Using The Noise Density Down Projection To Expose Splicing In JPEG Images

Thibault Julliand, Vincent Nozick, Isao Echizen, Hugues Talbot

Using The Noise Density Down Projection To Expose Splicing In JPEG Images

Thibault Julliand, Vincent Nozick, Isao Echizen and Hugues Talbot ǂ

September 14, 2017

Abstract

In digital image forensics, the detection of exogenous elements in an image is a difficult open problem, and several ways to solve it have already been proposed. In this paper, we use differences in the intrinsic noise in digital images to highlight this type of forgery. Our method analyses an histogram based on the noise density function at the local level in order to reveal suspicious areas, by using a new tool exploiting the uniform property of the Gaussian noise contained in JPEG images. Although the use of noise discrepancies to detect splicing has already been done multiple times, most existing methods tend to perform poorly on the current generation of high quality images, with high resolution and low noise. The improvements of our method over other state of the art approaches is demonstrated on a large set of randomly spliced natural images.

Index terms— Image forgery, noise, digital image forensics.

1 Introduction

Over the last two decades, the number of digital images have greatly increased, due to the popularization of digital cameras and the introduction of smartphones. As such, they form the vast majority of the pictures one can encounter nowadays. This increase has been accompanied by a rise in the number of image-altering tools such as Photoshop or GIMP, and consequently in the quantity of altered or outright falsified images. Digital image forensics was then developed with the goal to determine the authenticity of digital images [9, 31]. Several families of falsification exist, the two more common being the copy-rotate-move (or endogenous insertion) [7], which consists in duplicating areas of an image to

*T. Julliand and H. Talbot are with the Université Paris-Est, LIGM (UMR 8049), CNRS, ENPC, ESIEE Paris, UPEM, F-93162, Noisy-le-Grand, France email: {thibault.julliand, hugues.talbot}@esiee.fr.

†V. Nozick is with the JFLI (UMI 3527), CNRS, NII, Tokyo, Japan email: vincent.nozick@u-pem.fr.

‡I. Echizen is with the National Institute of Informatics, Tokyo, Japan email: iechizen@nii.ac.jp.

add or conceal elements, and the splicing (or exogenous insertion) [27], in which part of an image is inserted in a second, different image.
In this paper, we present a new method to detect splicing in digital images. While a considerable number of algorithms exploiting various elements of the information conveyed in the image already exists, only a small fraction uses the noise in the image. Our method exploits the statistical properties of the noise density function and is based on the assumption that the inherent noise in images is different enough from one image to the other that their density functions differ. We also presume that at least one of the images is otherwise unaltered.
This paper first presents a brief overview of the various approaches used to detect alterations in digital images. We then address the subject of noise density histogram, and its transformation to a noise contribution histogram. The next part introduces the concept of noise contribution down-projection, which is the core of our method. The following section shows how to use this new tool to detect splicing, accompanied by several pre- and post-processing steps. The tests are performed on a large database of real-world images, randomly spliced into each other. Finally, we expose our results and compare them to other state-of-the-art methods.

2 Digital Image Splicing Detection

2.1 General splicing detection

Many ways of detecting image splicing have already been proposed, which can be classified into a few broad categories. The first one is format-dependent, mainly exploiting the particularities of the JPEG format: the JPEG ghost detection developed by Farid [10] is based on detecting traces of the original compression quality, while the JPEG quantization analysis by Popescu et al. [29], He et al. [15], and Lin et al [23] tries to detect different rates of compression throughout the image. Double JPEG compression, such as proposed by Huang et al. [17] doesn't specifically target splicing, but can still be used in this optic. A second category is based on the use of the Color Filter Array (CFA), such as presented by Popescu and Farid [30], Gallagher and Chen [12], or Ferrara et al. [11]. These three methods use irregularities in the expected CFA interpolation pattern to highlight suspicious areas in the image. A third approach is based on machine learning and statistical analysis, such as presented by Bayram et al. [1], Fu et al. [4], or Han et al.[14]. The fourth category exploits geometrical and lighting informations, as shown by Johnson and Farid [18, 19] or Chennamma and Rangarajan [5], sometimes associating it with machine learning by Wang et al. [33], or using context clues in the image like Brogan et al. [2]. Finally, a set of methods uses noise image analysis.

2.2 Noise-based approaches

Noise is inherent to the image acquisition process, and is subjected to many alterations and transformations from the moment the light is captured on the sensors up to the point where a JPEG file is recorded on storage media [21]. However, it can be safely assumed that the overall noise will be consistent inside of a single image, and is dependent on the camera model, the lighting conditions, shutter speed, etc. It can be induced that the noise in two different images will tend to differ somewhat, and as such a variation of noise parameters in a specific zone of a single image is a strong indicator of alteration or splicing. Several approaches have started to exploit this. Lukáš et al. [24] and Chierchia et al. [6] use the Photo-Response Non Uniformity (PRNU), a type of fingerprint noise which is unique to each camera. However, it requires the possession of either several pictures taken by the camera, or the camera itself for identification purposes. Moreover, this type of noise is increasingly corrected for by the camera manufacturer. Mahdian and Saic [25] use a local analysis of wavelet decomposition to estimate the noise variance and highlight inconsistencies. He et al. [16] use a L1-norm error function followed by a clustering to detect splicings. Zeng et al. [34] perform a PCA-based block noise estimation to separate the image in two regions according to their level of noise. Julliand et al. [20] use block-based differentiation on the noise probability density function. However, their approach only works on RAW images. In this paper, we present an adaptation of the latter method to JPEG images.

2.3 Denoising

The first step of our method is to denoise the image. Authors in [20], following a similar approach, use the denoising algorithm proposed by Dabov et al. [8]. Although extremely effective on RAW images, this denoiser is designed for 16-bit images and relies on a Poisson-Gaussian model for the noise. As our method is to be applied on 8-bit JPEG images, and since in other works [21] we have shown that the noise of those images can be considered as a zero-mean Additive White Gaussian Noise (AWGN), we consider two state-of-the art denoisers: the non-local means method implemented by Buades et al. [3], and the BM3D implementation by Lebrun [22]. Although both methods give comparatively similar results, the approach by Buades et al. is faster, so we use that one in our experiments.

3 Noise Density Function Models

As explained in Sec. 2.2, the noise in a single image can be assumed to follow a single noise probability density function. As such, it is expected that noise in a spliced image can be divided in two parts: the noise from the original image, and the noise from the spliced element, as shown in Fig. 1. Although the overall noise probability density function should not be altered much if the size of the

spliced element is small compared to that of the image, a local analysis of this function may allow a differentiation between the two parts.

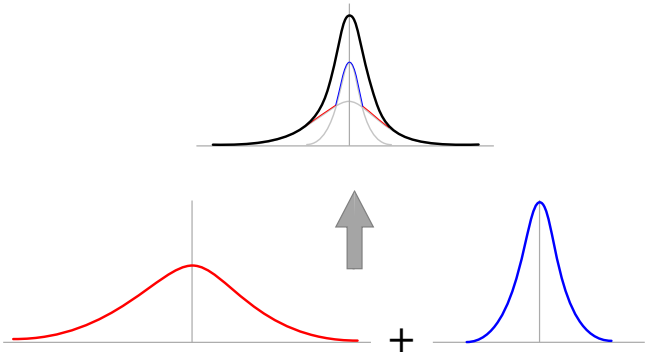


Figure 1: The noise density function in a spliced image remains Gaussian-like, and is the sum of two Gaussian functions.

3.1 Noise density histogram

The noise density histogram of a discrete signal is a representation of the probability density function of the noise contained in the signal. In our method, we build the noise density histogram based on the original image I_O and its denoised version I_D. As such, each pixel location p of the image is assigned a pair of values, $v_n(p)$ and $v_d(p)$, corresponding to its original noised value and denoised value respectively. Then, the value of the noise density histogram H at any point (i, j) is the number of pixels respecting the condition $(v_n(p), v_d(p)) = (i, j)$ or, more formally:

$$H(i,j) = \operatorname{card} P_{i,j},\ P_{i,j} = \{p \mid v_n(p) = i \text{ and } v_d(p) = j\}$$

However, for the histogram to be representative of a probability density function, each column j (denoised value) needs to be be normalized with a normalizing constant C_j, where $C_j = \operatorname{card} p \mid v_d(p) = j$.

If we assume an image with an equipartition of values, i.e. each possible intensity is represented by an equal number of pixels, and add a perfect AWGN of standard deviation σ, then the noise density histogram can be modeled as the basic Gaussian probability density shown in Fig.2. In this situation, we expect the value of the normalized histogram to be roughly equal to the equivalent Gaussian probability density:

$$H(i,j) \simeq \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(i-j)^2}{2\sigma^2}}$$

Fig. 3 depicts the noise density histogram computed on a real JPEG image, from Algo. 1. It can be seen that the comportment of the extremities of the histogram differs from the rest, which is due to the saturation of the noise. In our experiments, we ignore those two extremes.

Algorithm 1: Global density histogram generation

Input: original image: I_O
denoised image: I_D
Output: global density histogram: H

```
// init the histogram
H.fill(0)
// fill the histogram
for p ∈ I_O do
    Increment(H(I_O(p), I_D(p)))
// normalize the histogram
foreach column c of H do
    normalize(c) such ∑_i H(c, i) = 1
```

3.2 Noise contribution histogram

A noise contribution histogram represents the contribution of a subimage to the noise density histogram of the whole image. A value at any point in the histogram can be interpreted as the number of pixels fulfilling the values condition of this point contained in the subimage, divided by the number of pixels fulfilling this same condition contained in the full image. As such, it is an element-wise division of the noise density histogram of the subimage by the global noise density histogram, both non-normalized. More formally:

$$C_{sub}(v_d, v_n) = \frac{H_{sub}(v_d, v_n)}{H_{im}(v_d, v_n)}, \quad \forall (v_d, v_n)$$

The contribution histogram values are upper-bounded by 1, indicating that all the pixels couple values (v_d, v_n) are contained in the subimage. The sum of the contribution histograms of non-overlapping subimages covering the entire image will be an histogram with a shape resembling that of the global noise density histogram (i.e. a gaussian noise density function), with the values all equal to 1. The advantage of the contribution histograms is that they allow to distinguish noises variations, even when they are small. Assuming the noise is always centered, a small change in standard deviation (<1) will not alter the width of the distribution, but will change its overall shape, allowing differentiation as shown in Fig. 4. An example of the noise contribution histogram of a subimage is shown in Fig. 5.

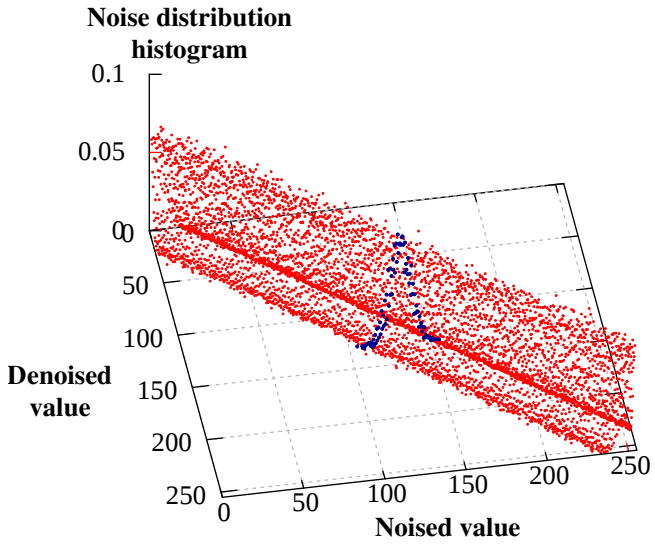


Figure 2: Ideal Gaussian noise density histogram. The dark line is a cross-section along a single denoised value, and represents a 1D Gaussian distribution.

4 Noise Contribution Down-Projection

One of the main issues with the noise contribution histograms is that their intensity distribution corresponds to the one of the subimage, e.g. a subimage with a majority of dark pixels will have an histogram with values concentrated in the lower rows and columns of the histogram. Consequently, it is hard to compare two subimages with highly different intensity ranges, since their contribution histograms will have little or no intersection. In [20], there is no real way around that, since the noise follows a Poisson-Gaussian law and as such is inherently spatially variant. However, when dealing with pure AWGN, we know that the noise distribution is independent of the base intensity. As such, if we first deskew the histogram as shown in Fig.6, so that it is centered around a single line instead of around the diagonal identity, then histogram rows may be interchanged freely.

The concept of noise contribution down-projection derives from this independance. If rows can be interchanged, then it is also possible to exchange single histogram bins between rows, if they have the same signed difference between their noised and denoised values, e.g. exchanging the point of coordinates (51, 2) with the one in (57, 2).

The construction of a noise contribution down-projection from a noise contribution histogram is as follows: first, the histogram is deskewed so that the identity axis is along a single value O, by default the 0 axis, using this equation:

$$\forall t \in H \begin{cases} H_R(t).x = H(t).x \\ H_R(t).y = H(t).x - H(t).y \end{cases}$$

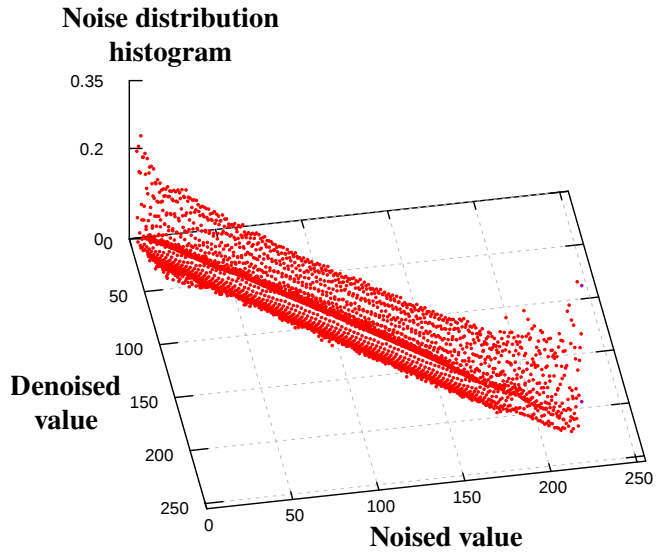


Figure 3: Gaussian noise density histogram obtained on a natural image with our algorithm.

where H_R is the skewed histogram and $.x$ and $.y$ respectively represent the first and second coordinates of a point in the histogram. Then, for each column, we remove all bins with a content of 0 and move down the upper elements of one histogram cell. The length of a column is then only the number of non-zero bins originally in it, as can be seen in Fig. 7 and applied to real data in Fig. 8.

The main justification of this data manipulation is that all of the separate contribution down-projections will have roughly the same appearance: the first row will be the largest, and they will get progressively smaller as we progress through. As a consequence, we will always compare the largest possible rows of a contribution down-projection to its average contribution equivalent, as depicted in Fig. 9.

5 Application to Splicing Detection

5.1 Tiling

In order to analyze the noise in an image I_O, we first denoise it, obtaining the denoised image I_D. As suggested in Section 4, these images are then divided into non-overlapping square subimages of identical sizes, referred to hereafter as tiles. The size of the tiles is an important consideration: we need enough pixels in a tile so that their contribution down-projections are statistically significant, but we also know that smaller tiles will yield a better resolution to pinpoint the location of the falsification. According to our tests, a size of 64×64 pixels provides enough statistical data, while giving a reasonable accuracy on medium

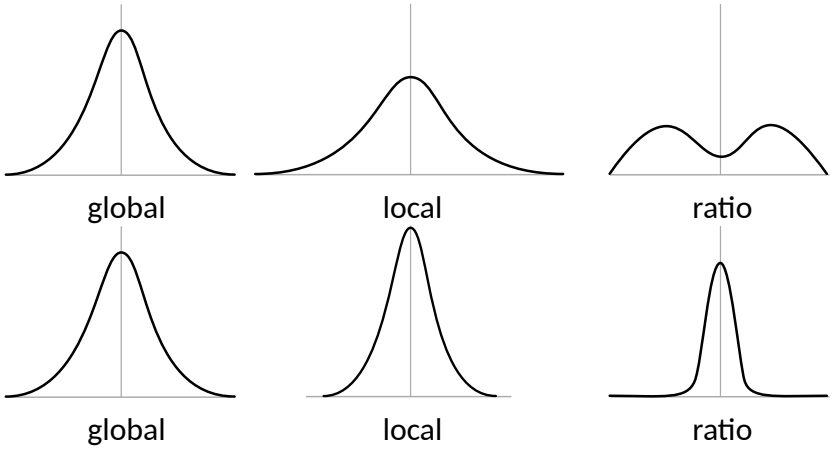


Figure 4: Ideal shapes of various contribution histograms. Top: spliced part with a higher standard deviation noise. Bottom: spliced part with a lower standard deviation noise.

to wide sized images.

5.2 Average Contribution down-projection

We then build the overall noise histogram, and the noise down-projection of each tile, using the method presented in Sec. 3. These down-projections are what will allow us to classify our subimages. In addition, we also construct an overall average contribution histogram (and its corresponding down-projection), by averaging all the non-zero contributions for each point of the histogram:

$$DP_{av}(v_d, v_n) = \frac{\sum_i^{Ns} DP_i(v_d, v_n)}{card\big(\{DP_i) \mid DP_i(v_d, v_n) \neq 0\}\big)}$$

where N_S is the number of subimages. Using all the contribution for this histogram (including the zero ones) would result in an flat histogram where each point has a value of $1/N_S$.

5.3 Classification of Sub-Images

For each tile, we compute the location of the maximal differences its contribution down-projection has compared to the average. Here, we consider only the lowest row of each down-projected histogram. It is expected that the tiles from the original image will have their maximal differences spread equitably around the central axis, while the tiles from the spliced element will have maximal differences either close to the axis or as far away from it as possible, depending on whether it has more or less noise than the original image, as seen on Fig.9. The

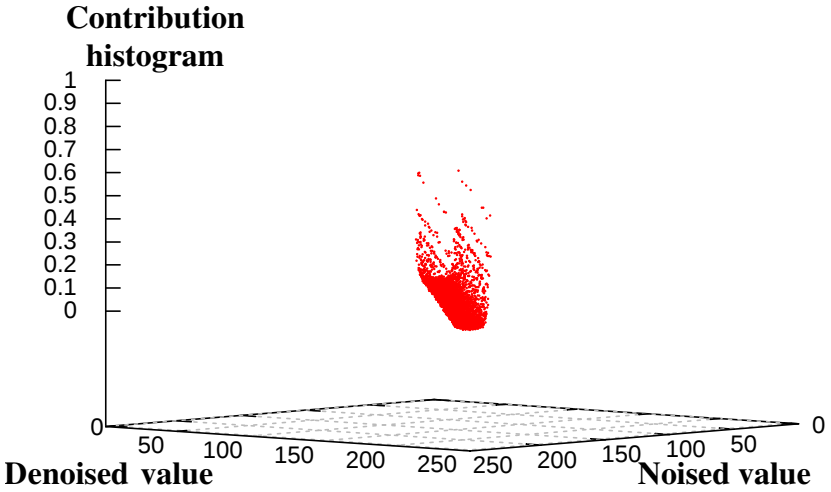


Figure 5: The noise contribution histogram of a subimage. The higher contributions on higher noises is normal, and is useful for identifying the spliced area.

score of a tile is then calculated based on those distances, and is obtained by taking the average distance between its highest computed differences and the axis.

5.4 Results Visualization Enhancements

The result given by the first process of classification, although readily usable, can be improved with several post-processing methods. The first thing that can be done is to apply a scaling on the results, while removing the current extreme values. This allows the classification to extend on a wider range, while avoiding to be misled by potential aberrant, outlier results. This normalized image is then submitted to a double threshold, whose values were found experimentally to give the best average results on our database, in order to improve its readability by a human observer, as shown in Fig. 10. This results in an easier way to highlight possible discrepancies on the input image by comparing it to the threshold image. Finally, the whole method (pre-processing, main algorithm, scaling, and thresholding) is applied on each color channel, and on the luminance channel for additional information.

5.5 Formal Algorithm

The overall pipeline is outlined in algorithm 3.

Algorithm 2: Image Classification

Input: set of contribution down-projections of each subimage: DP_i
average contribution down-projection: DP_{av}
Output: score of each tile: *score*

foreach DP_i **do**
  $score_i = 0$
  `// n: number of locations used`
  for $k = 0$ **to** $n - 1$ **do**
    $(maxDiff, maxLoc) = \mathbf{max}\big(DP_i(t) - DP_{av}(t),\ \forall t \in DP_i,\ DP_i(t) \neq 0\big)$
    $DP_i(maxLoc) = 0$
    $score+ = abs(maxDiff.x)$
  $score_i = score_i/n$

Algorithm 3: Method Formal Algorithm

Input: original image: I_O
Output: result image: I_R

`// cf. Sec. 3`
I_D = denoise(I_O)
`// cf. Algo. 1`
H = densityHistogramGeneration(I_O, I_D)
`// cf. Sec. 5.1`
T = createTiles(I_O,I_D)
`// compute the contribution for each map`
foreach tile T **do**
  `// cf. Sec 4`
  DP_i = contributionDPGeneration(T_O, T_D)
`// cf. Sec 5.2`
DP_{av} = averageContributionGeneration(DP_i)
`// cf. Algo. 2`
classifyTiles(DP_i, DP_{av})

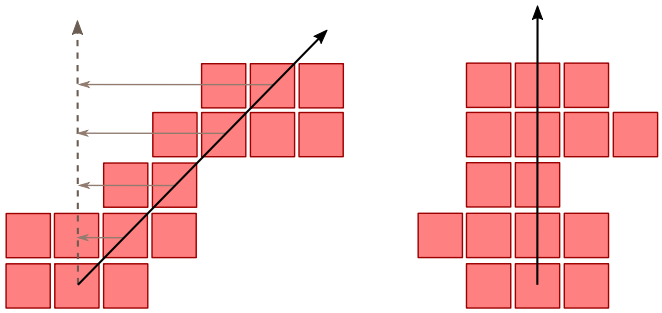


Figure 6: Deskewing the histogram. This is the first step before applying the down-projection.

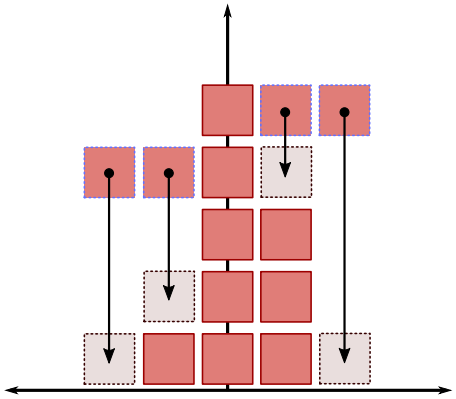


Figure 7: The process of down-projection.

6 Preprocessing

Due to the imperfect nature of the denoiser used, several steps are undertaken to decrease the impact of the denoising process on the final classification.
The first step is to remove the traces of JPEG blocking in both the original and the denoised image after denoising. Indeed, the denoiser used tends to consider the borders of the JPEG blocks as slightly more noised than the rest of the image, and as such denoise them more strongly. Although the suppression of the borders decreases the resolution of the image by around 43%, the removal of this added "artificial" noise increases the precision of the classification.

The second step consists in working around the contours in the original image. The denoising process has a tendancy to perform poorly around contours, and as such a replacement/elimination process is undertaken, in function of the amount of contour contained in a tile. We detect the contours in the image by applying a Laplacian filter, and then dilate them to ensure they contain all of the poorly denoised areas. Then, we check the percentage of pixels belong-

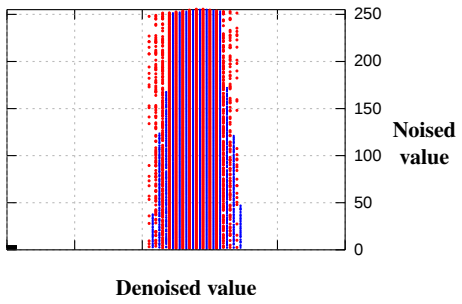
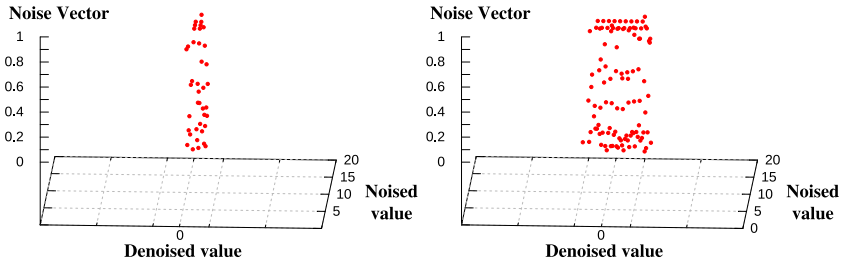


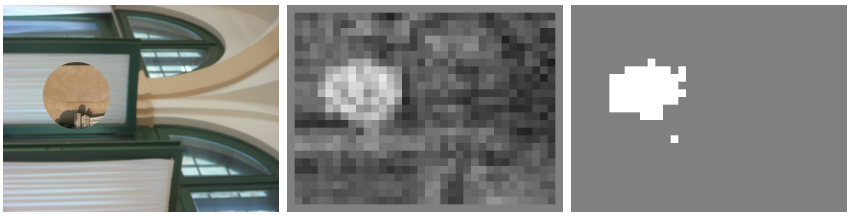
Figure 8: The skewed histogram and its down-projection side by side. We see that the down-projection is dense compared to the initial histogram.



(a) Contribution down-projection for a tile of the original image.

(b) Contribution down-projection for a tile of the spliced element.

Figure 9: The difference in contribution down-projections between a tile in the original image and one in the spliced element. We can see that the contributions of the spliced elementare much farther from the central axis on average.



(a) Spliced image

(b) Scaled result

(c) Thresholded result

Figure 10: An exemple of splicing detection.

ing to a dilated contour in each tile. If the percentage is lower than a defined threshold, set to 60% in our experimentations, we replace the contour pixels in the tile by random non-contour pixels also belonging to the tile. However, if the percentage is too high, we simply flag the tile so that it is removed from consideration for all subsequent computations. A similar process is applied to saturated tiles, depending on their number of pixels with either too high (>250)

or too low (<5) intensity.

Finally, in some cases the denoiser causes noise discrepancies that can appear even in non-spliced areas of the original image. This can be corrected by applying a linear amplification to the noise according to the image pixel intensity, with a minimal impact on the spliced element. This method is detailed in Algo. 4.

Algorithm 4: Noise preprocessing

Input: original image: I_O
noise image: I_N
Output: result noise image: I_{RN}

`// global scale coefficient`

$$k = \frac{\sum_{i,j} I_O(i,j).|I_N(i,j)|}{\sum_{i,j} \Big(I_O(i,j).|I_N(i,j)|\Big)^2}$$

`// new noise image using` I_O `intensity`

foreach pixel (i,j) **do**

$I_{RN} = k.I_O(i,j).I_N(i,j)$

7 Tests and Results

The method was tested on a set of 400 spliced images generated from 200 natural images from the Dresden dataset [13], with sizes between 2592×1944 and 3648×2736. This dataset was selected instead of the usual Columbia dataset [28] that has become outmoded, notably in term of image size and quality. The images used cover interior and exterior scenes, and were taken using different brands and models of cameras. The noise standard deviation range from 0.36 to 1.4. The algorithm runs in around 1 minute for the biggest images on consumer-grade hardware. The general results are are shown in Table 1.

Table 1: Average results in normal image condition

	F1 Score	MCC
Mahdian and Saic [25]	0.133	0.110
Zeng et al. [34]	0.078	0.098
He et al. [16]	0.139	0.120
Our Method	**0.203**	**0.178**

The results are computed using the F1 score [32] and the Matthews Correlation Coefficient (MCC) [26]. Indeed, although the F1 score is more frequently employed, in our database the spliced element tends to be small compared to the overall size of the image, usually around 1/16th of the image. As such, the

Table 2: Results with increased JPEG compression (using MCC)

	JPEG 95	JPEG 90	JPEG 85	JPEG 80
Mahdian and Saic [25]	0.110	0.068	0.068	0.073
Zeng et al. [34]	0.098	0.099	0.105	0.116
He et al. [16]	0.120	0.124	0.134	0.119
Our Method	**0.178**	**0.177**	**0.161**	**0.137**

MCC seems to be more adapted than the F1 score which gives the same weight to both areas, regardless of their respective sizes. This scoring method ranges between -1 and 1, with zero being a random guess on each pixel. Additionally, we ran our algorithm on a dataset of 300 artificial images, with a perfect denoising. The noise conditions were set to replicate those of natural images (ie, a noise standard deviation between 0.36 and 1.4, and the spliced element having randomly more or less noise than the base image). The average MCC on this dataset was 0.41. This shows a significant impact of imperfect denoising on the overall efficiency of our method. As such, it is reasonable to expect an improvement on detection capabilities as denoising methods will improve.

The difference in detection results can be seen in Fig. 11. We can note that on the previous methods' result images, the spliced element intensity tends to be roughly in the middle of the image intensity range, and not in an extreme. Consequently, an automated discrimination technique such as thresholding or clustering would not be able to separate it from the rest of the image. In comparison, our approach has the spliced element in an intensity extreme (either light or dark), and as such identifying it automatically is much easier.

Table 2 also shows that our approach is more impacted by a reduction in image quality than other state-of-the art methods, although our results still remain better than the alternatives. Finally, Table 3 shows that our method is also more resistant to strong downsampling.

The two most important factors to explain the robustness of our approach against compression and downsampling are the comparison to the global histogram and the post-processing enhancements. As compression and downsampling are applied equally across the whole image, the differentiation between the original pixels and the spliced is still possible, though the difference is attenuated. Using the global histogram down-projection allows us to reduce the impact that strong effects would have on small patches of the image. Although the final difference in scores is reduced in the result images, the normalization and thresholding applied afterwards enhances even the small variations in score, facilitating the final discrimination. This robustness can be seen as a consequence of the ability of our method to discriminate even on low-noise images.

Table 3: Results with downsampling (using MCC)

	Normal conditions	Downsample 75%	Downsample 50%
Mahdian and Saic [25]	0.110	0.093	0.010
Zeng et al. [34]	0.098	0.099	0.102
He et al. [16]	0.120	0.095	0.072
Our Method	**0.178**	**0.157**	**0.140**

8 Conclusion

In this paper, we present a new approach to detect splicing in JPEG images, based on the analysis of the noise density throughout the image. This approach uses a new tool titled down-projection contribution histogram to compare the contribution of a local area to the overall image noise with the average contribution throughout the image, with several pre- and post-processing operations. We have shown that our method performs better than other state of the art approaches on a dataset extracted from the Dresden image database, using two different metrics (the F1 score and the Matthews Correlation Coefficient). In addition, our method has been shown to be more robust to added alterations such as high compression or downsampling.

References

[1] S. Bayram, I. Avcibas, B. Sankur, and N. D. Memon. Image Manipulation Detection. *Electronic Imaging*, 15(4):1–17, 2006.

[2] Joel Brogan, Paolo Bestagini, Aparna Bharati, Allan Pinto, Daniel Moreira, Kevin Bowyer, Patrick Flynn, Anderson Rocha, and Walter Scheirer. Spotting the difference: Context retrieval and analysis for improved forgery detection and localization. *arXiv preprint arXiv:1705.00604*, 2017.

[3] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. Non-local means denoising. *Image Processing On Line*, 1:208–212, 2011.

[4] Wen Chen, Yun Q Shi, and Wei Su. Image splicing detection using 2-d phase congruency and statistical moments of characteristic function. In *Society of photo-optical instrumentation engineers (SPIE) conference series*, volume 6505, page 26, 2007.

[5] H Rand Chennamma and Lalitha Rangarajan. Image splicing detection using inherent lens radial distortion. *arXiv preprint arXiv:1105.4712*, 2011.

[6] Giovanni Chierchia, Giovanni Poggi, Carlo Sansone, and Luisa Verdoliva. A bayesian-mrf approach for prnu-based image forgery detection. *IEEE Transactions on Information Forensics and Security*, 9(4):554–567, 2014.

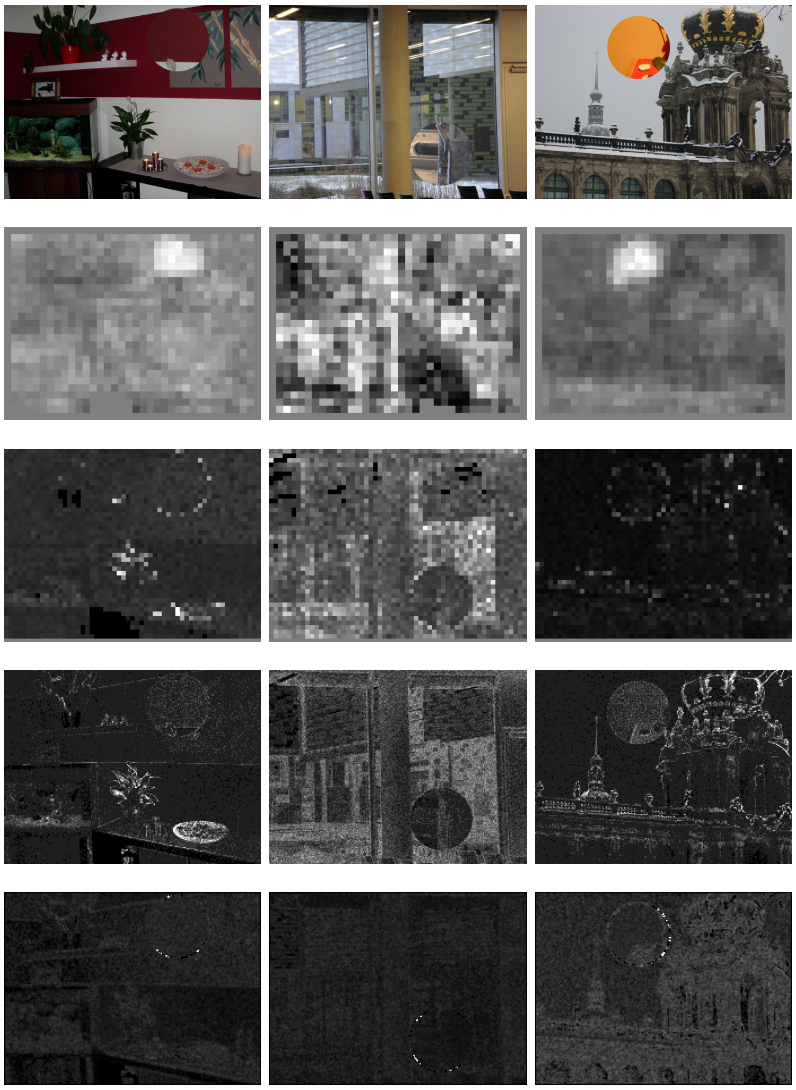


Figure 11: Results comparison. 11(a), 11(b) and 11(c) are the spliced images. 11(d), 11(e) and 11(f) are our results. 11(g), 11(h) and 11(i) are Zeng et al. [34] results. 11(j), 11(k) and 11(l) are Mahdian and Saic [25] results. 11(m), 11(n) and 11(o) are He et al. [16] results.

[7] Vincent Christlein, Christian Riess, Johannes Jordan, Corinna Riess, and Elli Angelopoulou. An evaluation of popular copy-move forgery detection approaches. *IEEE Transactions on information forensics and security*, 7(6):1841–1854, 2012.

[8] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image Denoising By Sparse 3D Transform-Domain Collaborative Filtering. *IEEE Trans. Image Process.*, 16(8):2080–2095, August 2007.

[9] H. Farid. A Survey Of Image Forgery Detection. *IEEE Signal Processing Magazine*, 26(2):26–25, 2009.

[10] H. Farid. Exposing Digital Forgeries From JPEG Ghosts. *IEEE Trans. Inf. Forensics Security*, 4(1):154–160, March 2009.

[11] Pasquale Ferrara, Tiziano Bianchi, Alessia De Rosa, and Alessandro Piva. Image forgery localization via fine-grained analysis of cfa artifacts. *IEEE Transactions on Information Forensics and Security*, 7(5):1566–1577, 2012.

[12] Andrew C Gallagher and Tsuhan Chen. Image authentication by detecting traces of demosaicing. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*, pages 1–8. IEEE, 2008.

[13] Thomas Gloe and Rainer Böhme. The dresden image database for benchmarking digital image forensics. *Journal of Digital Forensic Practice*, 3(2-4):150–159, 2010.

[14] Jong Goo Han, Tae Hee Park, Yong Ho Moon, and Il Kyu Eom. Efficient markov feature extraction method for image splicing detection using maximization and threshold expansion. *Journal of Electronic Imaging*, 25(2):023031–023031, 2016.

[15] Junfeng He, Zhouchen Lin, Lifeng Wang, and Xiaoou Tang. Detecting doctored jpeg images via dct coefficient analysis. In *European conference on computer vision*, pages 423–435. Springer, 2006.

[16] Xin He, Qingxiao Guan, Yanfei Tong, Xianfeng Zhao, and Haibo Yu. A novel robust image forensics algorithm based on l1-norm estimation. In *International Workshop on Digital Watermarking*, pages 145–158. Springer, 2016.

[17] Fangjun Huang, Jiwu Huang, and Yun Qing Shi. Detecting double jpeg compression with the same quantization matrix. *IEEE Transactions on Information Forensics and Security*, 5(4):848–856, 2010.

[18] Micah K Johnson and Hany Farid. Exposing digital forgeries through chromatic aberration. In *Proceedings of the 8th workshop on Multimedia and security*, pages 48–55. ACM, 2006.

[19] Micah K Johnson and Hany Farid. Exposing digital forgeries through specular highlights on the eye. In *International Workshop on Information Hiding*, pages 311–325. Springer, 2007.

[20] Thibault Julliand, Vincent Nozick, and Hugues Talbot. Automatic image splicing detection based on noise density analysis in raw images. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 126–134. Springer, 2016.

[21] Thibaut Julliand, Vincent Nozick, and Hugues Talbot. Image noise and digital image forensics. In *International Workshop on Digital Watermarking*, pages 3–17. Springer, 2015.

[22] Marc Lebrun. An analysis and implementation of the bm3d image denoising method. *Image Processing On Line*, 2:175–213, 2012.

[23] Z. Lin, J. He, X. Tang, and C. Tang. Fast, Automatic And Fine-Grained Tampered JPEG Images Detection Via DCT Coefficient Analysis. *Pattern Recognition*, 42(11):2492–2501, 2009.

[24] J. Luk, J. Fridrich, and M. Goljan. Detecting Digital Image Forgeries Using Sensor Pattern Noise. *Proc. SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VIII*, 6072:0Y1–0Y11, 2006.

[25] Babak Mahdian and Stanislav Saic. Using noise inconsistencies for blind image forensics. *Image and Vision Computing*, 27(10):1497–1503, 2009.

[26] Brian W Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451, 1975.

[27] T-T Ng and S-F Chang. A model for image splicing. In *Image Processing, 2004. ICIP'04. 2004 International Conference on*, volume 2, pages 1169–1172. IEEE, 2004.

[28] Tian-Tsong Ng, Jessie Hsu, and Shih-Fu Chang. Columbia image splicing detection evaluation dataset, 2009.

[29] Alin C Popescu and Hany Farid. Statistical tools for digital forensics. In *International Workshop on Information Hiding*, pages 128–147. Springer, 2004.

[30] C. Popescu and H. Farid. Exposing Digital Forgeries In Color Filter Array Interpolated Images. *Signal Processing, IEEE Transactions*, 53(10):1948–3959, September 2005.

[31] Judith A Redi, Wiem Taktak, and Jean-Luc Dugelay. Digital image forensics: a booklet for beginners. *Multimedia Tools and Applications*, 51(1):133–162, 2011.

[32] CJ Van Rijsbergen. Information retrieval. dept. of computer science, university of glasgow. *URL: citeseer. ist. psu. edu/vanrijsbergen79information. html*, 14, 1979.

[33] Wei Wang, Jing Dong, and Tieniu Tan. Effective image splicing detection based on image chroma. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 1257–1260. IEEE, 2009.

[34] Hui Zeng, Yifeng Zhan, Xiangui Kang, and Xiaodan Lin. Image splicing localization using pca-based noise level estimation. *Multimedia Tools and Applications*, pages 1–17, 2016.