Common Source Identification of Images in Large Databases

Floris Gisolf[a], Pelle Barens[a], Ewald Snel[a], Anwar Malgoezar[a], Mark Vos[a], Arjan Mieremet[a], Zeno Geradts[a]

[a]*Department of Digital Technology and Biometry, Netherlands Forensic Institute Laan van Ypenburg 6, 2497 GB, The Hague, The Netherlands*

Abstract

Photo-response non-uniformity noise patterns are a robust way to identify the source of an image. However, identifying a common source of images in a large database is almost impractical due to long computation times. In this paper a solution for large volume digital camera identification is proposed, which combines, and sometimes slightly modifies, existing methods for a 500 times faster common source identification. Single image comparisons are often plagued by considerable noise contamination from scene content and random noise, which makes it harder to accomplish reliable common source identification. Therefore a new method is introduced that can increase true positive rates by more than 45% at very low computation costs. Analysis of real data from a fraud case shows the effectiveness of the proposed method. As a whole the proposed solution makes it possible to analyze a large database in forensically relevant time, without resorting to large and expensive computer clusters.

Keywords: digital camera identification, large scale common source identification, Photo-Response Non-Uniformity, digital forensics

Email addresses: fgisolf@hotmail.com (Floris Gisolf), pelle@holmes.nl (Pelle Barens), ewald@holmes.nl (Ewald Snel), a.malgoezar@gmail.com (Anwar Malgoezar), mark.vos1@live.nl (Mark Vos), arjan@holmes.nl (Arjan Mieremet), z.geradts@nfi.minvenj.nl (Zeno Geradts)

1. Introduction

A robust way to identify the source of an image is the use of Photo Response Non-Uniformity (PRNU) noise [2]. PRNU is noise in images caused by imperfections of the image sensor. This causes the pixels on the image sensor to give a slightly higher or lower signal than other pixels which receive the same amount of radiation. This fingerprint-like noise is approximately the same in every image that a camera takes and can therefore be used to identify the camera.

The basic process of digital camera identification can be seen as a three step process:

1. PRNU noise extraction. The PRNU noise can be extracted from an image using a denoising filter.
2. Estimation of Sensor Pattern Noise (SPN), also known as the camera fingerprint. The PRNU pattern of multiple images originating from the same source are combined to estimate the SPN.
3. Comparison of the SPN to the PRNU noise of an image under investigation.

Many cases, such as child pornography cases, often deal with databases containing thousands to millions of images. It is often unknown which images originate from the same source, and no suspect cameras are available. This means that step 2 of the camera identification process cannot be performed. It is of interest to be able to identify which images originate from the same source because this can link cases or lead to a suspect.

For a database of N images, approximately $N^2/2$ comparisons have to be made to find out which images originate from the same source. Considering that child pornography cases sometimes deal with collections of more than a million images, it is not surprising that this will take a huge amount of computation time. It is therefore of importance to minimize the computation time per comparison. This minimization can be achieved either by using a fast computer or by reducing the complexity of the calculations. This paper discusses the second approach.

Several methods have been proposed to speed up the process of camera identification since its introduction in 2006 [2]. In this paper we aim for high speed common source identification on a standard desktop computer. Several methods for decreasing computation time will be discussed. Some of these methods were optimized for cases where a suspect camera is available,

so these had to be adjusted for use with single image comparisons. As will become clear in the next section of this paper, decreasing computation time is a trade-off between speed and the accuracy of the common source identification. A new method is proposed to counter this reduction in accuracy. This method clusters images with a common source by employing a novel approach to the second step of digital camera identification, applicable even when no suspect camera is available.

Section 2 describes all the steps involved in reducing computation time and the novel clustering method for increasing accuracy. The experimental setup is detailed in section 3. Section 4 will have the experiments, the results and a discussion of the results, section 5 shows all methods applied to a real case, and finally, section 6 summarizes and concludes the results.

2. Methods for high speed common source identification

The common source identification performed can be seen as an 8 step process:

- Step 1 - Grayscale conversion
- Step 2 - PRNU noise extraction
- Step 3 - Zero-mean
- Step 4 - Wiener filter
- Step 5 - Digestion
- Step 6 - Quantization
- Step 7 - Comparison
- Step 8 - Clustering

Steps 1–6 are the derivation of the PRNU pattern of a single image. Step 7 is the comparison between different PRNU patterns and finally in step 8 images are clustered into groups for which it is very likely that they originate from a common source. Below, each individual step is explained in more detail.

2.1. Step 1 - Grayscale conversion

Digital images are mostly color images. To reduce the complexity of the calculations, images are converted from RGB (red, green and blue) into grayscale (intensity). This conversion reduces the amount of data by a factor of 3. Conversion to grayscale has been suggested before [11], and it is shown in Section 4 that this conversion does not reduce the accuracy of common source identification.

2.2. Step 2 - PRNU pattern extraction

Having obtained a database DB with a collection of grayscale-images, scene content now has to be filtered out from these images such that the PRNU of the images remains. To obtain the PRNU pattern, the images are first denoised and this is subtracted from the original images:

$$W_{g,i,j} = I_{g,i,j} - F(I_{g,i,j}), \tag{1}$$

where $I_{g,i,j}$ is intensity value of pixel i, j of the g^{th} grayscale image in the database. F is the denoising filter, $F(I_{g,i,j})$ is the denoised image and $W_{g,i,j}$ is the PRNU pattern of image $I_{g,i,j}$.

Several filters have been proposed over the past few years, such as the Wavelet [2], Adaptive Spatial filtering [11], anisotropic filtering [12] and First Step Total Variation (FSTV) [13]. In [13] it was shown that FSTV gave the best results with the lowest computation time, therefore this filter is used in this paper. FSTV is denoted as

$$W_g = -\nabla \cdot \left(\frac{\nabla I_g}{|\nabla I_g|_\epsilon} \right), \tag{2}$$

where I_g is the observed intensity for pixels of a noisy image, ∇ the gradient operator and ϵ a small positive paramater to avoid singularities. For readability the image dimensions i, j are left out in 1 and the remainder of this paper.

2.3. Step 3 - Zero-mean

At this point a PRNU pattern is obtained. This pattern however could contain linear patterns due to color interpolation [3]. The presence of linear patterns increases the correlation between images taken by different cameras which use the same post-processing. Therefore, after the extraction of the

PRNU pattern, a zero-mean operation is performed. The zero-mean operation makes sure that the even and uneven subsets of columns and rows in a checkerboard pattern become zero and in this way remove linear patterns.

2.4. Step 4 - Wiener filter

Sometimes, images suffer from JPEG compression artifacts. The Wiener filter is used in the Fourier domain to remove these artifacts, as proposed by Chen et al. [3]. By applying this operation, the correlation between images taken by different cameras using the same JPEG compression is decreased. This in turn reduces the chance of false positives.

2.5. Step 5 - Digestion

The next step in reducing the size of W_g is to digest the signal. This idea was proposed by [6] and [4]. Before this method can be applied to single image comparisons some modifications are necessary; but first the original idea is explained. Digestion reduces the size of the PRNU pattern by selecting a fixed percentage of the total number of pixels. The selected pixels are those with the highest absolute PRNU values of W_g.

$$w_g = digest(W_g) \tag{3}$$

Where w_g is a vector with the digest of W_g, where only the selected pixel values and their respective locations are stored. In the original idea of [6] and [4], an SPN is assumed to be available. The SPN is digested, saving the relevant pixel values and locations. Then the PRNU pattern of an image under investigation is extracted and only those pixels at the same location as the SPN digest are stored and used for comparison. They showed that the SPN could be reduced to 1-4% of the number of pixels of its original size, without lowering the accuracy of the camera identification process.

In our case no SPN is available, meaning that a slightly alternative approach has to be taken. Each PRNU pattern of the images in a collection is digested, saving the highest absolute values and their respective positions. These absolute highest values likely have different locations for each image, meaning that there is only partial (or possibly even no) overlap of pixel locations between images. This will have an impact on Step 7, where the digested PRNU patterns are compared. When comparing two PRNU patterns, only those pixels that have the same location can be used. To reduce the chance of no overlapping pixels, a fixed number of 1 210 000 pixels is selected; although this number is arbitrarily chosen, it is generally higher than the 1-4%

suggested by previous research. The result of digestion is a vector with the pixel values and their respective locations for each PRNU pattern.

2.6. Step 6 - Quantization

In 2012 Bayram et al. proposed the use of quantization. [9] In their paper it was shown that keeping only the sign information (binarization) of the PRNU pattern still contained enough information for digital camera identification. This means that every pixel has the size of 1 bit, instead of 32 bits (single precision).

While only speculation at this point, it seems obvious that less bits results in a lower accuracy. In general, a trade-off between accuracy and speed has to be made. A rather arbitrary value of 6 bits per pixel was chosen. This combination of digestion and quantization allows for an efficient way of storing pixel values and their locations by using run-length encoding (RLE). Each pixel value used in the digest is stored, then followed by the number of unused pixels until the next used pixel value. Future research could attend to refine this trade-off; it is possible that a lower bit value results in lower computation time without sacrificing accuracy.

2.7. Summary of PRNU pattern formation

To summarize steps 1–6: first an image will be converted to grayscale, then the PRNU noise pattern will be extracted using FSTV. A zero-mean operation and the Wiener filter are then applied to reduce the chance of false positives. A major reduction is achieved with digestion, using 1.2 million absolute highest values, and this will be saved using quantization of 6 bits per pixel.

As an example, the PRNU pattern of a 5 megapixel color image that has not undergone size reduction is approximately 57 MB. After size reduction, the remaining PRNU pattern is approximately 2 MB. This reduction in size gives us a huge advantage in read/write times when saving and loading the PRNU patterns. It also reduces the amount of disc space needed, making it more desktop computer-friendly.

2.8. Step 7 - Comparison

Once the digested PRNU patterns of the images in a database have been extracted, they are compared to each other. Each comparison results in a similarity score. Several different methods have been proposed over the years for calculating this similarity score, such as Peak-to-Correlation Energy Ratio

(PCE) [1] and Normalized Cross Correlation (NCC). While PCE is more accurate [1], NCC is used because of several other advantages, namely:

- NCC is computationally faster than PCE, because it is less complex.
- NCC can be applied directly to the digested PRNU pattern, whereas for PCE the digested pattern has to be restored to a two-dimensional array, where any non-indexed pixel is set to 0. This further increases the computation time.
- NCC can be used in the final step: clustering.

NCC is defined as

$$NCC(w_g, w_h) = \frac{1}{n} \sum \frac{(w_g - \overline{w}_g) \times (w_h - \overline{w}_h)}{\sigma_{w_g} \sigma_{w_h}} \tag{4}$$

where

$$\overline{w}_g = \frac{1}{n} \sum w_g \tag{5}$$

$$\sigma_{w_g} = \sqrt{\frac{1}{n} \sum (w_g - \overline{w}_g)^2} \tag{6}$$

$$\overline{w}_h = \frac{1}{n} \sum w_h \tag{7}$$

$$\sigma_{w_h} = \sqrt{\frac{1}{n} \sum (w_h - \overline{w}_h)^2} \tag{8}$$

where w_g and w_h are the digested PRNU patterns of two images, n is the number of pixels of the images, and σ the standard deviation.

To further reduce computation time, a simplified version of the NCC is worked out. Recall that, a zero-mean operation is performed on the PRNU pattern. This implies that both $\overline{w}_g$ and $\overline{w}_h$ are 0. Furthermore, the division by n is obsolete, because all patterns have the same number of pixels (1 210 000). The simplified NCC formula becomes

$$NCC(w_g, w_h) = \sum \frac{w_g \times w_h}{\sigma_{w_g} \sigma_{w_h}}. \tag{9}$$

Once all N digested PRNU-patterns of the N images in the database have been compared to each other, an $N \times N$ matrix with NCC scores is obtained. This matrix is symmetric along its diagonal and essential for the last step: clustering of images into groups with a common source.

2.9. Step 8 - Clustering

The final and most important step is the clustering of the images into groups with a common source. The clustering process is an iterative process. At the start of the clustering process, the NCC-matrix is $N \times N$ and consists of N clusters each containing one image. During each iteration step the two clusters with the highest correlation are selected. If the correlation is below a predefined threshold, the clustering process stops, otherwise the clusters are merged, meaning that all images within these clusters are said to originate from a common source. Of course, when clusters are merged, the NCC-matrix shrinks and the NCC-values must be updated. This can be done by averaging all the images within the newly formed cluster and then perform steps 1–7 to calculate new NCC values. This is, however, computationally intensive, especially in the case of large databases. Fortunately, a more direct method is possible. Instead of averaging the original images, the NCC-values of the images are averaged. That is, when two clusters are merged, the NCC between this newly formed cluster and another cluster is calculated according:

$$\begin{aligned} NCC(C_{1,2}, C_3) = f \times NCC(C_1, C_3) \\ + (1 - f) \times NCC(C_2, C_3) \end{aligned} \tag{10}$$

with $C_{1,2}$ being the merged cluster of C_1 and C_2, and C_3 representing another cluster. The parameter f is a weighting function defined as

$$f = \frac{N_1}{N_1 + N_2} \tag{11}$$

with N_1, N_2 the number of images in cluster C_1 and C_2, respectively. f compensates for the different number of images in the two clusters, to ensure that each NCC value has the same weight.

To clarify things, an example is given below with a database consisting of four images. From each of these four images a digested PRNU-pattern is created as described by steps 1 – 6; w_1, w_2, w_3 and w_4. Step 7 of the process calculates the NCC-values between all pairs of images resulting in a 4×4 matrix of NCC-values (Table 1).

Table 1: Example matrix with NCC values

	w_1	w_2	w_3	w_4
w_1	1	0.0040	0.0069	0.0003
w_2	0.0040	1	0.0072	0.0010
w_3	0.0069	0.0072	1	0.0008
w_4	0.0003	0.0010	0.0008	1

Notice that the diagonal elements are equal to 1 and the matrix is symmetric around its diagonal. The clustering process starts with four clusters, each containing 1 image (actually 1 digested PRNU-pattern).

During the first iteration step, it is found that the highest correlation is found between w_2 en w_3 (0.0072) and since this is above the predefined threshold (assumed to be 0.0020) it is stated that w_2 and w_3 have a common source. Clusters 2 and 3 are therefore merged. Now, the NCC-matrix needs to be updated such that w_2 and w_3 is replaced by $w_{2,3}$ and the NCC between $w_{2,3}$ and all other clusters (w_1 and w_4 in this case) has to be calculated.

$$\begin{aligned} NCC(w_{2,3}, w_1) &= \frac{1}{1+1} \times NCC(w_2, w_1) \\ &+ \frac{1}{1+1} \times NCC(w_3, w_1) \\ &= \frac{1}{2} \times 0.0040 + \frac{1}{2} \times 0.0069 \\ &= 0.0055 \end{aligned} \tag{12}$$

$$\begin{aligned} NCC(w_{2,3}, w_4) &= \frac{1}{1+1} \times NCC(w_2, w_4) \\ &+ \frac{1}{1+1} \times NCC(w_3, w_4) \\ &= \frac{1}{2} \times 0.0010 + \frac{1}{2} \times 0.008 \\ &= 0.0009 \end{aligned} \tag{13}$$

Table 2: Updated example matrix with NCC values

	w_1	$w_{2,3}$	w_4
w_1	1	0.0055	0.0003
$w_{2,3}$	0.0055	1	0.0009
w_4	0.0003	0.0009	1

The first iteration step is now finished. Table 2 shows the updated matrix, now 3x3 instead of 4x4. During the second iteration step it is found that the highest correlation is between w_1 and $w_{2,3}$. Once again the correlation is above the predefined threshold meaning that w_1 and $w_{2,3}$ are merged. The NCC of $w_{1,2,3}$ with all other clusters (w_4 in this case) needs to be updated.

$$\begin{aligned} NCC(w_{1,2,3}, w_4) &= \frac{2}{2+1} \times NCC(w_{2,3}, w_4) \\ &+ \frac{1}{2+1} \times NCC(w_1, w_4) \\ &= \frac{2}{3} \times 0.0009 + \frac{1}{3} \times 0.0003 \\ &= 0.0007 \end{aligned} \tag{14}$$

Table 3: Final example matrix with NCC values

	$w_{1,2,3}$	w_4
$w_{1,2,3}$	1	0.0007
w_4	0.0007	1

During the third iteration step, the highest correlation found is 0.0007. This is below the predefined threshold and therefore the clustering process stops. The final result is that the NCC matrix is 2×2 (Table 3), meaning that two different sources are found. One source created images 1, 2 and 3, while another source created image 4.

Table 4: Overview of the cameras used in the experiments

Camera Database	Camera brand and model
NFI camera database	Blackberry Curve 9360
	Blackberry Curve 9300
	Canon Powershot A630
	Canon IXUS 220 HS
	Canon Powershot SX210
	Casio Exilim EX-FC100
	Nikon D3200
	Nikon D800
	Nikon Coolpix L27
	Nokia C3-00
	Panasonic Lumix FP7
	Panasonic Lumix FZ45
	Panasonic Lumix SZ5
	Samsung ST30
	Samsung NX1000
	Samsung Digimax L70
	Samsung Digimax S500
	Samsung Galaxy S3 Mini
	Sony Cyber-Shot DSC S500
	Sony Cyber-Shot DSC S930
	Sony Cyber-Shot DSC S800
	Sigma DP1S
Dresden Image Database	Olympus mju 1050 SW
	Pentax Optio A40
	Ricoh Caplio GX100

3. Experimental setup

To test the performance of the modified methods and for the novel clustering process, four experiments have been performed. These four experiments are:

- Testing of the validity of the grayscale conversion (step 1).
- Testing of common source identification before and after clustering (step 8).

- Testing of the characteristics of clustering
- Testing of the computation time

For each of these four tests a database with images has been created. These images were made with cameras from the NFI camera database and Dresden Image Database [7]. In Table 4 an overview of the camera brands and models in each database is given. All images in the NFI camera database were taken using the automatic settings of the camera, without digital or optical zoom. For each model multiple cameras were available. Cameras of the same brand and model often correlate higher with each other, than with cameras of a different brand and model, due to similar (post-)processing [8]. Using multiple cameras of the same brand and model in the experiments tells us how well the proposed methods perform, even under more difficult circumstances. Natural images were made from various objects and landscapes. When the PRNU pattern of an image of a camera is compared with the PRNU pattern of another image of the same camera it is called a match. If the PRNU pattern of an image is compared with the PRNU pattern of an image from a different camera, it is called a mismatch.

3.1. Grayscale conversion experiments

For the experiments on grayscale versus RGB performance DB_0 was composed. To detect a difference between RGB and grayscale the image quality had to be degraded. This was done by cropping the image and performing a JPEG compression with IrfanView. Each image was cropped in the center with a resolution of 512×512 and for each image a version with JPEG compression quality settings 100, 50 and 25 was created. A lower quality setting means more compression. The experiments made use of Basic SPN, which is the average of the PRNU patterns, to form the SPNs using flat-field images shot from a white wall. Although there are other methods available, such as Maximum Likelihood Estimator (MLE) [3] and Phase SPN [10], the simpler Basic SPN is sufficient for the experiment. 50 flat-field images per camera were used to create the SPNs, 100 natural images per camera were made for the experiment. Every image was duplicated and the duplication was converted to grayscale. The equal error rate (EER) is used to analyze the performance of grayscale images compared to RGB images. EER is where the rate of false positives is equal to the rate of false negatives. The method with a lower EER is generally more accurate.

3.2. Common source identification before and after clustering

The effects of decreasing the computation time before and after clustering is tested using two databases. For the initial testing a database DB_1 with 2700 natural images was composed. There are 100 images per camera, 9 different camera models with 3 cameras per model. All images were cropped to the same size of 3343×2509 pixels. DB_2 is a database with 7100 natural images of 71 different cameras. For most but not all camera models there were multiple cameras. To form a database of this size, all images had to be cropped to the size of the smallest image, 1600×1200 pixels. DB_1 and DB_2 are composed such that they are representative for real case data.

The performance is calculated by determining the true positive rate (TPR) at a false positive rate (FPR) of 0. A true positive is when two images have the same source camera, and the associated NCC score is above a predefined threshold. A false positive is when two images have a different source camera, but the associated NCC score is above the threshold. Likewise, a true negative is when two images have a different source camera and the associated NCC score is below the threshold; and a false negative is when two images have the same source camera, but the associated NCC score is below the threshold.

To calculate the TPR at 0 FPR before clustering, the threshold where there are no false positives is determined. Then, the number of true positives is divided by the total number of positives. To calculate the TPR at 0 FPR after clustering, the threshold where none of the clusters contain a false positive is determined. Then the number of images within all clusters is divided by the total number of images in a database.

3.3. Characteristics of clustering

DB_1 is also used to illustrate several characteristics of clustering, such as the effects that image size has on the results.

3.4. Computation time

To evaluate the computation time, database DB_4 is created containing 100 images with a size of 4320×3240 pixels.

All experiments have been performed on a 2.3 GHz quad core computer.

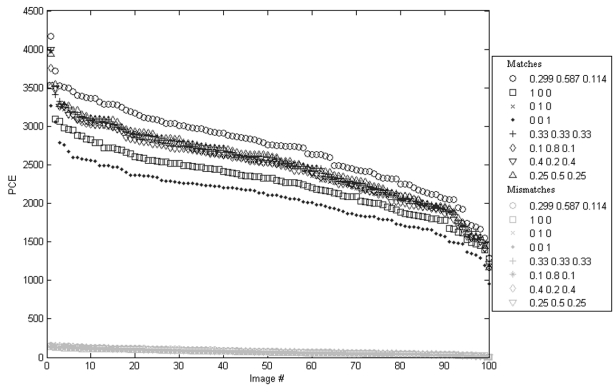

Figure 1: PCE match and mismatch values for several grayscale ratios. The upper curve shows the best results using the ratio [0.299 0.587 0.114]

4. Experiments, results and discussion

4.1. Grayscale conversion experiments

By converting RGB images to grayscale before analyzing the images, the amount of data is reduced from three channels to one. The following experiments are executed to determine the effects of grayscale conversion on the accuracy of the results. Grayscale conversion can be done with several ratios for each color channel. A small preliminary test was done to see if the ratios have a noticeable influence on the result. PRNU patterns of 100 natural images were compared to SPNs of cameras of the same brand and model. PCE was used to evaluate the ratios. The results for the different ratios can be found in Fig. 1. There was one specific combination that gave the best results: 0.299× red channel intensity, 0.587× green channel intensity and 0.114× blue channel intensity. The PCE match values of this ratio were higher than any of the other ratios, while the mismatch values were equal. This ratio is used in all following experiments.

The performance of RGB and grayscale images is tested using DB_0. PRNU patterns of the natural images were compared to SPNs of cameras of the same brand and model. Results for each image setting can be found in Table 5. The table shows us that there are no major differences between the EER of RGB images and grayscale images. Therefore it may be concluded that grayscale conversion can be used for a large reduction of computation time without loss of accuracy.

Table 5: Equal Error Rates (EER) for cameras at different JPEG compressions using RGB and grayscale images. Lowest EER per image setting in bold

Camera model	**JPEG compression**	**EER RGB**	**EER Grayscale**
Blackberry Curve 9360	100	**0**	**0**
	50	**0**	**0**
	25	0.03	**0.02**
Canon IXUS 220 HS	100	**0.02**	**0.02**
	50	**0.05**	**0.05**
	25	**0.15**	0.16
Canon PS A630	100	**0**	0.01
	50	**0.04**	**0.04**
	25	0.17	**0.16**
Canon PS SX210	100	**0.01**	0.02
	50	**0.02**	**0.02**
	25	0.22	0.22
Nikon Coolpix L27	100	**0.01**	**0.01**
	50	0.06	**0.05**
	25	0.13	**0.12**
Nikon D3200	100	**0**	**0**
	50	**0.13**	0.14
	25	**0.35**	0.38
Casio Exilim EX-FC100	100	**0**	**0**
	50	**0**	**0**
	25	0.06	**0.05**
Nokia C3-00	100	**0.03**	**0.03**
	50	**0.03**	**0.03**
	25	**0.05**	**0.05**
Panasonic LUMIX FZ45	100	0.01	**0.008**
	50	**0.01**	0.02
	25	**0.04**	0.05
Panasonic LUMIX SZ5	100	0	0
	50	**0.02**	**0.02**
	25	**0.09**	**0.09**
Panasonic LUMIX FP7	100	**0.02**	0.04
	50	**0.11**	0.12
	25	**0.23**	**0.23**
Samsung Digimax L70	100	**0.02**	0.04
	50	**0.11**	0.12
	25	**0.23**	**0.23**
Samsung Galaxy S3 mini	100	**0**	**0**
	50	**0**	**0**
	25	**0.08**	0.09
Samsung ST30	100	**0**	**0**
	50	**0.01**	**0.01**
	25	0.05	**0.03**

4.2. Results before and after clustering

The performance before and after clustering is tested using DB_1. All images in this database are compared to each other, resulting in a 2700×2700 matrix containing approximately 3.6 million comparisons. Before clustering, the TPR of DB_1 at a 0 experimental FPR is 0.6508. After clustering a TPR at a 0 FPR of 0.9463 is found. This is a significantly better result than before clustering (45% increase in TPR).

For DB_2 the TPR before and after clustering is calculated as well. Before clustering, the TPR at 0 FPR is 0.4496. Less than half of the images can be correctly identified. After clustering, the TPR at 0 FPR is 0.8744. Clustering improves the TPR by 94%. Clustering also seems more stable, than when clustering is not applied. There is only an 8% decrease in TPR when comparing the results of clustering DB_2 and DB_1, while there is a 45% decrease between DB_2 and DB_1 without clustering.

During the experiment with DB_2 we noticed that one camera, the Canon IXUS 220 HS, gave somewhat anomalous results. In order to identify each camera from this model, the threshold had to be increased twofold, compared to if this camera was not in the database. If the threshold was too low, all images from the three Canon IXUS 220 HS cameras in the database were classified as coming from a single camera. Leaving this camera out of the database and lowering the threshold would result in a TPR of 0.9554 at a 0 FPR. The Canon IXUS most likely has some model specific post-processing that causes a higher correlation between cameras of this model. While it was easy to separate these models from each other with a higher threshold, this caused more false negatives for all the other cameras. The Canon IXUS 220 HS was also present in DB_1. It is likely that this camera may have had a large effect on the threshold of this database as well. We recalculated the TPR for DB_1 leaving out the Canon IXUS 220 HS: 0.9833 at 0 FPR. To show that the Canon IXUS 220 HS really is an outlier in this database, the different thresholds per camera model to achieve no false positives for the respective camera in DB_1 and DB_2 are shown in Table 6. False positives in this case are when images of the camera model in question are grouped with images from another camera. False negatives are when images are not in any cluster at all. As the table shows, Canon IXUS 220 HS is clearly an outlier.

4.3. Characteristics of clustering

The following experiments focus on the characteristics of clustering. First, the effects of the size of the images in DB_1 is investigated. As shown in

Table 6: Lowest threshold for each camera at 0 FPR

Camera model	**Threshold** DB_1
Canon IXUS 220 HS	0.0017
Ricoh Caplio GX100	0.0007
Panasonic Lumix FP7	0.0006
Sony Cyber-Shot DSC-S930	0.0006
Pentax Optio A40	0.0006
Samsung ST30	0.0006
Nikon Coolpix L27	0.0006
Canon PowerShot SX210	0.0005
Samsung NX1000	0.0004
Nikon D3200	0.0003
Panasonic Lumix FZ45	0.0003
Panasonic Lumix SZ5	0.0003
Camera model	**Threshold** DB_2
Canon IXUS 220 HS	0.0033
Sony Cyber-Shot DSC-S930	0.0016
Sigma D1PS	0.0016
Samsung NX1000	0.0015
Samsung Digimax S500	0.0015
Samsung Digimax L70	0.0015
Nikon Coolpix L27	0.0015
Panasonic Lumix FP7	0.0014
Panasonic Lumix SZ5	0.0012
Blackberry Curve 9360	0.0009
Nikon D3200	0.0009
Nikon D800	0.0008
Canon PowerShot SX210	0.0008
Sony Cybershot DSC S800	0.0007
Canon IXUS 220 HS	0.0006
Blackberry Curve 9300	0.0005
Panasonic Lumix FZ45	0.0004
Samsung ST30	0.0003
Samsung Galaxy S3 mini	0.0002
Canon Powershot A430	0.0002

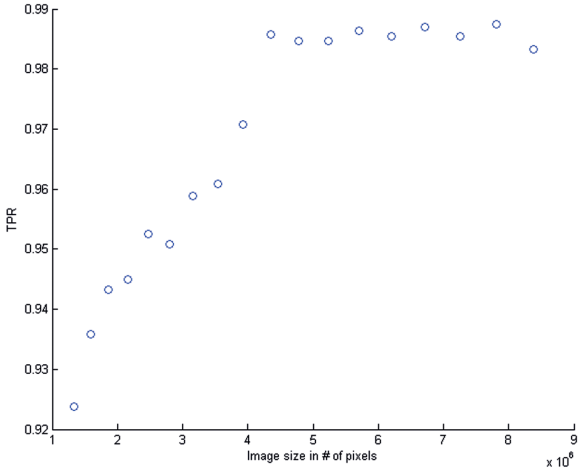

Figure 2: The True Positive Rate as a function of the image size

the previous section, the TPR of DB_1 was higher than the TPR of DB_2. The two major differences were the number of cameras and the image size, and we hypothesize that the image size has a large influence on the results. Table 6 also shows that the threshold needed for DB_1 is lower than for the same camera in DB_2. A characteristic of NCC is that the threshold varies for different image sizes for a fixed FPR, which was one of the reasons to introduce PCE ([1]). The question is whether this is also the case for clustering. For this experiment DB_1 is used; the Canon IXUS will be left out, because of the outlying results in the previous experiments. Image sizes will be tested from 1.3 million pixels to 8.4 million pixels, using cropped images of DB_1. Note that the digest size always stays the same.

Fig. 3 shows that the threshold to reach an FPR of 0 decreases with larger images, and eventually stabilizes around 8 million pixels. Fig. 2 shows that the TPR stabilizes after 4 million pixels. A higher TPR is expected for larger images, because of the following. PRNU is randomly distributed; with more pixels available, the chance of finding pixels with a stronger PRNU signal increases. In small images more pixels are selected with a less optimal PRNU signal, decreasing the matching correlations. While clustering does not completely prevent the problem that NCC thresholds vary with image size, it is robust to some point.

Subsequently, the number of pixels that are actually used in the correla-

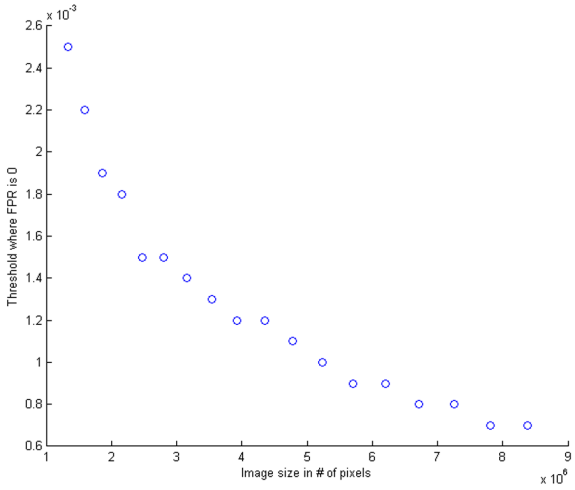

Figure 3: The threshold where the FPR is 0 as a function of the image size

tion are reviewed. As noted in Section 2, approximately 1.2 million pixels are used for the digest. However, not all pixels have the same location. It is possible that matches have more pixels at overlapping locations than mismatches. This seems reasonable because strong PRNU effects would have the same location when images originate from the same source. If the difference between the number of overlapping pixels of matches and mismatches is large enough, it could be used as an extra method for identification. Going through the number of overlapping pixels for all comparisons from DB_1 gave the following results. For matches the mean is 195817 pixels with a standard deviation of 10007. For mismatches the mean is 185613 pixels with a standard deviation of 10409. As can be expected, matches generally have a slightly higher number of pixels than mismatches. However, the difference is small compared to the standard deviation. It is therefore not helpful as additional identification method. The original image size was 3343×2509, which means that that approximately 2% of the number of pixels of the original image is compared. When smaller images are used, the number of overlapping pixels will go up, as less pixels are available to select. The opposite happens with larger images. It is possible that with very large images there are no overlapping pixels at all. A possible way to overcome this is by using only a part of such a large image.

4.4. Computation time

Step 7, the comparison of all images within a database, is the most time consuming part of common source identification. In this experiment the computation time of step 7 using digested and quantized PRNU patterns is compared to the computation time of step 7 using full sized PRNU patterns. All images in DB_4 were compared to each other. This resulted in 4900 comparisons. On the 2.3 Ghz CPU used in the experiments the combination of methods to decrease computation time reaches approximately 500 comparisons per second, completing the comparison of DB_4 in approximately 10 seconds. Using the full sized PRNU patterns on the other hand, took about an hour and a half, at a speed of 1 comparison per second. It should be noted that the computation time of full sized PRNU patters depends on the image size, whereas the digest and quantized PRNU patterns do not. Clustering of DB_4, step 8, takes less than 1 second, showing that it is very efficient. A GPU version of the methods was designed without much optimization and reached a speed of 3000 comparisons per second using an Nvidia Tesla K20c GPU. The Tesla is an expensive GPU, but nowadays cheaper alternatives designed for gaming form an adequate alternative. In our experiments it was assumed that all images in a database were unmodified (no cropping, scaling, lens distortion and/or rotating), other than our controlled modifications. In an uncontrolled (case) database these modifications would be unknown. To control this for each single image would take a large amount of computation time, but with clustering this would be reduced to the number of clusters, rather than the number of images. While we did no testing on these modifications, several methods have been described in the literature [1] [5]. These could be applied after clustering to decrease the false negative rate (FNR).

5. Real case

We had the opportunity to test real case data. A large fraud case in the Netherlands yielded image data from approximately 150 seized devices. Several documents had been stolen and photographed and then shared among people. Our analysis would not be used for the official court case, as the case was already closed. However, the police still wondered if the photographs of the stolen documents shared a common source with any of the images from the seized devices. Data from all the devices was scanned for images with the same size as the images from the stolen documents. Due to time limitations it was assumed that images in this database would not be rescaled, cropped or

rotated. This yielded a total of 9812 images that met the requirement. Next, the PRNU patterns of all images, including the photographs of the stolen documents, were extracted according to step 1–6. Then all PRNU patterns were compared to each other to acquire the matrix with NCC values (Step 7). Finally, this matrix was used to cluster the data (Step 8). In total 400 clusters were found, of which 42 clusters contained 10 or more images. Here we will focus only on those 42 clusters. These clusters contained 8760 images, with the largest cluster containing 1695 images. Combining information from the police and image content it was found that with a high degree of certainty the clusters were correct, insofar that we were unable to find false positives. The TPR was then calculated to be at least as high as 0.8928, but possibly higher, since we did not look at the remaining clusters with less than 10 images. It is still possible that there were also false positives in these remaining clusters.

All 42 clusters have been inspected by eye. It was observed that one cluster contained only images of shoes; another cluster contained images with text in a certain font and color added to it; a third cluster contained only images of a specific dance club and yet another cluster had pictures likely from a holiday trip to an Asian country. Because the relation of image content within the clusters seems logical, the observations confirm the correctness of the clustering process. The most interesting result was that the photographs of the stolen documents were all grouped in the same cluster, together with images depicting several individuals and locations. Information like this could help the police in finding a suspect.

6. Conclusion

Common source identification of images in large databases requires a lot of computations. In this paper we demonstrated that by applying several time-saving methods — grayscale conversion, digestion, quantization and the modified NCC formula — one can analyze a large database in forensically relevant time, without resorting to large and expensive computer clusters. Some of the methods had to be modified before they could be applied to single image common source identification. These methods greatly reduce the computation time by a factor of roughly 500, at the cost of accuracy. However, by applying our proposed method of clustering, accuracy is vastly increased at very low computation costs. Analysis of a fraud case showed that the proposed methods work not only in a controlled setting, but also on real case data. Besides the considerations of fast common source identification,

the proposed clustering method can also be applied to cases where no time-saving methods is required.

More research can be done to optimize parameters, such as the number of pixels used for the digest and the bits used in quantization. Additionally, it is possible that images in a database have been cropped, rescaled and/or rotated. Testing this after clustering might reduce the false negative rate even more. In this paper we focussed on the optimization of the time needed per comparison. Further optimization could be achieved by implementation of this software on a GPU.

References

[1] M. Goljan and J. Fridrich, "Camera identification from cropped and scaled images," Proc. SPIE, Electronic Imaging, Forensics, Security, Steganography, and Watermarking of Multimedia Contents X, vol. 6819, pp. OE-1-OE-13, Jan. 2008.

[2] J. Lukáš, J. Fridrich and M. Goljan, "Digital camera identification from sensor pattern noise," *IEEE Trans. Inform. Forensics Security*, vol. 1, no. 2, pp. 205-214, Jun. 2006.

[3] M. Chen, J. Fridrich and J. Lukáš, "Determining image origin and integrity using sensor noise," *IEEE Trans. Inform. Forensics Security*, vol. 3, pp. 74-90, Mar. 2008.

[4] M. Goljan and T. Filler, "Managing a large database of camera fingerprints," *Proc. SPIE, Electronic Imaging, Media Forensics and Security XII*, vol. 7541, pp. 01-12, Jan. 2010.

[5] M. Goljan and J. Fridrich, "Estimation of lens distortion correction from single images," *Proc. SPIE, Electronic Imaging, Media Watermarking, Security, and Forensics 2014*, vol. 9028, Feb. 2014.

[6] Y. Hu, B. Yu and C. Jian, "Source camera identification using large components of sensor pattern noise," *2nd International Conference on Computer Science and its Applications*, pp. 1-5, Dec. 2009.

[7] T. Gloe and R. Böhme, "The 'Dresden Image Database' for benchmarking digital image forensics," *Proc. 25th Symp. Applied Computing*, vol. 2, pp. 1585-1591

[8] T. Gloe, S. Pfennig and M. Kirchner, "Unexpected artefacts in PRNU-based camera identification: a 'Dresden image database' case-study," *Proceedings of the 14th ACM Multimedia and Security Workshop*, pp. 109-114, Sep. 2012.

[9] S. Bayram, H. T. Sencar, N. Memon, "Efficient sensor fingerprint matching through fingerprint binarization," *IEEE Trans. Inform. Forensics Security*, Vol. 7, no. 4, pp. 1404-1413, Apr. 2012.

[10] 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*, vol. 7, pp. 393-402, Apr. 2012.

[11] A. Cooper "Improved photo response non-uniformity (PRNU) based source camera identification," *Forensic Science International* vol. 226, pp. 132-41, Mar. 2012.

[12] W. van Houten and Z. Geradts, "Using Anisotropic Diffusion for Efficient Extraction of Sensor Noise in Camera Identification," *Journal of Forensic Sciences*, vol. 57, pp. 521-527, Feb. 2012.

[13] F. Gisolf, A. Malgoezar, T. Baar and Z. Geradts, "Improving source camera identification using a simplified total variation based noise removal algorithm," *Digital Investigation*, vol. 10, pp. 207-214, Oct. 2013.