsymbol{L}_i$ is a list recording the indices of the vertices adjacent to the $i$th fingerprint. All remaining batches are sequentially loaded from the disk one at a time. To proceed with the next batch, at least $q$ fingerprints have to be evicted from the RAM. We consider each dimension-reduced fingerprint as a vertex $v_i, i = 1, 2, ..., n$ in a graph $G$, and define a *potential* measurement $p_i$ characterizing its potential connectivities with other vertices:

$$p_i = \frac{\deg(v_i)}{A_i}, \tag{4.6}$$

where $\deg(v_i)$ is the *degree* of vertex $v_i$ (i.e., the length of $\boldsymbol{L}_i$) and $A_i$ is the accumu-

lated number of vertices whose connectivities with vertex $v_i$ have been investigated up to the latest batch. $p_i \in [0, 1]$ is initialized as 0 at the beginning. To accommodate the new incoming batch, the $q$ vertices with the lowest *potentials* are evicted from the RAM. We refer to this strategy as the *potential-based eviction*. The underlying motivation is that *the more fingerprints a camera fingerprint is connected with among those that have been investigated, the more likely that it will be connected with more fingerprints among the ones that have not been investigated yet.* By taking advantage of the asymmetry of the class distribution, the potential-based eviction allows for discovering more correlated fingerprint pairs based on the limited number of correlation calculations. Meanwhile, the I/O cost is minimized because every dimension-reduced fingerprint is loaded only once. It is worth noting that we use $p_i$ instead of $\deg(v_i)$ because the number of pairwise comparisons for the vertices in the posterior batches is less than that for the vertices in the previous batches. As a consequence, the vertices loaded earlier tend to have a higher *degree* than the ones loaded later.

After obtaining the potentially correlated fingerprint pairs, we aim to roughly but efficiently partition the database into groups. Many graph partitioning algorithms [97, 109–111] are feasible for the task. In our work, we chose a highly scalable and open-source[2] algorithm, GRACLUS [97], which works by repeatedly coarsening the original graph (corresponding to $\boldsymbol{L}$) to a point where very few vertices remain in the graph, then a spectral clustering method [112] is applied on the coarsest graph, the clustering result of which will be refined level by level up to the original graph. By empirically specifying the cluster number $n_c = \lceil n^{\frac{1}{4}} \rceil$, where $\lceil \cdot \rceil$ is a ceiling operator, GRACLUS partitions the $n$ fingerprints into $n_c$ coarse clusters with various sizes, among which some large coarse clusters will be recursively bisected into small clusters to ensure that each of them can be fit into the RAM at a time. The purpose of coarse clustering is to gather potentially correlated fingerprints into

[2]The source code can be downloaded from http://www.cs.utexas.edu/users/dml/Software/graclus.html

the same batch and therefore increase the probability of forming reliable clusters in the following fine clustering stage.

### 4.2.3 Fine clustering

After coarsely partitioning the whole database, the coarse clusters will be further split and merged in the fine clustering stage. Specifically, for each of the resultant coarse clusters, an accurate correlation matrix is calculated and binarized as in the coarse clustering stage, but using the full-length ($d$-dimensional) rather than the dimension-reduced ($k$-dimensional) fingerprints. Because the number of fingerprints in each coarse cluster is small, the processing of each coarse cluster can be quite efficient. The binarized correlation matrix is input to the Markov cluster algorithm (MCL) [98], which iteratively applies the expansion (matrix multiplication) and inflation (entry-wise matrix product) operators until it converges to a steady state. Finally, the clusters are discovered by interpreting the resultant matrix. Given a small enough coarse cluster, many alternative algorithms, such as [57, 58, 60, 61, 113], are feasible for the clustering task. But the following merits make MCL preferable to other methods:

1. MCL can be significantly sped up by taking advantage of the sparseness of the correlation matrix and pruning the small matrix entries during each iteration.

2. MCL tends to result in small cluster granularity but high precision rate. Given a large volume of fingerprints, it is difficult to achieve high recall rate and high precision rate simultaneously. Mixing the fingerprints of different cameras in the same cluster (low precision rate) can lead to error propagation in the ensuing clustering process. However, dispersing the fingerprints of the same camera into several clusters (low recall rate) still ensures high correctness and reliability of the clustering despite the extra computational cost. Therefore, if we have to sacrifice in one aspect to gain in the other, we would usually prefer

the high precision rate to the high recall rate in practice.

The split of coarse clusters may produce many small or even singleton sub-clusters, but only the sub-clusters with a score $\xi$ greater than a predefined threshold $t_s$ are collected for further use:

$$\xi = \sqrt{|C|}\frac{\sum_{v_i \in C}\sum_{v_j \in C} e_{ij}}{|C|(|C|-1)} > t_s, \tag{4.7}$$

where $|C|$ is the size of cluster $C$, $e_{ij}$ is a binary variable with 1 signifying that $v_i$ is connected with $v_j$. But due to the limited RAM size, sub-clusters are sorted according to the score $\xi$ and only the $\mathcal{H}$ sub-clusters with the highest scores are retained in RAM, where $\mathcal{H}$ is adaptive to the RAM size. We refer to this strategy as *score sorting*. In such a manner, we can make better use of the limited RAM and guarantee that the larger classes (usually related to the sub-clusters with a higher score) will be clustered preferentially.

In the coarse clustering stage, the fingerprints of the same camera are likely to be partitioned into different coarse clusters. Even if those fingerprints are grouped into the same coarse cluster, they may still be split into different sub-clusters in the ensuing splitting stage, so it is desirable to merge the sub-clusters of the same camera for both efficiency and accuracy reasons before proceeding further. Hereafter, we will refer to the correlation between the centroids of two clusters from the same camera as the *intra-class* correlation, and the correlation between the centroids of two clusters from different cameras as the *inter-class* correlation. Intuitively, the intra-class correlation increases with the cluster size because the random noises in the centroid of a larger cluster have been suppressed more significantly, while the inter-class correlation remains the same. If we know how the correlation between two centroids changes with the sizes of clusters, an adaptive threshold can be adopted to determine whether they are from the same camera. The adaptive thresholding problem for camera fingerprint clusters merging has been studied in [114] and [115]

by estimating the parameters of a prescribed function, but both of them do not generalize well across different cameras. According to the Central Limit Theorem (CLT), the NCC $\rho$ between two $d$-dimensional centroids, $\boldsymbol{X}$ and $\boldsymbol{Y}$, from different cameras conforms to a normal distribution with zero mean and $1/d$ variance, i.e., $\rho(\boldsymbol{X},\boldsymbol{Y})\sim\mathcal{N}(0,1/d)$. If $\boldsymbol{X}$ and $\boldsymbol{Y}$ are the centroids of two sub-clusters $S_x$ and $S_y$ from the same camera, we can derive that the distribution approaches to a normal distribution when $d\to\infty$ (please refer to the Derivation of Correlation Distribution in the Appendix), i.e.,

$$\rho(\boldsymbol{X},\boldsymbol{Y}) \xrightarrow{d} \mathcal{N}(\mu,\Sigma), \tag{4.8}$$

where

$$\begin{cases} \mu = \sqrt{\frac{n_x n_y \sigma_x^2 \sigma_y^2}{[(n_x-1)\sigma_x^2+1][(n_y-1)\sigma_y^2+1]}} \\ \Sigma = \frac{n_x n_y \sigma_x^2 \sigma_y^2+[(n_x-1)\sigma_x^2+1][(n_y-1)\sigma_y^2+1]}{[(n_x-1)\sigma_x^2+1][(n_y-1)\sigma_y^2+1]d}. \end{cases} \tag{4.9}$$

Here, $d$ is the length of fingerprints. $n_x$ and $n_y$ are the sizes (i.e., numbers of member fingerprints) of cluster $S_x$ and $S_y$, respectively. $\sigma_x^2$ and $\sigma_y^2$ are the average qualities of the true SPN in each member fingerprint in $S_x$ and $S_y$. When there is no ambiguity, the average quality of the true SPN in each member fingerprint of one cluster will be simply referred to as the quality of the cluster. For large clusters, the mean of the intra-class distribution is normally far from the zero mean of the inter-class distribution, so the threshold can be safely increased to reduce the false positives. An adaptive threshold $\tau$ characterizing the change in the intra-class distribution is therefore proposed as

$$\tau = \max\Big(t_b, \frac{\omega\sqrt{n_x n_y \sigma_x^2 \sigma_y^2}}{\sqrt{[(n_x-1)\sigma_x^2+1][(n_y-1)\sigma_y^2+1]}}\Big), \tag{4.10}$$

where $t_b$ is the threshold used for binarizing the similarity matrix in the fine clustering stage, $\omega$ is a predefined scaling factor. Two sub-clusters with a correlation between their centroids higher than $\tau$ are considered to be from the same camera.

In Equation (4.10), $\tau$ is related to the dynamic intra-class distribution rather than the constant inter-class distribution, so it is more reliable than the threshold determined by the Neyman-Pearson criterion when the cluster size keeps increasing. In such a way, $\tau$ effectively prevents the cluster merging from propagating errors into the merged clusters. $\sigma_x^2$ and $\sigma_y^2$ in Equation (4.10) can be estimated by calculating the mean of correlations (please refer to $\mu_1$ in Equation (A.6) with $\lambda$=1 in the Appendix). Therefore, when coarse clusters are split into $n_s$ sub-clusters, the quality of each sub-cluster is initially estimated as

$$\sigma_i^2 = \frac{1}{q_i} \sum_{k=1}^{q_i} \rho_k, i = 1, 2, \ldots, n_s, \tag{4.11}$$

where $q_i$ is the number of the calculated correlations within $S_i$. Following the analysis in Scenario 2 in the Appendix, if $S_x$ and $S_y$ are merged into $\hat{S}$ in the cluster merging stage, we update the quality of $\hat{S}$ as

$$\hat{\sigma}^2 = \frac{n_x \sigma_x^2 + n_y \sigma_y^2}{n_x + n_y}. \tag{4.12}$$

In particular, when $\sigma_x^2$=$\sigma_y^2$=$\sigma^2$, $\hat{\sigma}^2$ of the merged cluster $\hat{S}$ is the same as that of $S_x$ or $S_y$, i.e., $\hat{\sigma}^2$=$\sigma^2$. Equation (4.12) can be easily generalized for merging $c$ sub-clusters:

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{c} n_i \sigma_i^2}{\sum_{i=1}^{c} n_i}, \tag{4.13}$$

where $n_i$ and $\sigma_i^2$ are the size and the quality of $S_i$, respectively.

### 4.2.4 Attraction

The following stage is the centroid attraction, where the centroids of the merged clusters are used as the "attractors" to attract the fingerprints remaining in the database, so that the fingerprints belonging to the same cluster will be absorbed into the corresponding attractor. Specifically, one fingerprint $\boldsymbol{F}$ belongs to attractor

$\boldsymbol{A}^*$ if

$$\boldsymbol{A}^* = \underset{\boldsymbol{A}_i \in \mathcal{A}}{\operatorname{argmax}} \rho(\boldsymbol{A}_i, \boldsymbol{F}), \rho(\boldsymbol{A}^*, \boldsymbol{F}) > \tau_\rho, \tag{4.14}$$

where $\mathcal{A}$ is the set of attractors and $\tau_\rho$ is calculated from Equation (4.10). Finally, the attracted fingerprints belonging to the same cluster are merged into the corresponding cluster and the quality of the merged cluster is updated accordingly. Suppose an attractor $\boldsymbol{A}_x$ is averaged over $n_x$ fingerprints with an average quality $\sigma_x^2$ of the true SPN in each fingerprint, then according to $\mu$ in Equation (4.9), the average quality $\sigma_y^2$ of the $n_y$ attracted fingerprints can be estimated as

$$\sigma_y^2 = \frac{[(n_x - 1)\sigma_x^2 + 1] \sum_{k=1}^{n_y} \rho_k^2}{n_x n_y \sigma_x^2}. \tag{4.15}$$

The set of fingerprints corresponding to $\boldsymbol{A}_x$ and the set of $n_y$ attracted fingerprints can be viewed as two clusters with a quality of $\sigma_x^2$ and $\sigma_y^2$, respectively. So according to Equation (4.12), the quality of the attractor $\boldsymbol{A}_x$ and the number of member fingerprints are updated as

$$\begin{cases} \sigma_x^2 \leftarrow \dfrac{n_x^2 \sigma_x^4 + [(n_x - 1)\sigma_x^2 + 1] \sum_{k=1}^{n_y} \rho_k^2}{n_x (n_x + n_y) \sigma_x^2} \\ n_x \leftarrow n_x + n_y. \end{cases} \tag{4.16}$$

### 4.2.5 Post-processing

Up until now, the clustering process has formed a certain number of clusters, each of which is presented by a centroid, the names of the corresponding images, and the quality of the cluster. Notable clusters with a size no smaller than $\eta$ are stored as the final clusters. While those clusters with a size less than $\eta$, as well as the remaining unclustered fingerprints, are placed back to the database for the next round of iteration starting from the coarse clustering stage. Due to the nature

of the potential-based eviction in the coarse clustering stage and the score sorting in the fine clustering stage, the classes with a larger size are more likely to form notable clusters, so the majority of the fingerprints can be clustered in the first few iterations. During the coarse clustering stage of the next iteration, we put the unclustered fingerprints from the same batch in the previous iteration into different batches so as to increase the chance of discovering correlated fingerprint pairs. The algorithm terminates when no more notable clusters can be discovered.

## 4.3 Discussion

After presenting the algorithm as above, the reasons why the proposed clustering framework can cope with large-scale camera fingerprint databases become clear. By taking advantage of the dimension reduction technique and the sparseness of the pairwise similarity matrix, the clustering problem of a large database is broken down into the clustering of several smaller databases with the help of the fast, but approximate, graph partition algorithm. The adaptive thresholding significantly reduces the computational complexity and allows the clustering results of the smaller databases to be combined to give the solution to the $NC \gg SC$ problem. The ability of spotting small classes is conferred by the iterative clustering manner and the adaptive value of coarse cluster number $n_c = \lceil n^{\frac{1}{4}} \rceil$. On one hand, the iterative manner (thanks to the potential-based eviction and the score sorting strategy) ensures that most of the larger classes will be clustered in the first few iterations and the smaller classes will be more focused on in the ensuing iterations. On the other hand, with the decreasing number of coarse clusters, the probability that more fingerprints from smaller classes fall into the same coarse cluster increases, making them more easily to be discovered in the fine clustering stage.

The time complexity of the proposed framework depends on a complex interplay between the number of fingerprints $n$, the length of the fingerprints, the

class distribution in the dataset, the number of edges $|E|$, and the parameter settings. In the coarse clustering stage, it involves $nqk$ multiplications to calculate the sparse approximate correlation matrix, where $q$ is the size of one batch and $k$ is the reduced dimension. The fast graph partitioning algorithm GRACLUS [97] approximately has a time complexity $\mathcal{O}(q|E|/n)$. In the fine clustering stage, it involves around $nbd$ multiplications to calculate the accurate correlation matrix, where $b$ is the average size of the coarse clusters and usually much smaller than $q$, and $d$ is the length of the full-length fingerprint. The MCL has a time complexity of $\mathcal{O}(nK^2)$ in the worst case [98], where $K$ is the maximal number of nonzero entries in one column of the binarized correlation matrix. Finally, the attraction stage involves at most $ncd\theta$ multiplications, where $\theta \in (0, 1]$ is a factor accounting for the percentage of the $c$ classes that have been discovered. Considering the very high dimension of the fingerprints, the time complexity $\mathcal{O}(q|E|/n)$ of GRACLUS is negligible because even for an extremely tight and dense graph, $\mathcal{O}(q|E|/n) = \mathcal{O}(nq)$ is trivial when compared with the time complexity $\mathcal{O}(nqk + nbd)$. So the overall time complexity of the proposed clustering framework is approximately $\mathcal{O}(nqk + nbd + nK^2 + ncd\theta)$ in one iteration. $q$ and $b$ can be fixed or made adapted to the RAM size, while $K$, $c$ and $\theta$ depend on the class distribution of the dataset. With regard to the I/O cost of loading fingerprints from disk, it is $\mathcal{O}(2nd + nk)$ in the worst case. At the first glance, it is about two times as high as the minimal I/O cost $\mathcal{O}(nd)$. But given that one fingerprint needs to be loaded more than once to calculate the pairwise correlations for large datasets due to the limited size of RAM, $\mathcal{O}(2nd + nk)$ is still at an acceptable level and a considerable speedup can be expected by using a faster storage medium, such as Solid State Drive (SSD).

## 4.4 Experiments

### 4.4.1 Experimental setup

The proposed clustering framework was evaluated on the Dresden image database [53, 91]. After the removal of the dark and saturated images, involved in the experiments are $15,840$ images taken by 74 cameras, covering 27 models and 14 brands. The number of images taken by each camera varies from 154 to 460. All experiments were conducted on a PC with windows 7 OS, 3.2 GHz Intel Quad Core Processor, 16 GB RAM and 1 TB Hard Disk Drive (HDD). To alleviate the vignetting effects [94] and ensure a consistent size for all fingerprints, camera fingerprints were extracted from the central $1024 \times 1024$ block of the green channel of the full resolution images (i.e., $d$=$1,048,576$). The method proposed in [1] was used to extract the camera fingerprints, which were further preprocessed by two operations, zero-meaning (ZM) and Wiener filtering (WF) in the DFT domain [9], to suppress the non-unique artifacts. Note that we did not use the spectrum equalization algorithm (SEA) proposed in Chapter 3. The reason is that SEA suppresses the periodic artifacts by removing the peaks in the spectrum. However, unlike in the spectrum of the reference SPN (i.e., the average of multiple fingerprints), the peaks caused by periodic artifacts are not conspicuous in the spectrum of a single fingerprint.

The proposed framework is designed for clustering large-scale image database, but we are also interested in its performance on small datasets. Therefore, based on the Dresden database, we set up four different small datasets. As we know, images taken by different devices of the same model undergo the same or similar in-camera processing procedures, so it is more challenging to correctly cluster the fingerprints of the devices of the same model. We categorize the clustering difficulties into *model-* and *device-* levels. In the model-level, images in different classes are acquired by devices of different models, while in the device-level, some images are taken by devices of the same model. Moreover, it is common in practical applications that the num-

bers of images captured by different devices vary widely, which results in different class distributions within the dataset. We, therefore, categorize the distributions of images in different classes into *symmetric* and *asymmetric*. Finally, we set up the following four datasets for the experiments:

- $\mathcal{D}_1$: Model-level symmetric dataset. It consists of $1,000$ images taken by 25 cameras (each responsible for 40 images). The 25 cameras are of different models and cover nearly all of the popular camera brands, such as Canon, Nikon, Olympus, Pentax, Samsung and Sony.

- $\mathcal{D}_2$: Model-level asymmetric dataset. 20, 30, 40, 50 and 60 images were alternatively chosen from the images taken by the same 25 cameras as in $\mathcal{D}_1$.

- $\mathcal{D}_3$: Device-level symmetric dataset. It consists of $1,000$ images taken by 50 cameras (each responsible for 20 images). The 50 cameras only cover 12 popular models, so some of them are from the same model.

- $\mathcal{D}_4$: Device-level asymmetric dataset. 10, 15, 20, 25 and 30 images were alternatively chosen from the images taken by the same 50 cameras as in $\mathcal{D}_3$.

Notice that the size of each dataset was fixed to $1,000$, while the number of classes, as well as the ratio of the number of classes to the average size of classes, was raised in $\mathcal{D}_3$ and $\mathcal{D}_4$. These four datasets will be used for investigating parameters, evaluating the capability of the proposed framework in conquering the $NC \gg SC$ problem, and comparing the performances of different clustering algorithms on small datasets. The results on the entire Dresden database can be found in the last part of Section 4.4.3.

The quality of clustering is characterized in terms of the F1-measure

$$\mathcal{F} = 2 \cdot \frac{\mathcal{P} \cdot \mathcal{R}}{\mathcal{P} + \mathcal{R}}, \tag{4.17}$$

where the average precision rate $\mathcal{P}$ and the average recall rate $\mathcal{R}$ are calculated as

$$\begin{cases} \mathcal{P} = \dfrac{\sum_i |o_i|}{\sum_i |c_i|} \\ \mathcal{R} = \dfrac{\sum_i |o_i|}{\sum_i |\psi_i|}, \end{cases} \tag{4.18}$$

where $|c_i|$ is the size of cluster $c_i$, $|\psi_i|$ is the size of the most frequent class $\psi_i$ in cluster $c_i$, and $|o_i|$ is the size of the overlap between cluster $c_i$ and class $\psi_i$.

### 4.4.2 Parameter settings

There are a few parameters that need to be set for our proposed framework. We will investigate the impact of these parameters on performance and discuss how to determine the appropriate parameter settings.

The first parameter is the dimension $k$ of the subspace of random projection. As remarked in [96], we can apply, with a high level of accuracy, the results of *conventional random projection*, i.e., the i.i.d. entries of $\boldsymbol{R}$ are drawn from the standard normal distribution $\mathcal{N}(0, 1)$, to *very sparse random projection*. For example, we can determine the minimum $k$ that achieves an embedding error $\epsilon$ in pairwise correlation preservation using *Theorem 4*[3] in [116]. But this gives a very conservative estimation of the minimum $k$. We drew the distributions of $\epsilon$ for different $k$ using the camera fingerprints in dataset $\mathcal{D}_1$ ($n = 1,000$) and $\mathcal{D}_3$ ($n = 1,000$), as shown in Fig. 4.2(a) and Fig. 4.2(b), respectively. According to *Theorem 4* in [116], if we want to preserve 80% of the pairwise correlations with an error less than $\epsilon = 0.005$ for a dataset consisting of $n = 1,000$ points, the required minimum $k$ has to be higher than $2.8 \times 10^6$. But as can be seen in Fig. 4.2, the same level of accuracy can be achieved using $k = 65,536$, which is used in all of our experiments.

The second parameter is the binarization threshold $t_b$. For binarizing the accurate correlation matrix in the fine clustering stage, we set $t_b = 0.005$ to give a

[3]It states that let $Q$ be a set of $n$ unit-normal points in $\mathbb{R}^d$ and $\boldsymbol{R}$ be a random matrix with i.i.d. entries drawn from $\mathcal{N}(0, 1)$. Then for all $\boldsymbol{u}, \boldsymbol{v} \in Q$, $\Pr(|\langle \boldsymbol{u}, \boldsymbol{v} \rangle - \langle \boldsymbol{Ru}, \boldsymbol{Rv} \rangle| \geq \epsilon) \leq n^2 e^{(2-(\epsilon^2-\epsilon^3)\frac{k}{4})}$.

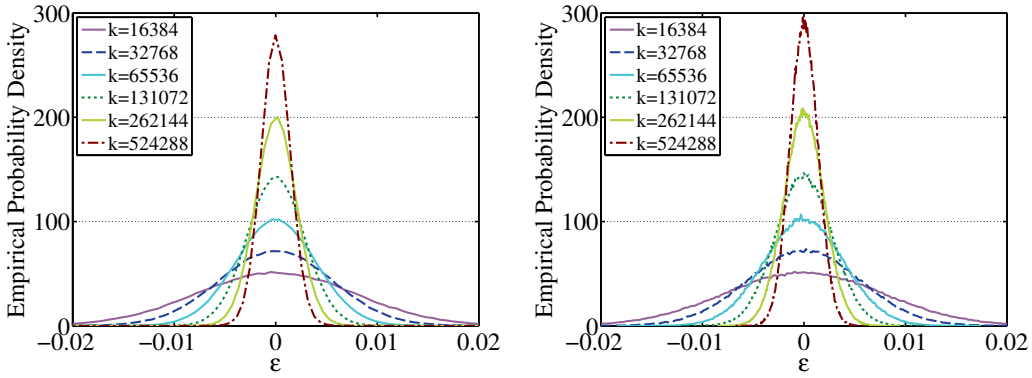



Figure 4.2: Probability density functions of the embedding error $\epsilon$ in pairwise correlations of $\mathcal{D}_1$ (a) and $\mathcal{D}_3$ (b).

theoretical false positive rate $\mathcal{Q}(d * t_b) \approx 1.5 \times 10^{-7}$, where $\mathcal{Q}$ is the complementary cumulative density function of the standard normal distribution $\mathcal{N}(0, 1)$. However, the actual false positive rate can be much higher due to the presence of non-unique artifacts. To see this, we obtained the ROC curves by varying a threshold from $-1$ to 1 and comparing it with the pairwise correlations of camera fingerprints in $\mathcal{D}_1$ and $\mathcal{D}_3$, as shown in Fig. 4.3(a) and Fig. 4.3(b), respectively. The false positive rate is approximately $1 \times 10^{-4}$ for the model-level dataset $\mathcal{D}_1$ and can be even higher for the device-level dataset $\mathcal{D}_3$. To compensate for the errors introduced by random projection, we increased $t_b$ to 0.008 for binarizing the entries of the approximate correlation matrix in the coarse clustering stage. As shown in Fig. 4.3, $t_b = 0.008$ gives a false positive rate around $2.5 \times 10^{-2}$. Note that the much larger false positive rate in the coarse clustering does not necessarily result in a low precision rate of the final clustering result, because the coarse clustering is followed by the accurate fine clustering. To give a practical reference, the curves corresponding to thresholds in $[0.004, 0.006]$ for the original fingerprint and thresholds in $[0.006, 0.01]$ for the dimension-reduced fingerprint are highlighted in yellow.

The third parameter is the scaling factor $\omega$ of the adaptive threshold $\tau$ in Equation (4.10). It actually determines a point that lies between the means of intra-

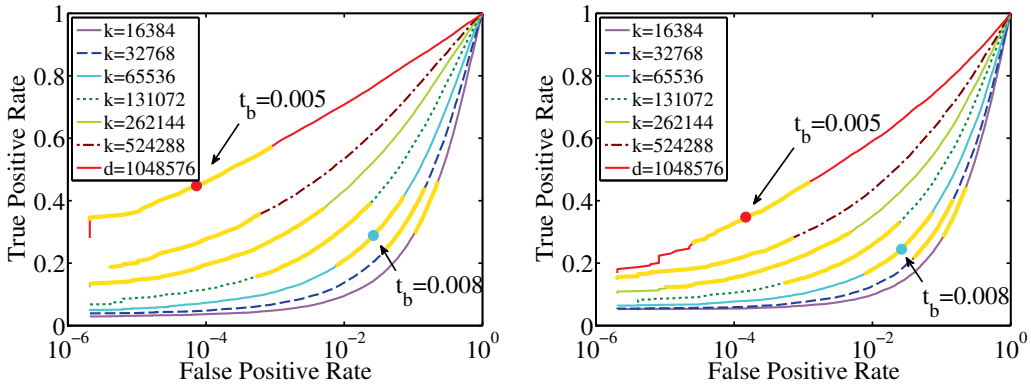
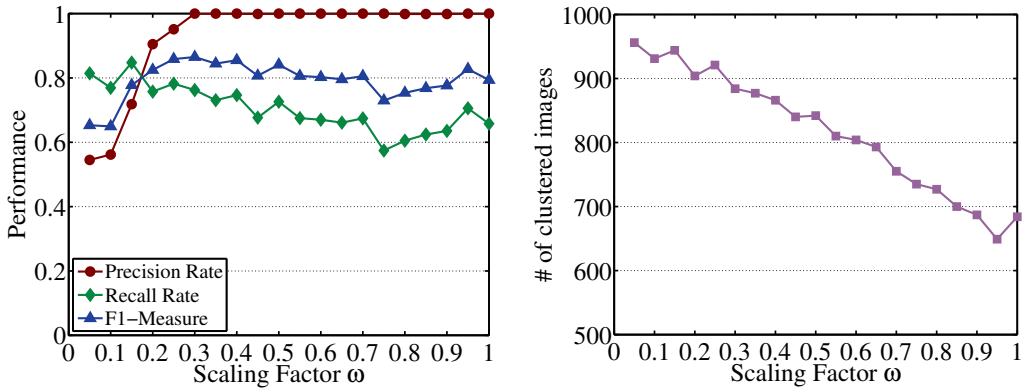



Figure 4.3: ROC curves obtained by varying a threshold from $-1$ to 1 and comparing it to the pairwise correlations of camera fingerprints in dataset $\mathcal{D}_1$ (a) and $\mathcal{D}_3$ (b).

class and inter-class distributions. To see how $\omega$ affects the clustering results, we conduct experiments on dataset $\mathcal{D}_1$. As shown in Fig. 4.4(a), a high $\omega$ reduces the false attribution error, and therefore gives rise to the precision rate. But the drawback of a high $\omega$ is that it produces many small clusters with a size less than $\eta$, which excludes a large amount of images from the final results. As can be seen in Fig. 4.4(b), the number of clustered images (i.e., the images in clusters with a size larger than $\eta$) decreases as $\omega$ goes up. We found that an $\omega$ in the range of $[0.3, 0.5]$ strikes a good balance between the performance and the number of clustered images. $\omega$ is set to 0.45 in our experiments.

The fourth parameter is the score threshold $t_s$ in Equation (4.7). Only the sub-clusters with a score larger than $t_s$ are collected for further use in the merge and attraction stages. Since the score $\xi$ is a function of the cluster size, $t_s$ is therefore closely related with the minimal cluster size $\eta$. The effect of these two parameters on the performance will be investigated jointly. As could be expected, these two parameters are more sensitive to datasets with small average class size. Therefore, we clustered the challenging dataset $\mathcal{D}_3$ (with a small average class size of 20) using different combinations of $t_s \in [1, 3]$ and $\eta \in [2, 10]$. As can be seen in Fig. 4.5, the




(a) (b)

Figure 4.4: Impact of $\omega$ on (a) the clustering performance and (b) the number of images included in the final results.

larger $t_s$, the better clustering performance. But the trade-off is that a higher $t_s$ makes small sub-clusters less likely to be collected for further investigation, which limits the capability of discovering small classes and excludes the majority of the images in the final results, as can be seen in Fig. 4.5(d). $t_s$ is therefore set to $\sqrt{2}$ in our experiments to ensure a good capability of discovering small classes. If applicable, $\eta$ can be set according to the prior information, such as the average class size, of the dataset. Otherwise, it is advised to set it to a value that is large enough for constructing a reliable cluster centroid. We set $\eta = 5$ in our experiments.

### 4.4.3 Analyses

We conduct a series of experiments to investigate: 1) the superiority of the potential-based eviction over the random eviction, 2) the effectiveness of the adaptive threshold $\tau$, 3) the capability of conquering the $NC \gg SC$ problem, and 4) the comparison with other clustering algorithms on both small and large-scale datasets.

#### Superiority of the potential-based eviction

To demonstrate the advantages of the potential-based eviction (please refer to Equation (4.6)), we compared it with the random eviction, which is exactly the same as

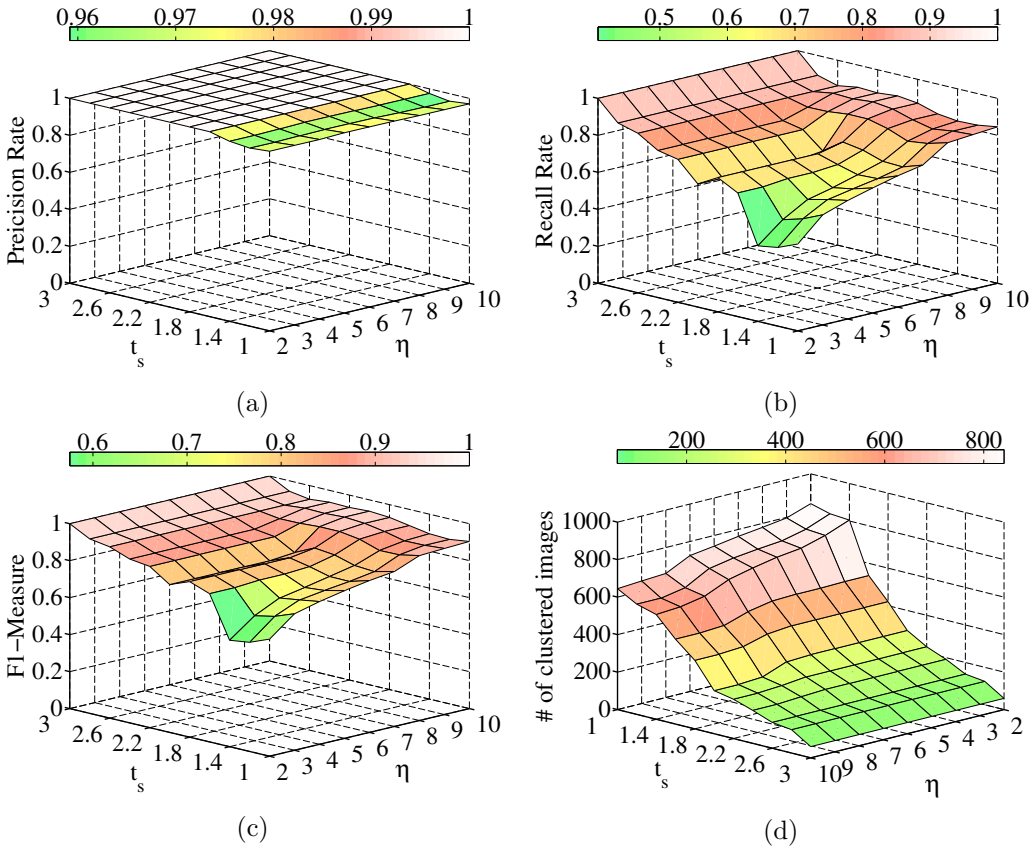



Figure 4.5: How the score threshold $t_s$ and the size $\eta$ of the minimal cluster affect the clustering results. (a) Precision rates. (b) Recall rates. (c) F1-Measures. (d) Number of clustered images.

the potential-based eviction except that all fingerprints in the previous batch, rather than those with lower potentials, will be evicted from RAM before loading the next batch. Suppose $n = 5,000$ fingerprints were equally divided into 10 batches (batch size $q = 500$), which will be sequentially loaded into RAM. Two synthetic experiments were conducted: in the first experiment, the class number $c$ was fixed to 20 to simulate the scenario where the $NC \gg SC$ problem is absent. $1,000$ class distributions were randomly generated under the constraint that the sum of the class sizes equals $5,000$. Based on the limited correlation calculations and I/O operations, a good eviction strategy can well explore the connectivity information of the graph consisting of fingerprints. Therefore, we used $R_d$, defined as the ratio of the number

of discovered edges to the total number of edges of the fingerprint graph, to measure the effectiveness of eviction strategies. Since the entropy is a good indicator of the symmetry of one distribution, we used the ratio of the entropy of the class distribution to that of the uniform distribution as the symmetry measurement $M_s$. Specifically,

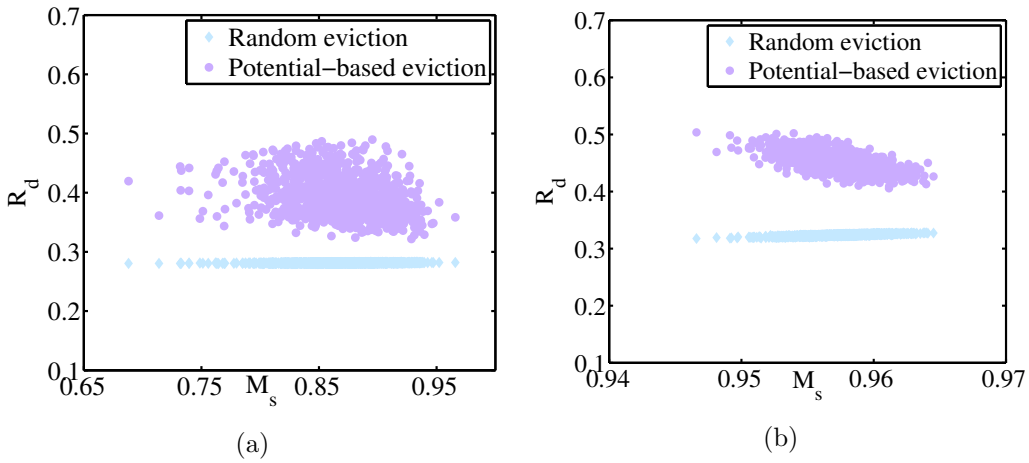
$$M_s = \frac{-1}{n \log_2 c} \sum_{i=1}^{c} n_i \log_2 \frac{n_i}{n}, \tag{4.19}$$

where $n_i$ is the number of fingerprints in the $i$th class, $n$ is the total number of fingerprints, and $c$ is the number of classes. The higher the $M_s$, the more symmetric the distribution, with 1 indicating the uniform distribution. In the second experiment, the same procedure was repeated with a different $c = 500$ to simulate the scenario where the number of classes is higher than the average class size (i.e., the $NC \gg SC$ problem is present).

Results of the two experiments are shown in Fig. 4.6(a) and 4.6(b), respectively. As can be seen, $R_d$ of the random eviction stays around the theoretical value $(3nq - 2q^2 - n)/(n^2 - n) \approx 0.28$, while $R_d$ of the potential-based eviction are consistently higher than those of the random eviction. As $M_s$ decreases, the class distribution becomes more unbalanced and the fingerprints from larger classes tend to have higher potentials. As a result, the edges connecting the fingerprints from larger classes are more likely to be discovered. Therefore, the advantage of the potential-based eviction grows as $M_s$ decreases, as more clearly shown in Fig. 4.6(b).

**Effectiveness of the adaptive threshold**

To investigate the effectiveness of the adaptive threshold $\tau$ in Equation (4.10), we randomly chose 900 images taken by three cameras (each responsible for 300 images), namely one Agfa DC-830i (referred as *Cam 1*) and two Kodak M1063s (referred as *Cam 2* and *Cam 3*), to simulate the inter-class and intra-class correlation distribu-




(a) (b)

Figure 4.6: Superiority of the potential-based eviction over the random eviction. (a) $c = 20$. (b) $c = 500$.

tions. Three sets of values of $n_x$ and $n_y$, i.e., $(n_x = 1, n_y = 10)$, $(n_x = 10, n_y = 30)$ and $(n_x = 30, n_y = 50)$, were tested. We first excluded the 300 images acquired by *Cam 3* to simulate the case that two cameras are of different models. The NCC $\rho$ between two centroids averaged over $n_x$ and $n_y$ camera fingerprints, which were randomly selected from the 300 images of *Cam 1* and the other 300 images of *Cam 2*, respectively, was calculated as one inter-class correlation. To prevent the same fingerprint from contributing to both centroids, the 300 images of *Cam 2* were equally divided into two groups. $n_x$ and $n_y$ images were randomly selected from the two groups and the correlation $\rho$ between the corresponding two centroids was calculated as one intra-class correlation. As indicated in Equation (4.11), $\sigma_x^2$ and $\sigma_y^2$ were estimated from the pairwise correlations of the camera fingerprints that were used to estimate the two centroids, respectively. We repeated the above procedure $2,000$ times to generate $2,000$ inter-class correlations and $2,000$ intra-class correlations, which were used to draw the inter-class distribution and intra-class distribution, respectively. To simulate the case where two cameras are of the same model, we replaced the images of *Cam 1* with those of *Cam 3* and repeated the above procedure again. Comparison with the other two adaptive thresholds pro-

posed in [114] and [115] were also performed. For the threshold proposed in [114], we set $n = \max(n_x, n_y)$ to make it applicable to more general cases.

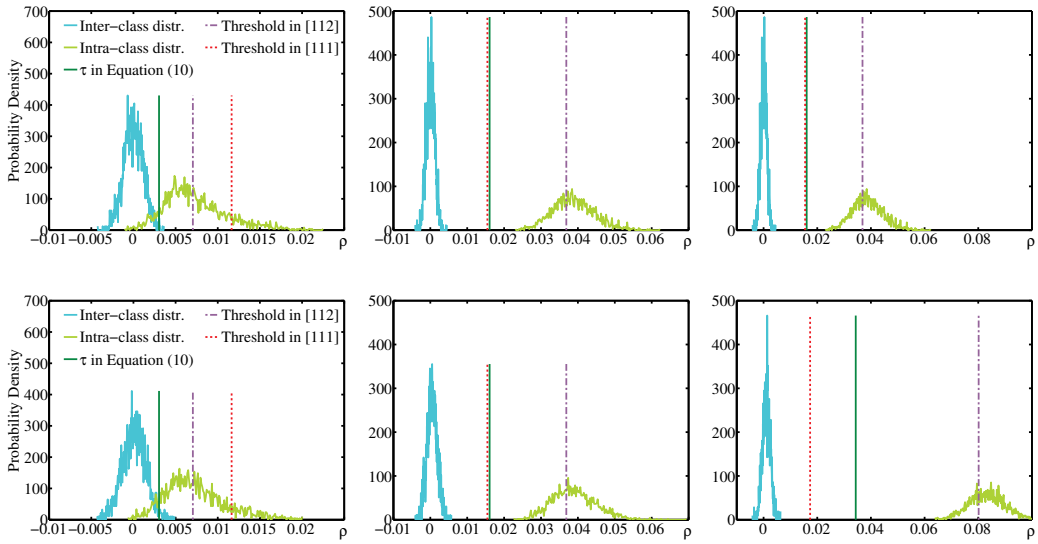



Figure 4.7: Comparison of the effectiveness of thresholds. The first and second rows show the results for two cases, i.e., cameras are of different models or the same model, respectively. From left to right, the columns show the results for four sets of values, i.e., $(n_x = 1, n_y = 10)$, $(n_x = 10, n_y = 30)$ and $(n_x = 30, n_y = 50)$, respectively.

As illustrated in the first row of Fig. 4.7, the inter-class distribution remains unchanged with regard to the increasing $n_x$ and $n_y$, while the intra-class distribution keeps shifting towards the right. The threshold proposed in [114] seems to be working well for large $n$, but closer inspection reveals that it almost stays the same with regard to the increasing $n$. As a consequence, it is either too conservative for large $n$ or overly aggressive for small $n$. Although the threshold proposed in [115] also shifts to the right, its shift is too aggressively, making some of the intra-class correlations misclassified as inter-class correlations. In fact, the unsatisfactory performance of the thresholds in [114] and [115] is to be expected because they only consider the number of fingerprints and ignore the fact that the correlation also heavily depends on the quality of fingerprints. That is the key reason why the threshold $\tau$ in Equation (4.10) is capable of adaptively finding a suitable breaking point between the inter-

class distribution and the intra-class distribution.

Similar results can be observed in the second row of Fig. 4.7, but the inter-class distribution moves to the right as $n_x$ and $n_y$ increase due to the non-unique artifacts shared by devices of the same model. What is worse, in practice it is possible that the fingerprints used to calculate the centroid are not solely from the same class due to the potential errors, which probably lengthens the right tail of the inter-class distribution and the left tail of the intra-class distribution. These two phenomenons make the two distributions more likely to overlap and therefore complicate the situation. A threshold indicated in Equation (4.10) approximately divides the margin evenly between the inter-class distribution and the intra-class distribution, so it can be expected to deliver a good performance in practice.

**Capability of conquering the $NC \gg SC$ problem**

To simulate the $NC \gg SC$ problem, we cropped 50 image blocks sized $1024 \times 1024$ at 50 different locations from each image in $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$ and $\mathcal{D}_4$. Since the SPN is location-based, such a simple and efficient way of image cropping simulates the process of generating images acquired by different sensors. For example, cropping 50 image blocks at different locations from each of the 40 images taken by one camera results in $2,000$ image blocks taken by 50 cameras (each accounting for 40 image blocks). Notice that the images blocks cropped from the same image undergo the same in-camera processing procedures, which possibly introduces non-unique artifacts and therefore makes the clustering task on the generated datasets even more challenging. The datasets generated from $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$ and $\mathcal{D}_4$ are referred to as $\mathcal{D}'_1$, $\mathcal{D}'_2$, $\mathcal{D}'_3$ and $\mathcal{D}'_4$, respectively, each of which consists of $50,000$ image blocks.

The clustering results are presented in Table 4.1, where $n_g$ is the ground-truth class number, $n_d$ is the number of the discovered *unique* classes with a size no smaller than $\eta$ in the final clustering results, and the remaining columns show the numbers of the discovered *unique* classes in each iteration. As can be seen, the clustering is

| Dataset | $n_g$ | $n_d$ | Iteration | | | | |
|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | 4 | 5 |
| $\mathcal{D}'_1$ | 1,250 | 1,145 | 586 | 445 | 253 | 58 | – |
| $\mathcal{D}'_2$ | 1,250 | 1,157 | 601 | 451 | 216 | 67 | – |
| $\mathcal{D}'_3$ | 2,500 | 1,982 | 637 | 532 | 606 | 304 | 108 |
| $\mathcal{D}'_4$ | 2,500 | 1,912 | 638 | 529 | 568 | 279 | 146 |

Table 4.1: Clustering results on the generated datasets

finished in a few iterations and the number of the discovered unique classes decreases significantly after the first few iterations. For the model-level datasets $\mathcal{D}'_1$ and $\mathcal{D}'_2$, about 92%∼93% of the classes can be discovered and the clustering can be finished in less iterations. But for the two more challenging datasets $\mathcal{D}'_3$ and $\mathcal{D}'_4$, only 76%∼79% of the classes are discovered mainly due to the small classes in them. Certainly, the number of the discovered unique classes can be increased by specifying a smaller $\eta$, but given the results in the first column of Fig. 4.7, if the cluster size is small, it is more likely that the two distributions are overlapped, which gives rise to false positives or false negatives even an appropriate threshold is chosen. Therefore, if the discovered clusters are about to be used for further merging or query (e.g., retrieving the similar clusters for the new incoming fingerprints), it is advisable to specify a reasonable high $\eta$ so as to improve the reliability of the system. So it is important to investigate the performance of the proposed framework when $\eta$ is set to a relatively high value.

Further investigations reveal that at least three reasons account for the difficulties of discovering small classes:

(a) Although the images are rotated to ensure the same horizontal orientation, they may still be incorrectly rotated (e.g., supposed to be rotated clockwise but rotated anti-clockwise). As a result, images from the same class are split into two classes, making small classes even more difficult to be identified.

(b) A threshold $t_b$ is specified to binarize the correlations. To achieve high pre-

cision rate of the clusters, $t_b$ should be large enough. But the side effect is that some fingerprints of the same class will be considered as unrelated due to the insufficiently large correlations, which has a more severe impact on small classes due to the limited number of intra-class correlations.

(c) Smaller classes are more sensitive to misclassification. Considering two classes, one consisting of five fingerprints and the other consisting of ten fingerprints, if four fingerprints in both the first class and the second class are wrongly classified, the first class will disappear from the final clustering results, while there is still a chance to form clusters for the second class.

It is the interplay of these reasons that makes discovering small classes in large-scale databases very challenging. Considering the fact that the size of classes in $\mathcal{D}'_3$ and $\mathcal{D}'_4$ can be as small as 10 or 20, and $\eta$ is set to a relatively high value 5, 76%∼79% is a very promising result.

**Comparison with other clustering algorithms**

In this section, we will compare the proposed clustering framework with other camera fingerprints clustering algorithms. Apart from $\mathcal{P}$, $\mathcal{R}$ and $\mathcal{F}$ as explained in Equations (4.17) and (4.18), we will also show the ratio of the number of discovered *unique* clusters to the ground-truth class number, i.e., $n_d/n_g$. It was observed in our experiments that the algorithm in [61] performs better and slightly faster than that in [60], so we will use [61] to represent the hierarchical clustering based algorithms. For the sake of convenience, we refer to Li's Markov random field based algorithm [57] as MRF, Liu's multi-class spectral clustering based algorithm [58] as SC, and Villalba's hierarchical clustering based algorithm [61] as HC. It is worth mentioning that 1/5 of the entire dataset is used as the membership committee in the first iteration of MRF to reduce the computational cost. For the proposed algorithm, the whole dataset is randomly partitioned into batches of equal size in the coarse

| Quality | Clustering algorithms | | | |
|---|---|---|---|---|
| | MRF [57] | SC [58] | HC [61] | Proposed |
| $\mathcal{P}$ | 0.9756 | 0.2367 | 0.5080 | **0.9980** |
| $\mathcal{R}$ | 0.8205 | **0.9833** | 0.8467 | 0.7320 |
| $\mathcal{F}$ | **0.8914** | 0.3816 | 0.6350 | 0.8442 |
| $n_d/n_g$ | **25**/25 | 6/25 | 13/25 | **25**/25 |

a

| Quality | Clustering algorithms | | | |
|---|---|---|---|---|
| | MRF [57] | SC [58] | HC [61] | Proposed |
| $\mathcal{P}$ | 0.9533 | 0.2605 | 0.5577 | **0.9879** |
| $\mathcal{R}$ | 0.8500 | **1.0000** | 0.8424 | 0.7382 |
| $\mathcal{F}$ | **0.8987** | 0.4134 | 0.6711 | 0.8446 |
| $n_d/n_g$ | **25**/25 | 5/25 | 15/25 | **25**/25 |

b

| Quality | Clustering algorithms | | | |
|---|---|---|---|---|
| | MRF [57] | SC [58] | HC [61] | Proposed |
| $\mathcal{P}$ | 0.8099 | 0.0600 | 0.8262 | **0.9940** |
| $\mathcal{R}$ | 0.7538 | **1.0000** | 0.6681 | 0.6023 |
| $\mathcal{F}$ | **0.7809** | 0.1132 | 0.7388 | 0.7499 |
| $n_d/n_g$ | 42/50 | 3/50 | **45**/50 | 43.3/50 |

c

| Quality | Clustering algorithms | | | |
|---|---|---|---|---|
| | MRF [57] | SC [58] | HC [61] | Proposed |
| $\mathcal{P}$ | 0.7027 | 0.0301 | 0.9478 | **0.9941** |
| $\mathcal{R}$ | 0.7942 | **1.0000** | 0.7236 | 0.6853 |
| $\mathcal{F}$ | 0.7456 | 0.0584 | **0.8207** | 0.8111 |
| $n_d/n_g$ | 31/50 | 1/50 | **46**/50 | 42.2/50 |

d

Table 4.2: Comparison of 4 different clustering algorithms on 4 different datasets: (a) $\mathcal{D}_1$, (b) $\mathcal{D}_2$, (c) $\mathcal{D}_3$, and (d) $\mathcal{D}_4$.

clustering stage, so as to give convincing results, we repeated the experiments 10 times and showed the average results for the proposed algorithm.

In the first experiment, four clustering algorithms were tested on the four

fixed-size small datasets, namely $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$ and $\mathcal{D}_4$. Results are shown in Table 4.2, where the highest value in each row is highlighted in bold. As can be seen, SC performs worst among the four algorithms in terms of the F1-measure. The underlying reasons is that SC terminates when the size of the smallest cluster equals 1. Due to the intrinsic characteristics of the fingerprint, if one camera fingerprint is severely contaminated, it can easily be regarded as unrelated to all other fingerprints and result in a singleton cluster. Such premature termination happens more often when the size of class is small. As a consequence, the number of unique clusters $n_d$ discovered by SC is significantly lower than that of the other three algorithms. MRF has the best performance on the model-level datasets, but its performance degrades on the device-level datasets, where the fingerprints of the cameras of the same model are more ambiguous and misleading for the clustering algorithm. Surprisingly, HC performs worse on the model-level datasets than on the device-level datasets. We looked into the clusters generated on the two model-level datasets and found that there are several large clusters containing the fingerprints from several cameras. So the rather contradictory results are caused by the incorrect agglomeration at an earlier stage, which will mislead and spoil the succeeding agglomerations. For the proposed clustering algorithm, the overall performance is balanced and stable across different datasets. The precision rate of the proposed algorithm keeps staying at 98%∼100% at the expense of a slightly lower recall rate, around 61%∼73%. This consequently retains the F1-measure of the proposed algorithm at a favorable level. But because of the reasons mentioned in Section 4.4.3, some classes in $\mathcal{D}_3$ and $\mathcal{D}_4$ are missing in the final results.

In the second experiment, we compare the time complexities and the clustering qualities of the four algorithms on various-size datasets. To generate datasets of various sizes, we incrementally added $1,000$ images captured by 10 cameras (100 images per camera) to an empty dataset until all the 74 cameras in the Dresden database had been covered. The four algorithms were run and evaluated on each of

these datasets. For the sake of completeness, we also evaluated the algorithms on the whole Dresden database, i.e., 15,840 images.

The running time (in seconds) is shown in Fig. 4.8(a). Since MRF, SC and HC require the calculation of pairwise correlations before clustering, the time used to load fingerprints from the disk and calculate the pairwise correlations is highlighted in green in the stacked bar. For the proposed clustering framework, the time used to load the projection matrix and calculate random projection is highlighted in pink in the stacked bar. What can be observed in Fig. 4.8(a) is that MRF is most time-consuming, followed by HC and SC. One interesting observation comes from MRF, for which the running time significantly increases by almost 100% when the image number slightly increases from 7,000 to 7,400. It was found that the required iteration number for the algorithm to converge becomes much higher for the slightly larger dataset, so the running time of MRF not only depends on the size but also the characteristics of the dataset, such as the quality of fingerprint and the class distribution. By applying the dimension reduction technique and the divide-and-conquer strategy, the running time of the proposed clustering framework is much lower than those of all the other three algorithms. For example, it only requires about 45 minutes to cluster the whole Dresden database on our ordinary desktop machine.

The precision rates, recall rates, F1-measures, and the ratios $n_d/n_g$ are illustrated in Fig. 4.8(b), 4.8(c), 4.8(d), and 4.8(e), respectively. Compared with the results given in Table 4.2, the performance of SC becomes better on the datasets containing large classes, because with more fingerprints in each class, it has a lower chance to trigger the premature termination. But compared with the other three algorithms, SC is still the worst algorithm and far from satisfactory. MRF is quite stable in terms of the recall rate, but the precision rate witnesses a remarkable and steady decrease as the image number increases, making it unsuitable for large-scale databases. For HC, both the number of discovered unique clusters and the precision

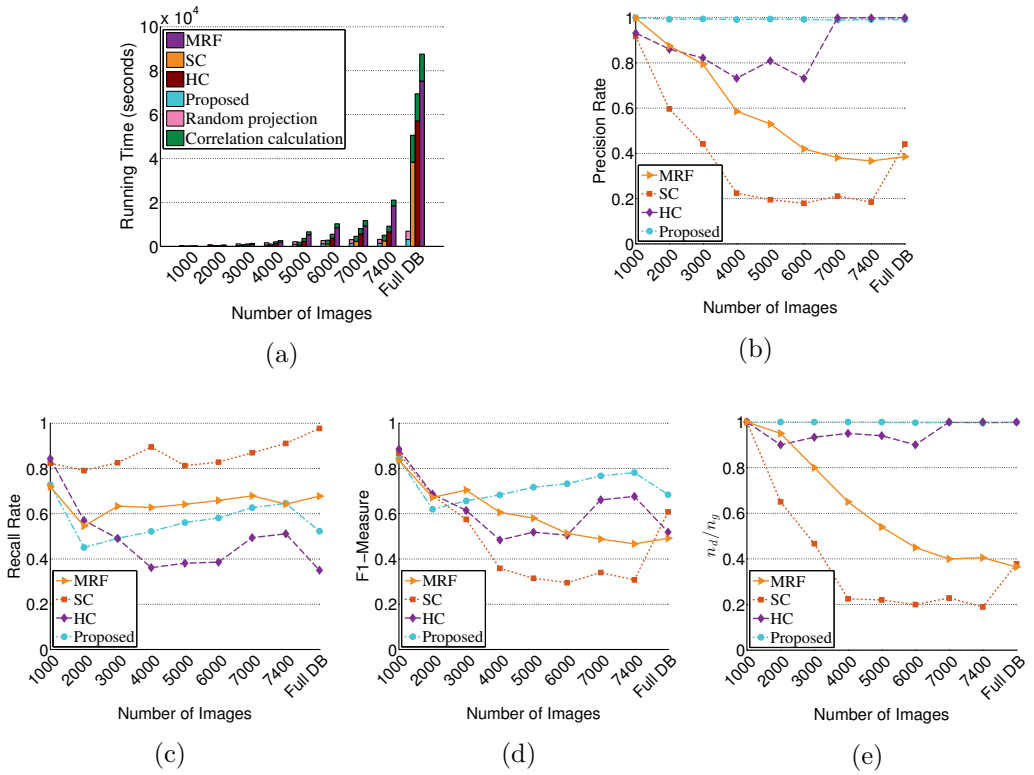



Figure 4.8: Running times and clustering qualities of different clustering algorithms on datasets with various sizes. (a) Running time (in seconds). (b) Precision rates. (c) Recall rates. (d) F1-Measures. (e) $n_d/n_g$.

rate stay at a high level, but its low recall rate indicates that it tends to produce small-size clusters. Moreover, the sudden increase of the precision rate of HC also indicates its instability across different datasets. While for the proposed algorithm, as the size of dataset increases, it not only achieves high and stable performance, but also discovers all the classes in the database. The high precision rate and the good capability of discovering classes makes the proposed algorithm attractive in many practical scenarios.

## 4.5 Conclusion

In this chapter, a novel clustering framework, without a training process, has been developed to address the $NC \gg SC$ problem commonly encountered when clus-

tering large-scale images based on camera fingerprint. By continually updating the qualities of clusters and applying the adaptive thresholding, the proposed framework works in a divide-and-conquer manner, and thus can be adopted to large-scale camera fingerprint databases. In comparison with the state-of-the-art camera fingerprints clustering algorithms, the proposed algorithm is much faster and delivers good clustering quality, especially for large databases. The high precision rate makes the proposed framework attractive in many practical applications such as information searching and retrieval. However, given the results in Table 4.1 and 4.2, some small classes cannot be discovered by the proposed algorithm due to the improper binarization and the potential misclassification errors. In fact, because of the nature of the noise-like camera fingerprints, it is hard to solve this issue especially in large-scale databases and further studies are needed to work around it. Another line of our future work will be the parallelization of the proposed framework. Different stages of the proposed framework, such as the calculation of the approximate correlation matrix, the fine clustering of the coarse clusters, and the centroid attraction can be processed in parallel. We believe that our research will serve as the effective tool for clustering large-scale camera fingerprints.

# CHAPTER 5

## Refining SPN-Based Image Forgery Detection

Sensor pattern noise (SPN) can be considered as a spread-spectrum watermark embedded in every image taken by the source imaging device. It has been effectively used for localizing forgeries in digital images. The noise residual extracted from the image in question is compared with the reference SPN in a block-wise manner. If their normalized cross correlation, which servers as a decision statistic, is below a pre-determined threshold (e.g., by the Neyman-Pearson criterion), the center pixel of the window is declared as forged. However, as mentioned in Chapter 2, the decision statistic is calculated over the forged and the non-forged regions when the detection block falls near the boundary of the two different regions. As a result, the pixels of the forged region near the boundary area are likely to be wrongly identified as genuine ones. To alleviate this problem, in this chapter, we propose an algorithm to refine the initial detection result along the boundary. We analyze the correlation distribution in the problematic region and refine the detection by weighting the decision threshold based on the altered correlation distribution. The effectiveness of the proposed refining algorithm is validated through the detection results on simulated image forgeries and three different kinds of realistic image forgeries.

The rest of this chapter is organized as follows. In Section 5.1, we will revisit the constant false acceptance rate (CFAR) method. In Section 5.2, we will introduce the missing detection problem arising from the CFAR method. Section 5.3 presents the proposed algorithm and Section 5.4 validates the proposed algorithm by detecting both simulated and realistic image forgeries. Finally, Section 5.5 concludes this chapter.

## 5.1 Background

Detecting image forgeries is an interesting while very challenging task due to the variety of image manipulations a user can perform with increasingly powerful image editing softwares. As mentioned in Chapter 1, active techniques, such as digital watermarking, are effective in verifying the authenticity of an image, but the requirement of originally embedding into the protected image limits their widespread use in practice. Therefore, there has been growing interest in passive techniques. As one of the most promising passive techniques, SPN has been proven to be a powerful and robust tool for exposing image forgery [6, 8, 9, 72, 75, 117]. Its capability of detecting image forgeries irrespective of the specific type of forgery arouses wide attention of the researchers in the field of digital image forensics. In [9], an image forgery detection technique based on SPN was proposed. In what follows, we will revisit this algorithm in detail and explain how the missing problem arises from the constant false acceptance rate (CFAR) criterion it applies.

Like in [9], we formulate the problem of detecting SPN signal in noise residual $w$ as a binary hypothesis test

$$\begin{cases} h_0 : w = v \\ h_1 : w = r + v, \end{cases} \tag{5.1}$$

where $r$ is the signal of interest (i.e., the reference SPN) and $v$ is the SPN-irrelevant noise. If an image region has been tampered with, the SPN signal in noise residual $w$ of that region is lost. Therefore, image forgeries can be exposed by identifying the image regions where the SPN signal is absent. Since SPN, by its very nature, is a very weak noise-like signal, its reliable detection requires jointly processing a large number of pixel samples, typically in a block-wise manner. For a target pixel $q_i$, a decision statistic $\rho_i$ is calculated based on the normalized cross correlation (NCC)

between $w_{N_i}$ and $r_{N_i}$:

$$\rho_i = \frac{\sum_{j \in N_i} (w_j - \bar{w})(r_j - \bar{r})}{\sqrt{\sum_{j \in N_i} (w_j - \bar{w})^2} \sqrt{\sum_{j \in N_i} (r_j - \bar{r})^2}}, \tag{5.2}$$

where $N_i$ is the pixel indices within a $n \times n$ sliding detection block centered at $q_i$. To reveal the forgery, $\rho_i$ is then compared with a threshold $\gamma$:

$$\hat{u}_i = \begin{cases} 1, \rho_i < \gamma \\ 0, \text{otherwise} \end{cases} \tag{5.3}$$

where $\hat{u}_i \in \{0, 1\}$ is a binary value indicating the forgery (1 for forgery and 0 for genuine pixel). $\gamma$ is usually selected according to the Neyman-Person criterion to ensure a small false acceptance rate (FAR), i.e., $Pr(\hat{u}_i = 0 | u_i = 1)$, with $u_i \in \{0, 1\}$ the ground truth. However, even for the non-forged pixels, the NCC coefficients might happen to be very low in the image areas of dark, saturated or highly textured. Based on the correlation predictor proposed in [9] (please refer to Section 2.3.2 for more details), this problem is addressed by predicting the correlation distribution $p(x|h_1)$ and correcting the tampered pixels for which the false rejection

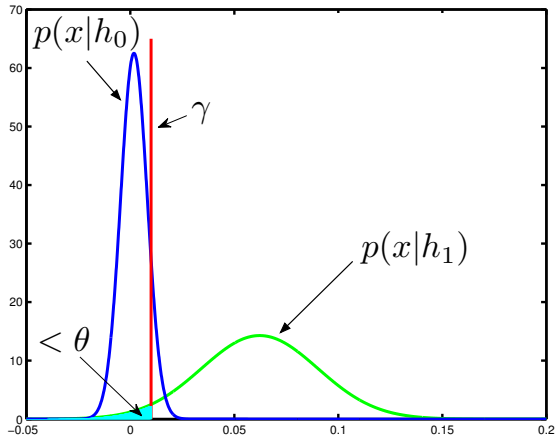



Figure 5.1: How to determine the thresholds for given inter-camera and intra-camera correlation distributions.

rate (misidentifying non-tampered as tampered) higher than a threshold $\theta$, i.e.,

$$\int_{-\infty}^{\gamma} p(x|h_1)dx > \theta, \tag{5.4}$$

to non-tampered. Fig. 5.1 shows how to determine the thresholds $\gamma$ and $\theta$ when $p(x|h_0)$ and $p(x|h_1)$ are given.

We would like to spend more words on the estimation of the correlation distribution under hypothesis $h_0$ and $h_1$ (i.e., $p(x|h_0)$ and $p(x|h_1)$). It was observed in our experiments that if the reference SPN $r$, is preprocessed by the Wiener Filtering in DFT domain (WF) [9] or our recently proposed spectrum equalization algorithm (SEA) [118], $p(x|h_0)$ fits quite well with the Gaussian distribution $\mathcal{N}(0, 1/d)$, where $d = n \times n$ is the number of pixels within the square detection block. One example for $d = 128\times128$ is shown in Fig. 5.2, where $p(x|h_0)$ is estimated using 19200 image blocks cropped from images taken by other cameras, and $p(x|h_1)$ is estimated using 15360 image blocks cropped from images taken by the same camera. For the estimation of $p(x|h_1)$, we use a Gaussian model rather than the generalized Gaussian model in [9] due to its simplicity. With the help of the correlation predictor

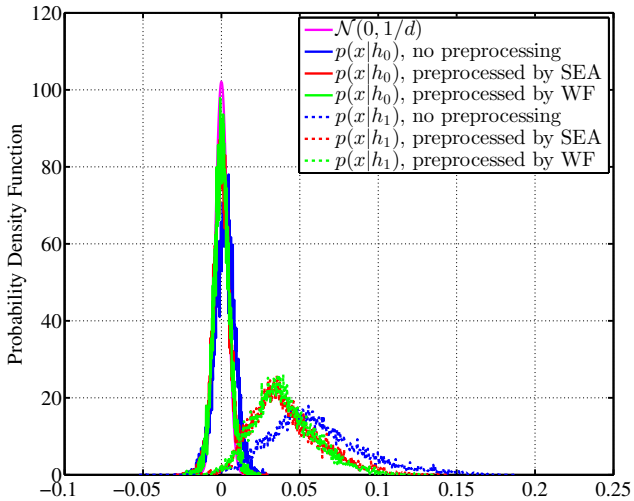



Figure 5.2: How preprocessing affects the correlation distribution $p(x|h_0)$ and $p(x|h_1)$.

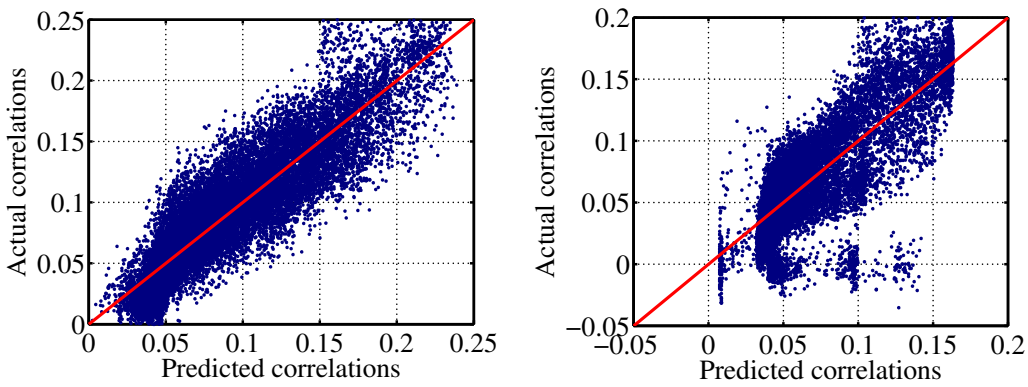



Figure 5.3: Predicted correlations obtained with the correlation predictor proposed in [9] using 20480 image blocks of size $d = 128 \times 128$ pixels for (a) a Canon IXY500 and (b) a Canon IXUS 850IS.

described in Section 2.3.2, the mean and the variance of the Gaussian distribution are the predicted correlation and the variance of prediction errors, respectively. To see how well the correlation predictor performs, we showed the predicted and actual correlations for two cameras, a Canon IXY500 and a Canon IXUS 850IS, in Fig. 5.3(a) and 5.3(b), respectively. As can be seen, most of the points scatter along the line $y = x$, which means the predicted correlations agree well with the actual correlations. However, we have two remarks to make for the correlation predictor:

1. As illustrated in Fig. 5.3(a), the variance at the low end tends to be smaller than that at the high end. The lower variance is not due to the insufficient image blocks with correlations at the low end, because the correlations lower than 0.05 takes up more than 1/3 of the entire population. The result shown in Fig. 5.3(a) indicates that the prediction error given by the correlation predictor does depend on the predicted correlation, which is opposite to the conclusion in [9].

2. In Fig. 5.3(b), there are a small number of outliers lying along the line corresponding to the actual correlation $\rho = 0$. Further investigation reveals that the outliers result from the image blocks with low intensities or saturation. These

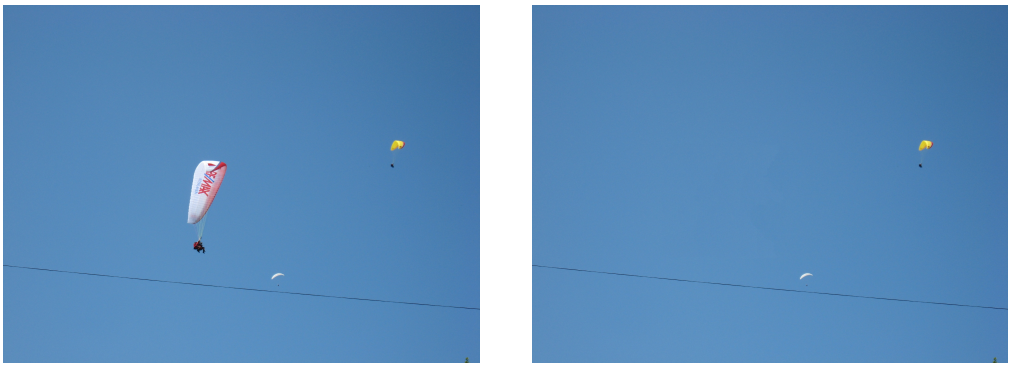
outliers are also responsible for the inaccurate variance estimation for $p(x|h_1)$, because the variance is calculated over the entire population of training data.

## 5.2 Missing Detection Problem

The algorithm proposed in [9] applies the constant false acceptance rate (CFAR) (i.e., misidentifying a tampered block as non-tampered) method to determine a fixed threshold. However, as pointed out in [75, 117], when the detection block falls near the boundary of the tampered and the non-tampered regions, the decision statistic becomes a weighted average of two different contributions and may lead to a high false acceptance rate (FAR), as illustrated in Fig. 5.5. This problem can be alleviated by means of hard [117] or soft [75] image segmentation to obtain the boundary information before detection. These two algorithms share the same essence of making use of the extra structure information of the image content, but their drawbacks are twofold: Firstly, the detection result heavily depends on the quality of image segmentation or the pilot image [76], but an accurate image segmentation or high-quality pilot image is not easy to obtain. The second and most critical drawback is that they are incapable of detecting the *occlusive* forgery, where objects in the original scene are hidden by placing a homogeneous background on them, or the image forgeries based on image inpainting [119, 120] or texture synthesis [121–123]. One example of image inpainting is shown in Fig. 5.4(b), where there is no structure information in the forged area that can be used for image segmentation.

## 5.3 Proposed Method

The proposed algorithm aims at improving the resolution of SPN-based image forgery detection. But unlike the algorithms based on image segmentation, it approaches the missing detection problem from a different perspective. Starting from an initial detection, we model how the decision statistic changes as the detection



(a) (b)

Figure 5.4: An example of image inpainting. (a) Original image (b) Forged image.

block moves across the boundary of two different regions (i.e., tampered and non-tampered) and adjust the decision threshold accordingly to achieve a more satisfactory detection. We assume that the $d$-dimensional signal within the detection block, either for the estimated $r_{N_i}$ or $w_{N_i}$, is standardized to have zero mean and unit variance, which means each element, $r_j$ or $w_j$ ($j \in N_i$), is independently drawn from the identical normal distribution $\mathcal{N}(0,1)$. Presumably, each element in the standardized signal can be modeled as the sum of the true SPN signal and other irrelevant interferences:

$$\begin{cases} w_j = x_j + \alpha_j \\ r_j = y_j + \beta_j, \end{cases} \tag{5.5}$$

where $x_j$ follows a Gaussian distribution $\mathcal{N}(0,\sigma^2)$ and $\alpha_j$ conforms to $\mathcal{N}(0,1-\sigma^2)$. Likewise, $y_j \sim \mathcal{N}(0,\lambda\sigma^2)$ and $\beta_j \sim \mathcal{N}(0,1-\lambda\sigma^2)$. Here, $\sigma^2$ and $\lambda\sigma^2$ can be viewed as the quality of the true SPN signal in $w_{N_i}$ and $r_{N_i}$, respectively. Note that $\lambda$ accounts for the different qualities of the SPN signal in $w_{N_i}$ and $r_{N_i}$. With the standardized signal, the decision statistic $\rho_i$ in Equation (5.2) is simplified as

$$\rho_i = \frac{1}{d}\sum_{j\in N_i}(x_j y_j + \alpha_j y_j + \beta_j x_j + \alpha_j \beta_j). \tag{5.6}$$

If $x$ and $y$ are from two different cameras (i.e., under hypothesis $h_0$), using the Central Limit Theorem (CLT), $\rho_i$ follows a Gaussian distribution $\mathcal{N}(\mu_{h_0}, \sigma^2_{h_0})$, where $\mu_{h_0} = 0$ and $\sigma^2_{h_0} = 1/d$. While under hypothesis $h_1$, we have $y_i = \sqrt{\lambda} x_i$. Therefore, Equation (5.6) can be rewritten as

$$\rho_i = \frac{1}{d} \sum_{j \in N_i} (\sqrt{\lambda} x_j^2 + \sqrt{\lambda} \alpha_j x_j + \beta_j x_j + \alpha_j \beta_j). \tag{5.7}$$

It is known that $x_j^2/\sigma^2$ follows the Chi-square distribution with 1 degree of freedom, $\chi^2(1)$. So based on the assumption that $x_j$, $\alpha_j$ and $\beta_j$ are mutually independent, we can easily arrive at

$$\rho_i \sim \mathcal{N}(\mu_{h_1}, \sigma^2_{h_1}), \tag{5.8}$$

where

$$\begin{cases} \mu_{h_1} = \sqrt{\lambda}\sigma^2 \\ \sigma^2_{h_1} = (1 + \lambda\sigma^4)/d. \end{cases} \tag{5.9}$$

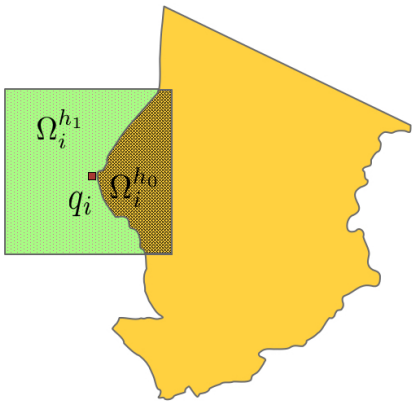



Figure 5.5: The square detection block across the non-tampered region $\Omega_i^{h_1}$ and the tampered region $\Omega_i^{h_0}$.

Equation (5.8) is the decision statistic distribution if the detection block falls completely on the non-tampered region. To see how the problematic region is incompatible with the distribution, let us look at Fig. 5.5, which shows a detection block sliding across the non-tampered region the tampered region $\Omega_i^{h_0}$ (in

yellow background) and $\Omega_i^{h_1}$ (in green background). The decision statistic therefore becomes a weighted average of two different contributions:

$$\begin{aligned}\rho_i &= \frac{1}{n_i^{h_0}} \sum_{j \in \Omega_i^{h_0}} (x_j y_j + \alpha_j y_j + \beta_j x_j + \alpha_j \beta_j) \\ &+ \frac{1}{n_i^{h_1}} \sum_{j \in \Omega_i^{h_1}} (\sqrt{\lambda} x_j^2 + \sqrt{\lambda} \alpha_j x_j + \beta_j x_j + \alpha_j \beta_j),\end{aligned} \tag{5.10}$$

where $n_i^{h_0}$ and $n_i^{h_1}$ are the number of pixels in $\Omega_i^{h_0}$ and $\Omega_i^{h_1}$ that surrounds pixel $q_i$, respectively. $x_j$ and $y_j$ are independent in the second summation term for region $\Omega_i^{h_0}$. Therefore,

$$\rho_i \sim \mathcal{N}(\mu, \Sigma) \tag{5.11}$$

where

$$\begin{cases} \mu_i = \sqrt{\lambda} \sigma^2 n_i^{h_1} / d \\ \sigma_i^2 = 1/d + \lambda n_i^{h_1} \sigma^4 / d^2, \end{cases} \tag{5.12}$$

To drop the unknown $\lambda$ and $\sigma^2$, we finally rewrite Equation (5.12) as the form of weighting $\mu_{h_1}$ and $\sigma_{h_1}^2$

$$\begin{cases} \mu_i = n_i^{h_1} \mu_{h_1} / d \\ \sigma_i^2 = (n_i^{h_1} d \Sigma_1 + d - n_i^{h_1}) / d^2, \end{cases} \tag{5.13}$$

where $d = n_i^{h_0} + n_i^{h_1}$. This is the final expression of the distribution of the decision statistic in the boundary region. As shown in Fig. 5.6, if $n_i^{h_1} = 0$, which means the detection block entirely falls in the tampered region, i.e., $n_i^{h_1} = d$, the decision statistic $\rho_i$ conforms to $\mathcal{N}(\mu_{h_0}, \sigma_{h_0}^2)$. As the block moves away and completely falls in the non-tampered region, $\rho_i$ follows the distribution $\mathcal{N}(\mu_{h_1}, \sigma_{h_1}^2)$. Note that in Equation 5.13, we only concern about the number of non-tampered pixels, so we will drop the superscript "$h_1$" for "$n_i^{h_1}$" afterwards. After analyzing the correlation distribution in the problematic area near the boundary of the tampered and the

non-tampered regions, we therefore propose an algorithm to alleviate the missing detection problem as follows:

1. Calculate the correlation $\rho_i$ between the noise residual $w_{N_i}$ and the estimated reference SPN signal $r_{N_i}$ within the detection block $N_i$ centered at pixel $q_i$.

2. Estimate the expected correlation $\bar{\rho}_i$ and the variance $\sigma^2_{h_1}$ of the NCC coefficients under hypothesis $h_1$ using the correlation predictor proposed in [9];

3. Select two thresholds $\gamma$ and $\theta$ to obtain an initial detection result $\hat{u}_i$ using Equation (5.3) and (5.4).

4. Obtain the number $n_i$ of pixels in the detection block centered at pixel $q_i$ that belong to the non-tampered region by convoluting a $n \times n$ matrix of ones with the initial detection result.

5. Calculate a new threshold $\gamma_i$ for each pixel by solving the following equation

$$\frac{1}{\sigma_i}\int_{-\infty}^{\gamma_i} e^{-\frac{(t-\mu_i)^2}{2\sigma_i^2}}\,dt = \int_{-\infty}^{\gamma} e^{-\frac{t^2}{2}}\,dt, \tag{5.14}$$

   where

$$\begin{cases} \mu_i = \bar{\rho}_i n_i / d \\ \sigma_i^2 = (\sigma^2_{h_1} d n_i + d - n_i)/d^2. \end{cases} \tag{5.15}$$

   The purpose of Equation (5.14) is to guarantee the same desired FAR along the boundary as in other regions according to the altered distribution as formulated in Equation (5.13).

6. Label a pixel $q_i$ as tampered ($\hat{u}_i = 1$) if $\rho_i < \gamma_i$.

7. Label a tampered pixel ($\hat{u}_i = 1$) as non-tampered if

$$\frac{1}{\sqrt{2\pi}\sigma_{h_1}}\int_{-\infty}^{\gamma} e^{-\frac{(t-\bar{\rho}_i)^2}{2\sigma^2_{h_1}}}\,dt > \theta. \tag{5.16}$$

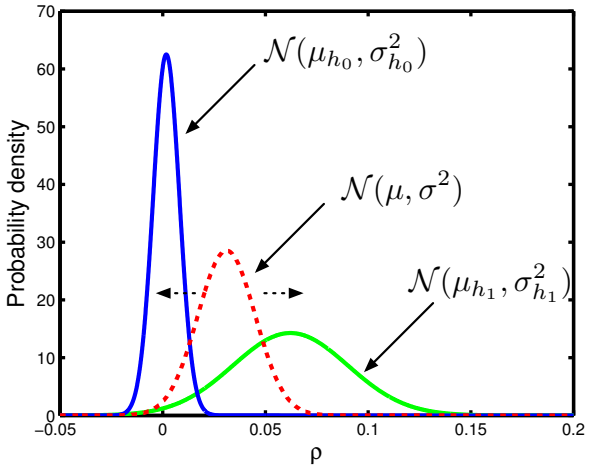



Figure 5.6: How the correlation distribution changes as the detection block moves across the boundary region.

## 5.4 Experiments

### 5.4.1 Experimental Setup

In this section, we will report some preliminary experiments meant to support the idea of weighting the threshold accordingly along the boundary to improve the detection resolution. Our experiments were carried out on images taken by six cameras, a Canon IXUS 850IS, a Canon PowerShot A400, a Canon IXY500, a FujiFilm FinePix S602, a FujiFilm FinePix A920, and a Olympus C730UZ. The images contain a wide variety of natural indoor and outdoor scenes taken during holidays, around campus and cities, in offices and laboratories, etc. For each camera, 80 randomly selected natural images were used for training the correlation predictor, and another 50 blue sky images were used for constructing the reference SPN, which is further preprocessed by the algorithm proposed in [118] to make the correlation distribution under hypothesis $h_0$ fit better to the theoretical distribution $\mathcal{N}(0, 1/d)$. Another 100 testing images from each camera (i.e., 600 images in total) will be used for evaluating the performance of the proposed algorithm. All images have the same

size of $1024 \times 1024$ pixels, and were cropped from the central region of the original images of size $1536 \times 2048$ pixels. To improve the quality of the noise residual, the nonlocal denoising filter BM3D [5] was used as the SPN extractor. Note that, only the green channel of images was considered in our experiments, but better performance can be expected by combining the results on other two color channels.

### 5.4.2 Detecting Simulated Forgeries

In the first experiment, we will evaluate the proposed algorithm by detecting simulated image forgeries. To generate the forged images, we randomly and equally partitioned the 100 testing images of each camera into 5 groups. Denoted by $G_i^j[k], i = 1, ..., 6, j = 1, ..., 5, k = 1, ..., 20$ is the $k$th image in the $j$th group of camera $i$. Each of the 5 groups of one camera is paired with one group from other 5 cameras. In this way, we can obtain 15 paired groups, i.e., $\{G_1^1, G_2^1\}, \{G_1^2, G_3^1\}, ..., \{G_5^5, G_6^5\}$, as shown in Fig. 5.7. In the paired groups, there is a one-to-one correspondence between the 20 images in one group and the 20 images in the other group. An image, say $G_1^1[k]$, is forged by swapping the central image block with its corresponding image, i.e., $G_2^1[k]$. By doing so, the images of one camera are forged with the images from all the other 5 cameras. To study how performance depends on forgery size, we used forgeries of three sizes (i.e., the size of the swapped central image block), $384 \times 384$, $256 \times 256$, and $128 \times 128$ pixels, creating thus three test subsets of 100 images each. We also used detection blocks of three sizes, $256 \times 256$, $128 \times 128$, and $64 \times 64$ pixels. The performance is characterized by the ROC (receiver operating characteristic) curve and the area under ROC curve (AUC). Like in [72], each ROC is computed by varying $\gamma$ from $-1$ to 1 and $\theta$ from 0 to 1, and then taking the upper envelope of the resulting $(\mathrm{FPR}, \mathrm{TPR})$ points.

ROC curves obtained with different detection block sizes for different forgery sizes, as well as the comparisons with the CFAR method [9], are shown in Fig. 5.8. The AUCs for forgeries of size $384 \times 384$, $256 \times 256$, and $128 \times 128$ pixels

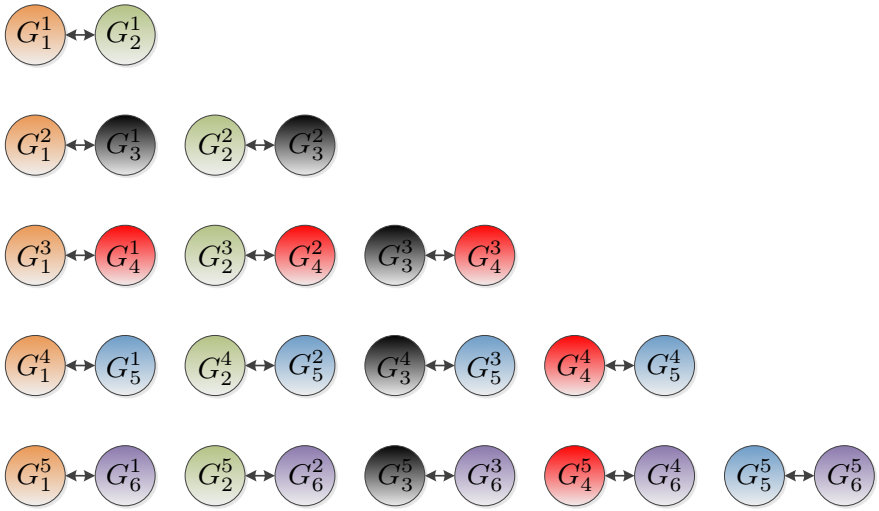



Figure 5.7: 15 paired groups for generating simulated image forgeries.

are presented in Table 5.1, 5.2, and 5.3, respectively. By refining the detection results along the boundary, the proposed algorithm performs uniformly better than the CFAR method. But the improvement varies from different forgery sizes and detection block sizes. Generally speaking, the larger the forgery is, the easier it can be detected. So it is not surprising to see that both the most significant AUC improvement 6.99% and the best AUC performance 94.38% are achieved on images with a forgery size $384 \times 384$ pixels. What may be rather contradictory to our expectation is that, a larger detection block does not necessarily result in a better performance. For example, if a $256 \times 256$ detection block is used to detect the forgeries of size $128 \times 128$ pixels, the AUC can be as low as around 60%, but a much higher AUC 88.5% can be obtained with a $64 \times 64$ detection block. Actually, when a large detection block is used to detect much smaller forgeries, the detectability of the forgery will be dramatically reduced, because the forged region only accounts for a small part of the area within the detection block, and the calculated statistics would become irrelevant. For a reasonably large forgery, say larger than $128 \times 128$ pixels, we found that a detection block of size $128 \times 128$ pixels delivers a satisfactory performance.

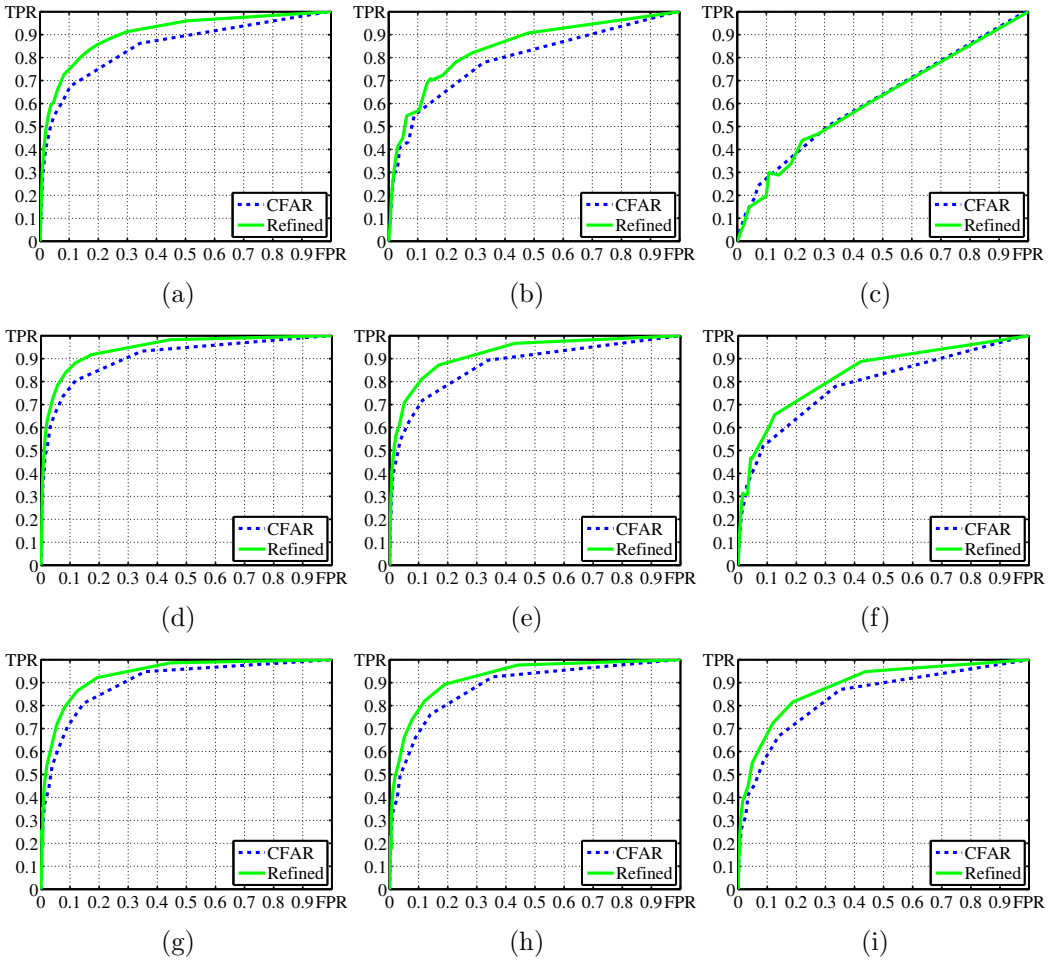



Figure 5.8: ROC curves obtained with different detection block sizes for different forgery sizes. (a) Forgery size: 384×384, $d$=256×256. (b) Forgery size: 256×256, $d$=256×256. (c) Forgery size: 128×128, $d$=256×256. (d) Forgery size: 384×384, $d$=128×128. (e) Forgery size: 256×256, $d$=128×128. (f) Forgery size: 128×128, $d$=128×128. (g) Forgery size: 384×384, $d$=64×64. (h) Forgery size: 256×256, $d$=64×64. (i) Forgery size: 128×128, $d$=64×64.

| AUCs | Detection block size $d$ | | |
|---|---|---|---|
| | 256×256 | 128×128 | 64×64 |
| CFAR | 0.8393 | 0.8933 | 0.9006 |
| Refined | **0.9052** | **0.9438** | **0.9376** |

Table 5.1: AUCs for forgeries of size 384 × 384 pixels.

| AUCs | Detection block size $d$ | | |
|---|---|---|---|
| | 256×256 | 128×128 | 64×64 |
| CFAR | 0.7803 | 0.8709 | 0.8678 |
| Refined | **0.8425** | **0.9219** | **0.9235** |

Table 5.2: AUCs for forgeries of size 256 × 256 pixels.

| AUCs | Detection block size $d$ | | |
|---|---|---|---|
| | 256×256 | 128×128 | 64×64 |
| CFAR | 0.6078 | 0.7727 | 0.8358 |
| Refined | **0.6101** | **0.8321** | **0.8850** |

Table 5.3: AUCs for forgeries of size 128 × 128 pixels.

### 5.4.3 Detecting Realistic Forgeries

Apart from detecting the simulated image forgeries, it would be interesting to see the performance of the proposed algorithm on detecting realistic image forgeries. To this end, we generated three forged images using different types of forgery techniques. As shown in Fig. 5.9-5.11, three different types of forgeries were involved:

- Scaling forgery (image 1): A direction board in an image taken by FujiFilm FinePix A920 is enlarged.
- Cut-and-paste forgery (image 2): A car in an image is cut and pasted onto another image. The two images are both taken by Olympus C730UZ.
- Copy-and-move forgery (image 3): A computer in an image taken by Canon IXY500 is copied and moved to a new location in the same image.

All the three forged images have the same size of 1536 × 2048 pixels.

We attempt to compare the proposed algorithm, which weights the threshold controlling the FAR in boundary region, with the method based on the constant false acceptance rate (CFAR) decision rule in [9]. The detection results shown in Fig. 5.9, 5.10 and 5.11, were obtained with detection blocks of sizes 256 × 256, 128 × 128

and $64 \times 64$ pixels. $\gamma$ and $\theta$ were set to 0.01 and 0.05, respectively. The forged area is highlighted in white, the correctly detected area is highlighted in green, while the area falsely labeled as tampered is highlighted in red. The AUCs for image 1, image 2, and image 3 are shown in Table 5.4, 5.5, and 5.6, respectively.

The first image is a simple case, where an enlarged direction board is placed on a smooth and bright wall. It is not surprising to see both the proposed algorithm and CFAR can accurately detect the forged area even using a detection block as small as $64 \times 64$ pixels. But the proposed refining algorithm clearly does a better job in the upper and the bottom boundary of the forged direction board. Furthermore, most of the "holes" in the middle of the forged area are filled up by the proposed refining algorithm. The second columns of Fig 5.9-5.11 show the detection results of the cut-and-paste forgery. As can be seen, the false positives are hard to avoid due to the dark area of the traffic lights and the complex background, e.g., trees and grass. In spite of the slightly more false positives, most of the forged car is reliably detected. Similar result can be observed in detecting smaller tampered area, as shown in the third columns of Fig. 5.9-5.11. The refined detection result reveals part of the forgery in the stand of the monitor, which is almost ignored by CFAR.

Given the detection results in Fig. 5.9, we would like to point out the fundamental differences between our proposed refining algorithm and the image dilation operation used as a post-processing step in [9]. If we compare the detection results of image 2 given by the CFAR method and the proposed refining algorithm in Fig. 5.9, we will find that the refining algorithm extends the detected areas in the bottom and the right part of the forged car, but it does not further extend the false positive area on the upper-left corner of the forged car. In other words, the proposed algorithm refines the boundary area by considering the neighboring information. It is fundamentally different from the isotropic image dilation operation, which indiscriminately enlarges the boundaries of forged regions and heavily depends on the shape and size of the applied structuring kernel.

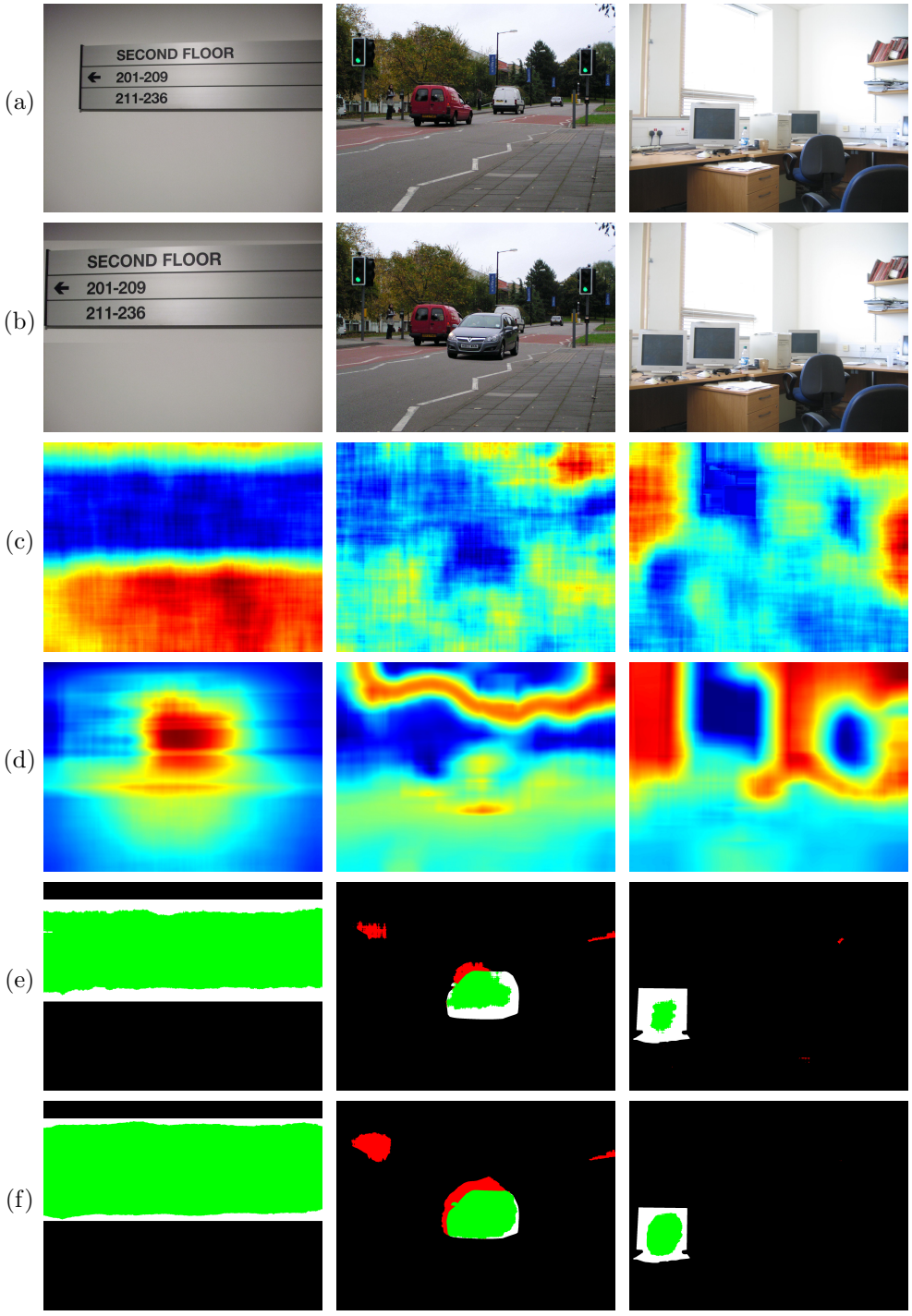



Figure 5.9: Forgery detection results for scaling (the first column), copy-and-move (the second column) and cut-and-paste (the third row) forgery using a detection block of 256 × 256 pixels. (a) Original image. (b) Forged image. (c) Predicted correlation field. (d) Actual correlation field. (e) Detection result by CFAR. (f) Refined detection result by our proposed algorithm.

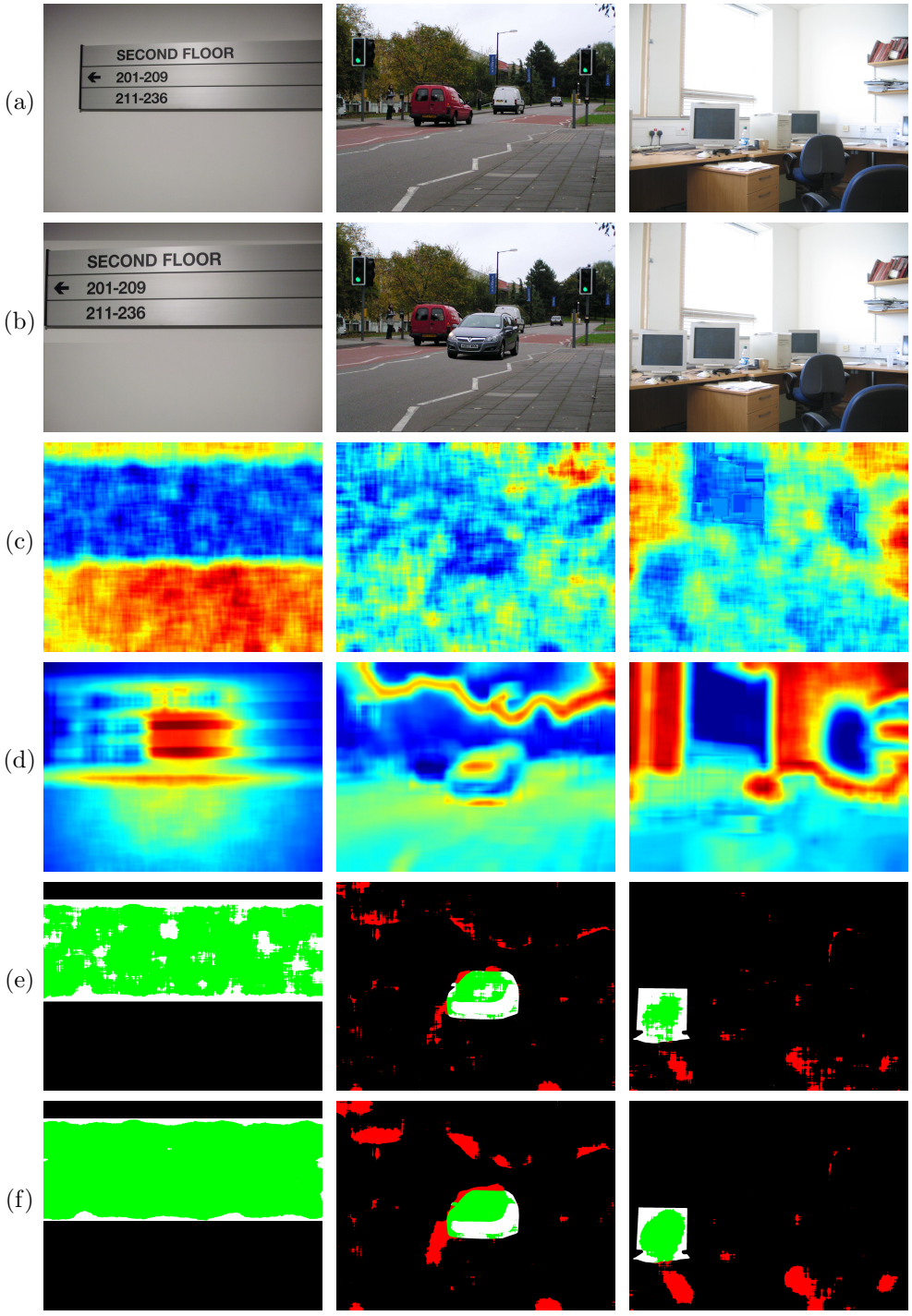



Figure 5.10: Forgery detection results for scaling (the first column), copy-and-move (the second column) and cut-and-paste (the third row) forgery using a detection block of $128 \times 128$ pixels. (a) Original image. (b) Forged image. (c) Predicted correlation field. (d) Actual correlation field. (e) Detection result by CFAR. (f) Refined detection result by our proposed algorithm.

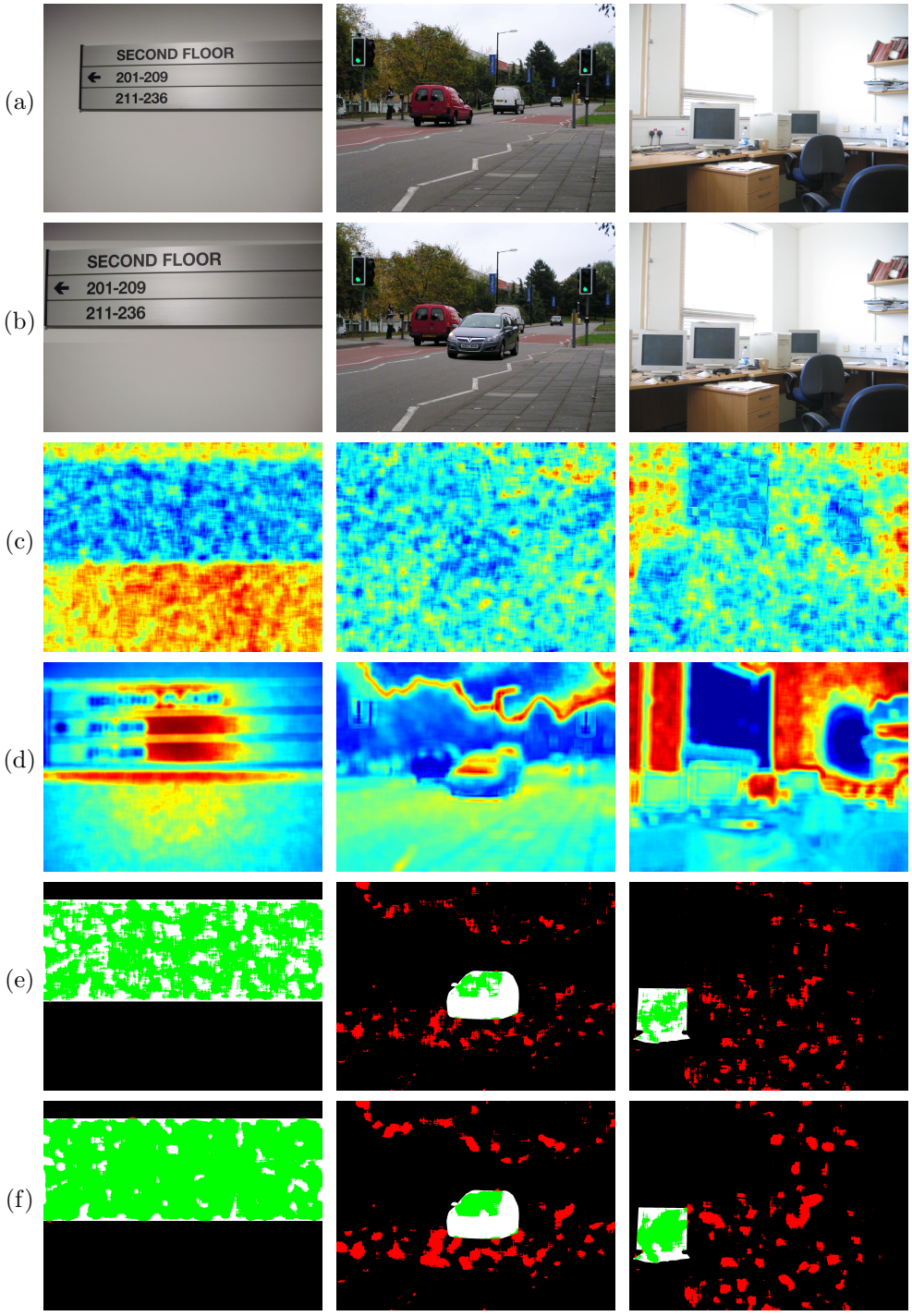



Figure 5.11: Forgery detection results for scaling (the first column), copy-and-move (the second column) and cut-and-paste (the third row) forgery using a detection block of $64 \times 64$ pixels. (a) Original image. (b) Forged image. (c) Predicted correlation field. (d) Actual correlation field. (e) Detection result by CFAR. (f) Refined detection result by our proposed algorithm.

We also show the ROC curves obtained with detection blocks of different sizes in Fig. 5.12 and report the AUCs in Table 5.4-5.6. By revealing more possible forgeries along the boundary, the refined detection result apparently fits more closely to the actual shape of the tampered area, which can potentially provide the forensic investigator with more detailed information. What can be seen in Fig. 5.12 is that the superiority of the refining algorithm seems more evident in the more challenging detection tasks. The AUC improvement on image 1 is very limited, but on more challenging image 2 and 3, the AUC improvement can be as high as 4.4%. It is worth mentioning that when a small detection block of size $64 \times 64$ pixels is used, the performance drops dramatically on image 2 due to its complex image background.

| AUCs | Detection block size $d$ | | |
|---|---|---|---|
| | 256×256 | 128×128 | 64×64 |
| CFAR | 0.9977 | 0.9985 | 0.9974 |
| Refined | **0.9989** | **0.9995** | **0.9992** |

Table 5.4: AUCs on image 1.

| AUCs | Detection block size $d$ | | |
|---|---|---|---|
| | 256×256 | 128×128 | 64×64 |
| CFAR | 0.9831 | 0.9563 | 0.8651 |
| Refined | **0.9944** | **0.9802** | **0.9092** |

Table 5.5: AUCs on image 2.

| AUCs | Detection block size $d$ | | |
|---|---|---|---|
| | 256×256 | 128×128 | 64×64 |
| CFAR | 0.9587 | 0.9522 | 0.9412 |
| Refined | **0.9882** | **0.9765** | **0.9604** |

Table 5.6: AUCs on image 3.

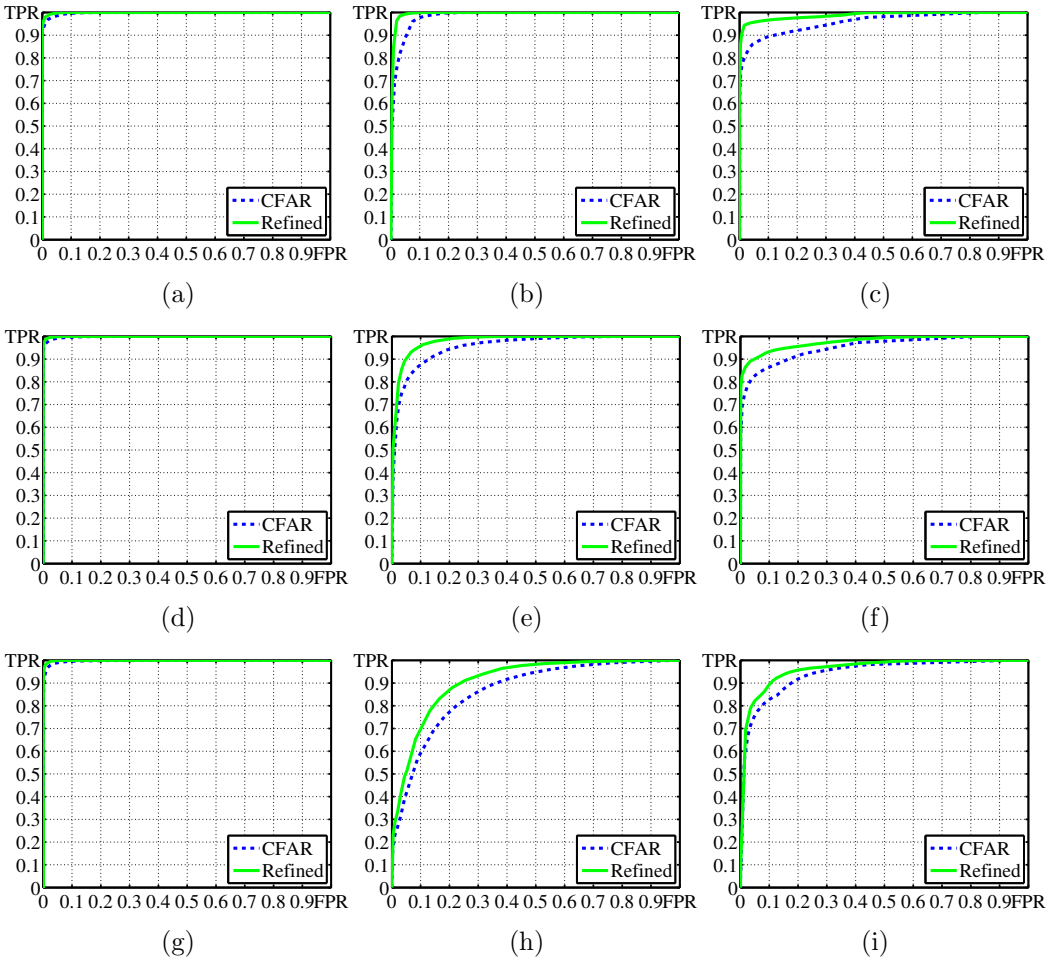



Figure 5.12: ROC curves for three forged images obtained with different detection block sizes. (a) Image 1, $d$=256×256. (b) Image 2, $d$=256×256. (c) Image 3, $d$=256×256. (d) Image 1, $d$=128×128. (e) Image 2, $d$=128×128. (f) Image 3, $d$=128×128. (g) Image 1, $d$=64×64. (h) Image 2, $d$=64×64. (i) Image 3, $d$=64×64.

## 5.5 Conclusion

In this chapter, we have proposed a refining scheme for SPN-based detection of image forgeries. We model the correlation distribution near the boundary across the tampered and non-tampered regions and weight the threshold accordingly to achieve the desired false acceptance rate. Despite some possible false positives (e.g., $\gamma$ is set too small), the overall better performance has been verified in the tasks of detecting both large-scale simulated image forgeries and three different types

of realistic image forgeries. We believe that the proposed refining algorithm will facilitate forensic investigators to get a more accurate detection result.

# CHAPTER 6

## Conclusions and Future Work

The work presented in this thesis has been concerned with digital image forensic techniques based on sensor pattern noise (SPN). Since SPN has been proven to be an effective and robust form of device fingerprint, it has attracted considerable attention due to its attractive characteristics such as uniqueness to individual devices, stability over environmental conditions, and robustness against common image processing operations. It has been successfully used for identifying the source device, linking devices, and image forgery detection. In this thesis, a novel preprocessing algorithm that suppresses the non-unique artifacts, shared amongst cameras subjected to the same or similar in-camera processing procedures, has been proposed in Chapter 3 to reduce the false positives for SPN-based source camera identification. A clustering framework that deals with large-scale device fingerprint databases was also proposed in Chapter 4. It is capable of addressing the $NC \gg SC$ problem without a training process and delivers a good clustering performance. Finally, for the task of image forgery detection based on SPN, a refining algorithm was presented in Chapter 5 to overcome the missing detection problem along the boundary area of the forged and non-forged regions. The following three sections summarize the key contributions and draw conclusions with regard to the three proposed algorithms. The directions for future research will be given in the last section of this chapter.

## 6.1 Preprocessing Reference SPN via Spectrum Equalization

The performance of SPN-based source camera identification heavily relies on the quality of SPN, which might be severely contaminated by image content, dark current noise, shot noise, and periodic artifacts introduced by periodic in-camera processing procedures, such as periodic sensor operations, color interpolation and JPEG compression. These non-unique periodic artifacts are usually shared among the cameras of the same model or brand, and often give rise to false identifications. There are existing approaches aiming at suppressing these undesired periodic artifacts, but we found that these approaches only suppress part of the periodic artifacts and leave room for improvement.

Since the periodic artifacts manifest themselves as salient peaks in the DFT spectrum of the reference SPN, we, therefore, proposed a spectrum equalization algorithm (SEA) in Chapter 3 to detect and suppress the peaks in the spectrum. Combined with six SPN extraction methods, the effectiveness of the proposed algorithm was evaluated on the Dresden image database [53, 91] and compared to the state-of-the-art preprocessing schemes in terms of both overall ROC curve and the kappa statistic computed from a confusion matrix. The experimental results [118] showed that the proposed algorithm can significantly improve the performance depending on the size of image blocks. For example, the true positive rate at the false positive rate of $1\times10^{-3}$ can be improved by $10\% \sim 20\%$ when the image block is small (e.g., $256 \times 256$ pixels). Apart from the effectiveness of the proposed algorithm, we also investigated its robustness against JPEG compression on our own uncompressed image database. We found that SEA tends to be more robust against JPEG compression for medium ($256\times256$ pixels) and small ($128\times128$ pixels) image blocks, but ZM+WF outperforms SEA in the case of large image blocks ($1024\times1024$ pixels).

As mentioned in [53], some unexpected artifacts, which may stem from the dependencies between sensor noise and special camera settings or some advanced in-camera post-processing, were observed in the images taken by Nikon CoolPix S710, FujiFilm FinePix J50 and Casio EX-150. So we conducted independent experiments and presented detailed analyses for these three cameras. Despite no performance improvement on large image blocks, the significantly better performance on medium and small image blocks makes the proposed algorithm still favorable in digital camcorder identification and image/video forgery localization.

## 6.2 Large-Scale Image Clustering Based on Device Fingerprints

In some forensic circumstances, it is desirable to cluster images into a number of groups, each including the images captured by the same device, so that the forensic investigators can link different crime scenes and evidence. This task can be accomplished by resorting to the use of the SPNs (camera fingerprints) extracted from images. Several approaches have been proposed to cluster device fingerprints, but all of them are performed on the pairwise similarity matrix, which could be computationally expensive to obtain for large-scale and high-dimensional camera fingerprint databases. The difficulties of clustering large-scale camera fingerprint databases can be further aggravated when the Number of Classes (i.e., the number of cameras) is much higher than the average Size of Class (i.e., the number of images acquired by each camera). We refer to this as the $NC \gg SC$ problem, which is not uncommon in many practical scenarios.

In view of the limitations of existing clustering algorithms, we proposed a clustering framework for large-scale camera fingerprint databases that is capable of addressing the $NC \gg SC$ problem in Chapter 4. By taking advantage of the dimension reduction and the sparseness of the similarity matrix, we first coarsely

partitioned the entire database into small batches while keeping most of the fingerprints of the same camera in the same batch. When calculating the similarities between fingerprints, we applied the potential-based eviction strategy to guarantee that larger classes will be clustered preferentially. Accurate clustering is then performed on each of the small batches to produce a number of sub-clusters, which will be merged based on an adaptive threshold calculated according to the quality and size of the sub-clusters. The centroids of the merged clusters will be used as "attractors" to classify the remaining fingerprints in the database based on their similarities to the "attractors". The above procedures will be repeated until all fingerprints have been clustered.

The results on the Dresden image database showed that it exhibits high capability of conquering the $NC \gg SC$ problem. For a challenging dataset consisting of 50000 fingerprints with $NC = 2500$ and $SC = 20$, around 80% classes can be identified. Compared to other camera fingerprint clustering algorithms, it has a comparable performance on small datasets, but it is much faster and achieves a better clustering quality than other clustering algorithms. Furthermore, it outputs clusters with high purity, which make it attractive in many practical applications, such as information searching and retrieval.

## 6.3 Refining Image Forgery Detection Based on SPN

SPN serves as the fingerprint of a camera, so image forgeries in the images taken by the camera can be revealed by detecting the absence of the SPN signal. Because SPN is a very weak signal, its reliable detection requires the joint process of a large number of pixels. Therefore, SPN-based image forgery detection usually works in a block-wise manner. The center pixel of a detection block is deemed to be forged if the test statistic calculated on all the pixels within the detection block is smaller than a predefined threshold. However, when the detection block falls near the boundary

between the forged and non-forged regions, the test statistic is a weighting average of two different contributions, which results in missing detection along the boundary. Image level segmentation can be used to alleviate this problem, but it is helpless in the scenario where the "occlusive" forgeries present.

In Chapter 5, we proposed a refining algorithm to address the missing detection problem along the boundary between the forged and non-forged regions. We first detected the image forgeries using the constant false acceptance rate (CFAR) method described in Section 2.3.2. With the initial guess of the boundary between the forged and non-forged regions given by CFAR method, we analyzed how the test statistic changes when the detection block moves across the boundary and adjusted the decision threshold accordingly to correct the falsely identified pixels.

The improvement brought about by the proposed refining algorithm has been confirmed by a large set of random forged images and three realistic forgery examples. We found that the improvement in terms of *area under ROC curve* (AUC) can be as high as 5.4% depending on the size of detection block and the difficulty of detection. Furthermore, the refined detection result apparently fits more closely to the actual shape of the tampered area, which can potentially provide the forensic investigator with more detailed information.

## 6.4 Future Research Directions

This thesis focuses on three tasks of digital image forensics, namely source camera identification, source-oriented image clustering and image forgery detection, by resorting to the use of SPN. Actually, digital image forensics is only a branch of multimedia forensics, which is a much more broad and diverse discipline encompassing the recovery and investigation of multimedia signals (audio, images and videos). A multitude of directions for future work are opening up in front of us. Some possible lines of investigations are as follows.

1. In the SPN-based forensic tasks, a threshold has to be determined for decision-making. This is usually done by applying the Neyman-Pearson criterion based on the inter-class distribution. The problem is that the Neyman-Pearson criterion only concerns about the inter-class distribution but without considering the intra-class distribution. As a result, the threshold given by the Neyman-Pearson criterion is not "optimal" in most cases. Taking SPN-based image forgery detection as an example, the threshold determined by the Neyman-Pearson criterion only guarantees that the false positive rate does not exceed a user-defined value, but it might not be the threshold that "optimally" separates the forged and non-forged regions. A good threshold should be the one concerns both the inter-class and inter-class distribution, the latter of which heavily depends on the quality of SPN. However, this is very challenging because the quality of SPN is affected by many forms of interferences. We presented a method to roughly estimate the quality of SPN in [124] and will continue to improve it.

2. In [124], we modeled the true SPN signal as independent and identically distributed (i.i.d.) White Gaussian Noise in very noisy background and developed an ad-hoc clustering algorithm to address the $NC \gg SC$ problem (i.e., the Number of Classes is much higher than the average Size of Class) in large-scale source oriented image clustering. We may have found a special case for a more generic problem (e.g., when the i.i.d. assumption does not hold), which can be applied to other modality of large-scale datasets and therefore is worthy of further study.

3. As mentioned in Chapter 4, small classes may disappear in the final results due to the inappropriate binarization. The difficulty of determining an appropriate binarization threshold comes from the varying quality of SPN. In [57], $K$-means ($K = 2$) was used to adaptively determine the reference similarity that

separates the inter-class and intra-class distribution. We also found that this problem is also closely related with the problem of clustering data with various densities. We will carry out further investigations to solve this problem.

4. Chen et al. proposed a correlation predictor in [9] to estimate the intra-class distribution. They assumed that there is a linear relationship between the correlations and the extracted image features and their second-order terms (for more details, please refer to Section 2.3.2). However, we found in our experiments that they do not strictly conform to a linear relationship, which is largely responsible for the prediction errors. Therefore, another line of our future research is to design a better and more robust correlation predictor.

# APPENDIX A

## Derivation of Correlation Distribution

To simplify the derivation of the correlation distribution, some assumptions have to be made. Following the assumptions in [125], we assume that each $d$-dimensional fingerprint has the same quality and is standardized to have zero mean and unit variance before calculating the correlation. Presumably, each standardized fingerprint can be modeled as the sum of the true SPN $f_i$ and other interferences $\alpha_i$:

$$F[i] = f_i + \alpha_i, \;\; i = 1, 2, 3, ..., d, \tag{A.1}$$

where $d$ is the length of the estimated fingerprint, $f_i$ follows a normal distribution $\mathcal{N}(0, \sigma^2)$ and $\alpha_i \sim \mathcal{N}(0, 1-\sigma^2)$ is White Gaussian Noise (WGN). For convenience, $\sigma^2$ will be referred to as the quality of the true SPN, while $1-\sigma^2$ will be referred to as the level of the interferences in the standardized fingerprint. We further assume that the fingerprints of two different cameras are independent. The normalized cross correlation (NCC) $\rho$ between two fingerprints or centroids, $\boldsymbol{X}$ and $\boldsymbol{Y}$, is given in Equation (4.3), where $\hat{X}[i]=x_i + \alpha_i$, $\hat{Y}[i]=y_i + \beta_i$, $x_i \sim \mathcal{N}(0, \sigma_x^2)$, $\alpha_i \sim \mathcal{N}(0, 1-\sigma_x^2)$, $y_i \sim \mathcal{N}(0, \sigma_y^2)$, and $\beta_i \sim \mathcal{N}(0, 1-\sigma_y^2)$. Therefore, we can rewrite Equation (4.3) as

$$\rho(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{d} \sum_{i=1}^{d} (x_i y_i + \alpha_i y_i + \beta_i x_i + \alpha_i \beta_i). \tag{A.2}$$

When it comes to the situation of determining whether to merge two clusters, we will consider the two centroids averaged over the $n_x$ fingerprints in one cluster and the $n_y$ fingerprints in the other cluster, respectively. Next, we will derive the distribution of inter-class correlation and intra-class correlation in different scenarios.

### A.0.1 Scenario 1: $n_x$=$n_y$=1, $\sigma_x^2 \neq \sigma_y^2$

In this scenario, we assume the qualities of the true SPN in $\boldsymbol{X}$ and $\boldsymbol{Y}$ are different. Notice that it does not conflict with the assumption that the true SPNs of the same class are of the same quality. Because even the qualities of the true SPN in all individual fingerprints are the same, when different numbers of fingerprints are averaged to estimate the centroids, the qualities of the true SPN in the resultant centroids may vary significantly. In this scenario, although $\boldsymbol{X}$ and $\boldsymbol{Y}$ are referred to as two fingerprints with different qualities, they can actually be viewed as two centroids averaged over two clusters with different numbers of fingerprints. Under this circumstance, we assume $x_i \sim \mathcal{N}(0, \sigma^2)$, $\alpha_i \sim \mathcal{N}(0, 1-\sigma^2)$, $y_i \sim \mathcal{N}(0, \lambda\sigma^2)$ and $\beta_i \sim \mathcal{N}(0, 1-\lambda\sigma^2)$. For two fingerprints of different cameras, using the Central Limit Theorem (CLT), $\rho(\boldsymbol{X}, \boldsymbol{Y})$ approaches to a normal distribution $\mathcal{N}(0, 1/d)$ when $d \to \infty$. But if $\boldsymbol{X}$ and $\boldsymbol{Y}$ are of the same camera, we have $y_i = \sqrt{\lambda} x_i$. Therefore, Equation (A.2) can be rewritten as

$$\rho(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{d} \sum_{i=1}^{d} (\sqrt{\lambda} x_i^2 + \sqrt{\lambda} \alpha_i x_i + \beta_i x_i + \alpha_i \beta_i). \tag{A.3}$$

It is known that $x_i \sim \mathcal{N}(0, \sigma^2)$, therefore $x_i^2/\sigma^2$ follows the Chi-squared distribution with 1 degree of freedom $\chi^2(1)$. We can easily obtain the mean and variance for $x_i^2$: $E[x_i^2] = \sigma^2$, $Var[x_i^2] = 2\sigma^4$. Based on the assumption that $x_i$, $\alpha_i$, and $\beta_i$ are mutually independent, we can easily derive the mean and variance of the $i$th element as

$$\begin{cases} E[\sqrt{\lambda} x_i^2 + \sqrt{\lambda} \alpha_i x_i + \beta_i x_i + \alpha_i \beta_i] = \sqrt{\lambda} \sigma^2 \\ Var[\sqrt{\lambda} x_i^2 + \sqrt{\lambda} \alpha_i x_i + \beta_i x_i + \alpha_i \beta_i] = 1 + \lambda \sigma^4. \end{cases} \tag{A.4}$$

According to CLT, when $d \to \infty$,

$$\rho(\boldsymbol{X}, \boldsymbol{Y}) \xrightarrow{d} \mathcal{N}(\mu_1, \Sigma_1), \tag{A.5}$$

where

$$\begin{cases} \mu_1 = \sqrt{\lambda}\sigma^2 \\ \Sigma_1 = (1 + \lambda\sigma^4)/d. \end{cases} \tag{A.6}$$

### A.0.2 Scenario 2: $n_x > 1$, $n_y > 1$, $\sigma_x^2 \neq \sigma_y^2$

Suppose $\boldsymbol{X}$ and $\boldsymbol{Y}$ are two centroids generated by averaging $n_x$ fingerprints and $n_y$ fingerprints, respectively. In this more complicated and general scenario, if we can figure out the qualities of the true SPN in $\boldsymbol{X}$ and $\boldsymbol{Y}$, then we can use the conclusion of Scenario 1 to determine the distribution of the correlation between $\boldsymbol{X}$ and $\boldsymbol{Y}$. For the centroids from two different classes, we still have the same conclusion $\rho(\boldsymbol{X}, \boldsymbol{Y}) \xrightarrow{d} \mathcal{N}(0, 1/d)$. For two centroids from the same class, we assume that the qualities of the true SPN in all individual fingerprints in the same cluster are the same, $\sigma^2$. If $n$ such fingerprints are averaged before standardization, the quality of the true SPN, $\sigma^2$, remains unchanged, but the level of interferences declines to $(1 - \sigma^2)/n$. So after standardization, the quality of the true SPN in the centroid becomes

$$\frac{\sigma^2}{\sigma^2 + (1 - \sigma^2)/n} = \frac{n\sigma^2}{(n-1)\sigma^2 + 1}. \tag{A.7}$$

Replacing $n$ in Equation (A.7) with $n_x$ and $n_y$, $\sigma^2$ with $\sigma_x^2$ and $\sigma_y^2$ yields the distributions for the $i$th element of the two centroids:

$$\begin{cases} x_i \sim \mathcal{N}(0, n_x\sigma_x^2/[(n_x - 1)\sigma_x^2 + 1]) \\ \alpha_i \sim \mathcal{N}(0, (1 - \sigma_x^2)/[(n_x - 1)\sigma_x^2 + 1]) \end{cases} \tag{A.8}$$

and

$$\begin{cases} y_i \sim \mathcal{N}(0, n_y\sigma_y^2/[(n_y - 1)\sigma_y^2 + 1]) \\ \beta_i \sim \mathcal{N}(0, (1 - \sigma_y^2)/[(n_y - 1)\sigma_y^2 + 1]). \end{cases} \tag{A.9}$$

Following the conclusion of Scenario 1, when $d \to \infty$, the distribution of $\rho$ between two centroids of the same class approaches to a normal distribution

$$\rho(\boldsymbol{X}, \boldsymbol{Y}) \xrightarrow{d} \mathcal{N}(\mu_2, \Sigma_2), \tag{A.10}$$

where

$$\begin{cases} \mu_2 = \sqrt{\frac{n_x n_y \sigma_x^2 \sigma_y^2}{[(n_x-1)\sigma_x^2+1][(n_y-1)\sigma_y^2+1]}} \\ \Sigma_2 = \frac{n_x n_y \sigma_x^2 \sigma_y^2+[(n_x-1)\sigma_x^2+1][(n_y-1)\sigma_y^2+1]}{[(n_x-1)\sigma_x^2+1][(n_y-1)\sigma_y^2+1]d}. \end{cases} \tag{A.11}$$

By setting $n_x$=$n_y$=1, Equation (A.11) becomes Equation (A.6). But in practice, $\lambda$ varies from different fingerprints, making the intra-class correlation distribution a Gaussian mixture distribution rather than a unimodal Gaussian distribution. The mean indicated in Equation (A.11) indicates where most of the correlations are scattered around.

# Bibliography


[1] J. Lukas, J. Fridrich, and M. Goljan. Digital camera identification from sensor pattern noise. *IEEE Transactions on Information Forensics and Security*, 1(2):205–214, 2006.

[2] M.K. Mhak, I. Kozintsev, and K. Ramchandran. Spatially adaptive statistical modeling of wavelet image coefficients and its application to denoising. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 6, pages 3253–3256, Mar 1999.

[3] C.-T. Li. Source camera identification using enhanced sensor pattern noise. *IEEE Transactions on Information Forensics and Security*, 5(2):280–287, 2010.

[4] X. Kang, J. Chen, K. Lin, and P. Anjie. A context-adaptive SPN predictor for trustworthy source camera identification. *EURASIP Journal on Image and Video Processing*, 2014(1):1–11, 2014.

[5] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080–2095, Aug 2007.

[6] C.-T. Li and Y. Li. Color-decoupled photo response non-uniformity for digital image forensics. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(2):260–271, 2012.

[7] Y. Hu, C. Jian, and C. T. Li. Using improved imaging sensor pattern noise for source camera identification. In *IEEE International Conference on Multimedia and Expo (ICME)*, pages 1481–1486, July 2010.

[8] J. Lukáš, J. Fridrich, and M. Goljan. Detecting digital image forgeries using sensor pattern noise. In *SPIE*, pages 362–372, 2006.

[9] M. Chen, J. Fridrich, M. Goljan, and J. Lukás. Determining image origin and integrity using sensor noise. *IEEE Transactions on Information Forensics and Security*, 3(1):74–90, 2008.

[10] H. T. Sencar and N. Memon. Overview of state-of-the-art in digital image forensics. *Algorithms, Architectures and Information Systems Security*, 3:325–348, 2008.

[11] M. U. Celik, G. Sharma, E. Saber, and A. M. Tekalp. Hierarchical watermarking for secure image authentication with localization. *IEEE Transactions on Image Processing*, 11(6):585–595, Jun 2002.

[12] P. Wong and N. Memon. Secret and public key image watermarking schemes for image authentication and ownership verification. *IEEE transactions on image processing*, 10(10):1593–1601, 2001.

[13] J. Fridrich and M. Goljan. Images with self-correcting capabilities. In *International Conference on Image Processing (ICIP)*, volume 3, pages 792–796, 1999.

[14] X. Zhu, A. T. Ho, and P. Marziliano. A new semi-fragile image watermarking with robust tampering restoration using irregular sampling. *Signal Processing: Image Communication*, 22(5):515–528, 2007.

[15] X. Zhao, A. T. Ho, H. Treharne, V. Pankajakshan, C. Culnane, and W. Jiang. A novel semi-fragile image watermarking, authentication and self-restoration technique using the slant transform. In *International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIHMSP)*, volume 1, pages 283–286, 2007.

[16] X. Zhao, A. T. Ho, and Y. Q. Shi. Image forensics using generalised Benford's law for accurate detection of unknown JPEG compression in watermarked images. In *International Conference on Digital Signal Processing*, pages 1–8, 2009.

[17] A. T. Ho. Semi-fragile watermarking and authentication for law enforcement applications. In *International Conference on Innovative Computing, Information and Control*, pages 286–286, 2007.

[18] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[19] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Workshop on the Theory and Application of Cryptographic Techniques*, pages 10–18, 1984.

[20] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.

[21] D. Johnson, A. Menezes, and S. Vanstone. The elliptic curve digital signature algorithm (ECDSA). *International Journal of Information Security*, 1(1):36–63, 2001.

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

[23] P. Vaidyanathan. Quadrature mirror filter banks, m-band extensions and perfect-reconstruction techniques. *IEEE ASSP Magazine*, 4(3):4–20, Jul 1987.

[24] M. Vetterli. A theory of multirate filter banks. *IEEE Transactions on Acoustics, Speech, and Signal Processing (ICASSP)*, 35(3):356–372, Mar 1987.

[25] M. Kharrazi, H T Sencar, and N. Memon. Blind source camera identification. In *IEEE International Conference on Image Processing (ICIP)*, volume 1, pages 709–712, 2004.

[26] J. Lukáš and J. Fridrich. Estimation of primary quantization matrix in double compressed JPEG images. In *Digital Forensic Research Workshop*, pages 5–8, 2003.

[27] A. C. Popescu and H. Farid. Statistical tools for digital forensics. In *Information Hiding*, pages 128–147, 2004.

[28] A. C. Popescu and H. Farid. Exposing digital forgeries by detecting traces of resampling. *IEEE Transactions on Signal Processing*, 53(2):758–767, 2005.

[29] M. Kirchner. Fast and reliable resampling detection by spectral analysis of fixed linear predictor residue. In *ACM workshop on Multimedia and security*, pages 11–20, 2008.

[30] M. C. Stamm and K. R. Liu. Forensic detection of image manipulation using statistical intrinsic fingerprints. *IEEE Transactions on Information Forensics and Security*, 5(3):492–506, Sept 2010.

[31] M. C. Stamm and K. R. Liu. Forensic estimation and reconstruction of a contrast enhancement mapping. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICAPSS)*, pages 1698–1701, 2010.

[32] G. Cao, Y. Zhao, and R. Ni. Forensic estimation of gamma correction in digital images. In *IEEE International Conference on Image Processing (ICIP)*, pages 2097–2100, 2010.

[33] X. Lin, C.-T. Li, and Y. Hu. Exposing image forgery through the detection of contrast enhancement. In *IEEE International Conference on Image Processing (ICIP)*, pages 4467–4471, 2013.

[34] X. Lin, X. Wei, and C.-T. Li. Two improved forensic methods of detecting contrast enhancement in digital images. In *IS&T/SPIE Electronic Imaging*, pages 90280X–90280X, 2014.

[35] J. Fridrich, D. Soukal, and J. Lukáš. Detection of copy-move forgery in digital images. In *Digital Forensic Research Workshop*, 2003.

[36] A. C. Popescu and H. Farid. Exposing digital forgeries by detecting duplicated image regions. Technical Report TR2004-515, Department of Computer Science, Dartmouth College, 2004.

[37] G. Li, Q. Wu, D. Tu, and S. Sun. A Sorted Neighborhood Approach for Detecting Duplicated Regions in Image Forgeries Based on DWT and SVD. In *IEEE International Conference on Multimedia and Expo*, pages 1750–1753, July 2007.

[38] M. K. Johnson and H. Farid. Exposing digital forgeries by detecting inconsistencies in lighting. In *Workshop on Multimedia and Security*, pages 1–10, 2005.

[39] E. Kee, J. F. O'brien, and H. Farid. Exposing photo manipulation from shading and shadows. *ACM Transactions on Graphics*, 33(5):1–21, 2014.

[40] Kai San Choi, Edmund Y Lam, and Kenneth KY Wong. Automatic source camera identification using the intrinsic lens radial distortion. *Optics express*, 14(24):11551–11565, 2006.

[41] A. Swaminathan, M. Wu, and K. R. Liu. Nonintrusive component forensics of visual sensors using output images. *IEEE Transactions on Information Forensics and Security*, 2(1):91–106, 2007.

[42] Z. Lin, Wang R., Tang X., and Shum H.-Y. Detecting doctored images using camera response normality and consistency. In *IEEE Computer Society Con-*

*ference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 1087–1092, June 2005.

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

[44] H. Farid. Exposing Digital Forgeries from JPEG Ghosts. *Transactions on Information Forensics and Security*, 4(1):154–160, March 2009.

[45] X. Kang, Y. Li, Z. Qu, and J. Huang. Enhancing source camera identification performance with a camera reference phase sensor pattern noise. *IEEE Transactions on Information Forensics and Security*, 7(2):393–402, 2012.

[46] S. McCloskey. Confidence weighting for sensor fingerprinting. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–6, June 2008.

[47] F. Gharibi, F. Akhlaghian, J. RavanJamjah, and B. ZahirAzami. Using the local information of image to identify the source camera. In *IEEE International Symposium on Signal Processing and Information Technology*, pages 515–519, 2010.

[48] L.-H. Chan, N.-F. Law, and W.-C. Siu. A confidence map and pixel-based weighted correlation for prnu-based camera identification. *Digital Investigation*, 10(3):215–225, 2013.

[49] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[50] A. Lawgaly, F. Khelifi, and A. Bouridane. Weighted averaging-based sensor pattern noise estimation for source camera identification. In *IEEE International Conference Image Processing (ICIP)*, pages 5357–5361, Oct 2014.

[51] G. Wu, X. Kang, and K. R. Liu. A context adaptive predictor of sensor pattern noise for camera source identification. In *IEEE International Conference on Image Processing (ICIP)*, pages 237–240, 2012.

[52] Y. Hu, B. Yu, and C. Jian. Source camera identification using large components of sensor pattern noise. In *International Conference on Computer Science and its Applications*, pages 291–294, 2009.

[53] T. Gloe, S. Pfennig, and M. Kirchner. Unexpected artefacts in PRNU-based camera identification: A 'Dresden Image Database' Case-Study. In *ACM Workshop on Multimedia and Security*, pages 109–114, 2012.

[54] M. Goljan. Digital camera identification from images estimating false acceptance probability. In *Digital Watermarking*, volume 5450, pages 454–468. 2009.

[55] M. Goljan, J. Fridrich, and T. Filler. Large scale test of sensor fingerprint camera identification. In *IS&T/SPIE Electronic Imaging*, pages 72540I–72540I, 2009.

[56] Camera fingerprint-matlab implementation, 2012. http://dde.binghamton.edu/download/camera_fingerprint/.

[57] C.-T. Li. Unsupervised classification of digital images using enhanced sensor pattern noise. In *IEEE International Symposium on Circuits and Systems*, pages 3429–3432, May 2010.

[58] B.-B. Liu, H.-K. Lee, Y. Hu, and C.-H. Choi. On classification of source cameras: A graph based approach. In *IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–5, Dec 2010.

[59] S. X. Yu and J. Shi. Multiclass spectral clustering. In *the 9th IEEE International Conference Computer Vision*, pages 313–319, 2003.

[60] R. Caldelli, I. Amerini, F. Picchioni, and M. Innocenti. Fast image clustering of unknown source images. In *IEEE International Workshop on Information Forensics and Security*, pages 1–5, Dec 2010.

[61] L. J. G. Villalba, A. L. S. Orozco, and J. R. Corripio. Smartphone image clustering. *Expert System with Applications*, 42(4):1927–1940, 2015.

[62] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[63] R. T. Ng and J. Han. CLARANS: A method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(5):1003–1016, 2002.

[64] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, volume 96, pages 226–231, 1996.

[65] S. Guha, R. Rastogi, and K. Shim. CURE: an efficient clustering algorithm for large databases. In *ACM SIGMOD Record*, volume 27, pages 73–84, 1998.

[66] S. Guha, R. Rastogi, and K. Shim. ROCK: A robust clustering algorithm for categorical attributes. In *of 15th International Conference on Data Engineering*, pages 512–521, 1999.

[67] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: an efficient data clustering method for very large databases. In *ACM Sigmod Record*, volume 25, pages 103–114, 1996.

[68] G. Karypis, E.-H. Han, and V. Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75, 1999.

[69] G. Cao, Y. Zhao, R. Ni, and X. Li. Contrast enhancement-based forensics

in digital images. *IEEE Transactions on Information Forensics and Security*, 9(3):515–525, March 2014.

[70] X. Pan and S. Lyu. Region duplication detection using image feature matching. *IEEE Transactions on Information Forensics and Security*, 5(4):857–867, 2010.

[71] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, and G. Serra. A sift-based forensic method for copy-move attack detection and transformation recovery. *IEEE Transactions on Information Forensics and Security*, 6(3):1099–1110, Sept 2011.

[72] G. Chierchia, G. Poggi, C. Sansone, and L. Verdoliva. A Bayesian-MRF approach for PRNU-based image forgery detection. *IEEE Transactions on Information Forensics and Security*, 9(4):554–567, 2014.

[73] P. L. Combettes and J.-C. Pesquet. Primal-dual splitting algorithm for solving inclusions with mixtures of composite, lipschitzian, and parallel-sum type monotone operators. *Set-Valued and Variational Analysis*, 20(2):307–330, 2012.

[74] G. Chierchia, S. Parrilli, G. Poggi, L. Verdoliva, and C. Sansone. Prnu-based detection of small-size image forgeries. In *IEEE International Conference on Digital Signal Processing*, pages 1–6, July 2011.

[75] G. Chierchia, D. Cozzolino, G. Poggi, C. Sansone, and L. Verdoliva. Guided filtering for prnu-based localization of small-size image forgeries. In *IEEE International Conference Acoustics, Speech, Signal Processing (ICASSP)*, pages 6231–6235, 2014.

[76] K. He, J. Sun, and X. Tang. Guided image filtering. In *European Conference Computer Vision*, pages 1–14. Springer, 2010.

[77] K. Kurosawa, K. Kuroki, and N. Saitoh. Ccd fingerprint method-identification of a video camera from videotaped images. In *IEEE International Conference on Image Processing (ICIP)*, volume 3, pages 537–540, 1999.

[78] Z. J. Geradts, J. Bijhold, M. Kieft, K. Kurosawa, K. Kuroki, and N. Saitoh. Methods for identification of images acquired with digital cameras. In *Enabling Technologies for Law Enforcement*, pages 505–512, 2001.

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

[80] M. J. Sorell. Digital camera source identification through JPEG quantisation. *Multimedia forensics and security*, pages 291–313, 2008.

[81] E. J. Alles, Z. J. Geradts, and C. J. Veenman. Source Camera Identification for Heavily JPEG Compressed Low Resolution Still Images. *Journal of forensic sciences*, 54(3):628–638, 2009.

[82] K. San Choi, E. Y. Lam, and K. K. Wong. Source camera identification using footprints from lens aberration. In *Electronic Imaging*, pages 60690J–60690J, 2006.

[83] T. V. Lanh, S. Emmanuel, and M. S. Kankanhalli. Identifying source cell phone using chromatic aberration. In *IEEE International Conference on Multimedia and Expo*, pages 883–886, July 2007.

[84] O. Celiktutan, B. Sankur, and I. Avcibas. Blind identification of source cell-phone model. *IEEE Transactions on Information Forensics and Security*, 3(3):553–566, Sept 2008.

[85] T. Filler, J. Fridrich, and M. Goljan. Using sensor pattern noise for camera

model identification. In *IEEE International Conference on Image Processing (ICIP)*, pages 1296–1299, Oct 2008.

[86] G. Chierchia, S. Parrilli, G. Poggi, C. Sansone, and L. Verdoliva. On the influence of denoising in prnu based forgery detection. In *ACM Workshop on Multimedia in Forensics, Security and Intelligence*, pages 117–122, NY, USA, 2010.

[87] D. Cozzolino, D. Gragnaniello, and L. Verdoliva. A novel framework for image forgery localization. *arXiv preprint arXiv:1311.6932*, 2013.

[88] C.-C. Hsu, T.-Y. Hung, C.-W. Lin, and C.-T. Hsu. Video forgery detection using correlation of noise residue. In *IEEE Workshop on Multimedia Signal Processing*, pages 170–174, 2008.

[89] M. Chen, J. Fridrich, M. Goljan, and J. Lukáš. Source digital camcorder identification using sensor photo response non-uniformity. In *Electronic Imaging 2007*, pages 65051G–65051G, 2007.

[90] A. El Gamal, B. A. Fowler, H. Min, and X. Liu. Modeling and estimation of fpn components in cmos image sensors. In *Photonics Electronic Imaging*, pages 168–177, 1998.

[91] T. Gloe and R. Böhme. The 'Dresden Image Database' for Benchmarking Digital Image Forensics. *Journal of Digital Forensic Practice*, 3(2-4):150–159, 2010.

[92] Independent JPEG Group et al. Independent JPEG groups free JPEG software, March 1998. http://www.ijg.org/.

[93] Bm3d matlab software, January 2014. http://www.cs.tut.fi/~foi/GCF-BM3D/.

[94] C.-T. Li and R. Satta. Empirical investigation into the correlation between vignetting effect and the quality of sensor pattern noise. *IET Computer Vision*, 6(6):560–566, 2012.

[95] J. Carletta. Assessing agreement on classification tasks: the kappa statistic. *Computational linguistics*, 22(2):249–254, 1996.

[96] P. Li, T. J. Hastie, and K. W. Church. Very sparse random projections. In *ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, pages 287–296, 2006.

[97] I. S. Dhillon, Y. Guan, and B. Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE Transactions Pattern Analysis Machine Intelligence*, 29(11):1944–1957, 2007.

[98] Stijn M. Van D. *Graph clustering by flow simulation.* PhD thesis, University of Utrecht, Netherlands, 2000.

[99] D. Rafiei and A. Mendelzon. Similarity-based queries for time series data. In *ACM SIGMOD Record*, volume 26, pages 13–25, 1997.

[100] R. Agrawal, C. Faloutsos, and A. N. Swami. Efficient Similarity Search In Sequence Databases. In *International Conference Foundations Data Organization Algorithms*, pages 69–84, 1993.

[101] D. Wu, A. Singh, D. Agrawal, A. El Abbadi, and T. R. Smith. Efficient retrieval for browsing large image databases. In *International Conference Information Knowledge Management*, pages 11–18, 1996.

[102] K.-P. Chan and A.-C. Fu. Efficient time series matching by wavelets. In *the 15th International Conference Data Engineering*, pages 126–133, 1999.

[103] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Dimensionality

reduction for fast similarity search in large time series databases. *Knowledge Information System*, 3(3):263–286, 2001.

[104] I. Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[105] J. B. Tenenbaum, V. De Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[106] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *International Journal Computer Vision*, 70(1):77–90, 2006.

[107] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[108] D. Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal Computer System Science*, 66(4):671–687, 2003.

[109] George Karypis and Vipin Kumar. Metis: a software package for partitioning unstructured graphs. *International Cryogenics Monograph*, pages pgs. 121–124, 1998.

[110] Chris Walshaw and Mark Cross. Mesh partitioning: a multilevel balancing and refinement algorithm. *SIAM Journal on Scientific Computing*, 22(1):63–80, 2000.

[111] François Pellegrini and Jean Roman. Scotch: A software package for static mapping by dual recursive bipartitioning of process and architecture graphs. In *Proceedings of the International Conference and Exhibition on High-Performance Computing and Networking*, pages 493–498, London, UK, 1996. Springer-Verlag.

[112] I. S. Dhillon, Y. Guan, and B. Kulis. A Unified View of Kernel k-means, Spectral Clustering and Graph Cuts. Technical Report TR-04-25, Department of Computer Sciences, University of Texas, Austin, Texas, February 2004.

[113] A. Y. Ng, M. I. Jordan, Y. Weiss, et al. On Spectral Clustering: Analysis and an algorithm. *Advances in Neural Information Processing System*, 2:849–856, 2002.

[114] G. J. Bloy. Blind camera fingerprinting and image clustering. *IEEE Transactions Pattern Analysis Machine Intelligence*, 30(3):532–534, 2007.

[115] Josef E. Source Camera Classification and Clustering from Sensor Pattern Noise. Master's thesis, Chalmers University of Technology, Sweden, 2012.

[116] D. Valsesia, G. Coluccia, T. Bianchi, and E. Magli. Compressed fingerprint matching and camera identification via random projections. *IEEE Transactions on Information and Forensics Security*, 10(7):1472–1485, 2015.

[117] G. Chierchia, S. Parrilli, G. Poggi, L. Verdoliva, and C. Sansone. PRNU-based detection of small-size image forgeries. *IEEE International Conference Digital Signal Processing*, pages 1–6, 2011.

[118] X. Lin and C.-T. Li. Preprocessing Reference Sensor Pattern Noise via Spectrum Equalization. *IEEE Transactions Information Forensics and Security*, 11(1):126–140, 2016.

[119] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Annual Conference on Computer Graphics and Interactive Techniques*, pages 417–424, 2000.

[120] A. Criminisi, P. Pérez, and K. Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on Image Processing*, 13(9):1200–1212, 2004.

[121] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision*, volume 2, pages 1033–1038, 1999.

[122] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *Annual Conference on Computer Graphics and Interactive Techniques*, pages 341–346, 2001.

[123] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *Annual Conference on Computer Graphics and Interactive Techniques*, pages 479–488, 2000.

[124] X. Lin and C.-T. Li. Large-scale image clustering based on device fingerprints. *Accepted for publication in IEEE Transactions on Information Forensics and Security*, 2016.

[125] J. Fridrich and M. Goljan. Derivation of ROCs for Composite Fingerprints and Sequential Trimming. Technical report, Department of Electrical and Computer Engineering, Binghamton University, Binghamton, NY, USA, January 2010.