Demosaicing to Detect Demosaicing and Image Forgeries

Quentin Bammey, Rafael Grompone von Gioi, Jean-Michel Morel

Demosaicing to Detect Demosaicing and Image Forgeries

Quentin Bammey*, Rafael Grompone von Gioi†, Jean-Michel Morel‡
Centre Borelli, École Normale Supérieure Paris-Saclay, Université Paris-Saclay
*ORCID: *0000-0003-2280-2349, †0000-0002-6309-7116, ‡0000-0002-6108-897X*

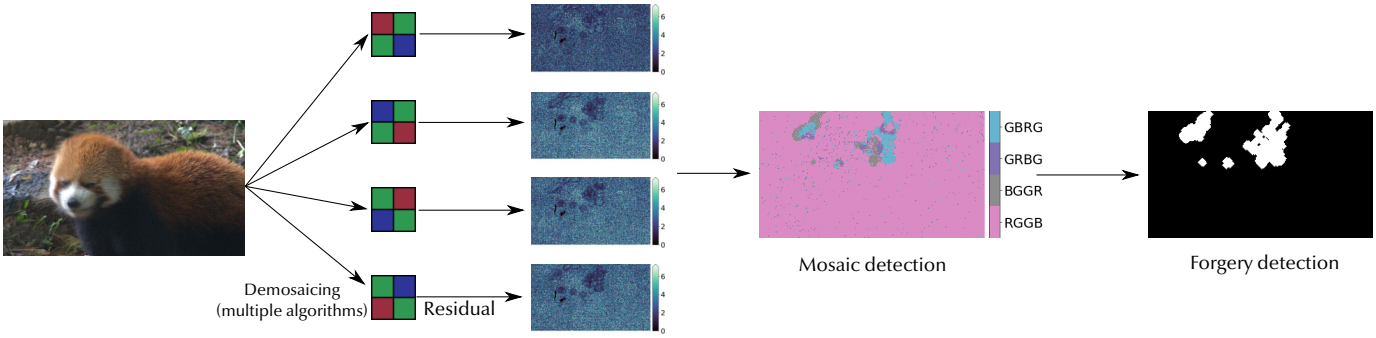


Fig. 1: The proposed method detects image forgeries by resimulating the mosaic in all four positions and demosaicing it again with multiple algorithms. The original mosaic is most likely to yield the lowest residual, thus enabling its identification. Regions with a different mosaic are then checked for forgery with an *a contrario* paradigm that enables control of false detections. Image from the Korus dataset [1].

***Abstract*—The trustfulness of images is a critical concern. Digital photographs can no longer be assumed to be truthful; indeed, digital image editing tools can easily and convincingly alter the semantic content of an image. Being able to analyse an image to check for forgeries becomes of the utmost importance in many domains, from police investigations to fact-checking and journalism. We propose here to analyse traces left by the camera during demosaicing, one of the first steps of image formation. When an object is added or displaced on an image, the demosaicing traces can be disrupted, leaving forgery clues. In order to detect these inconsistencies, we explore the possibilities offered by *double demosaicing*. Computing the demosaicing residual of an image with different demosaicing algorithms and patterns enables one to find image regions with inconsistent demosaicing traces. We render the method fully automatic by a simple *a contrario* scheme computing forgery detection thresholds with statistical guarantees on the number of false alarms.**

I. Introduction

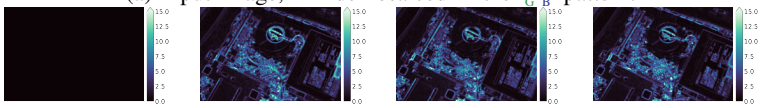
Once considered reliable evidence, photographic images can no longer be assumed to depict the naked truth. With the advent of digital photography and the progress of photo editing tools, altering a picture has never been easier. While most of these modifications solely seek to enhance the image, they can potentially alter its very semantics. Concealing, modifying, or adding a foreign object can give an image a new and false meaning. Although these forgeries can easily be made visually realistic, they still distort the very fabric of the image. The formation of a digital image, from the camera sensors to storage, leaves traces, which act like a signature for the image. Modifying an image distorts these traces, creating detectable inconsistencies.

Most cameras cannot see colour directly and instead place a colour filter array (CFA) in front of the sensors, so each pixel samples the value in one colour. By using filters of different colours on neighbouring pixels, the missing colours can then be interpolated. The Bayer CFA is used in almost all commercial cameras and is thus the focus of research on demosaicing detection. This matrix samples half the pixels in green, a quarter in red, and the last quarter in blue. Depending on the offset of the CFA, an image can thus be sampled in one of four patterns: $\begin{smallmatrix}R&G\\G&B\end{smallmatrix}$, $\begin{smallmatrix}B&G\\G&R\end{smallmatrix}$, $\begin{smallmatrix}G&R\\B&G\end{smallmatrix}$ or $\begin{smallmatrix}G&B\\R&G\end{smallmatrix}$. No demosaicing method is perfect – after all, it is a matter of reconstructing missing information – and each produces some level of artefacts. Therefore, it is possible to detect these artefacts to obtain information on the demosaicing method applied to the image.

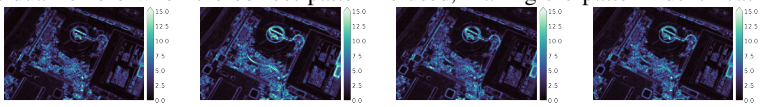
In this paper, we propose to analyse traces left by the process of demosaicing. Detecting its local inconsistencies yields forgery clues. Yet, most existing demosaicing analysis tools make unrealistic hypotheses on the linearity or channel-independence of the demosaicing operation. These hypotheses can strongly limit their ability to detect inconsistencies in general scenarios, as well as their robustness to JPEG compression. Instead, we propose to explore the potential of so-called *double demosaicing* to identify the CFA pattern. Local selection of the most suitable demosaicing algorithm enables one to detect the CFA pattern even when the original demosaicing algorithm is unknown.

Let S be the original colour image signal, which is not fully observed. Let $\mathcal{M}^{\begin{smallmatrix}R&G\\G&B\end{smallmatrix}}$ be the mosaic function for the $\begin{smallmatrix}R&G\\G&B\end{smallmatrix}$ pattern, which samples one colour on each pixel accord-

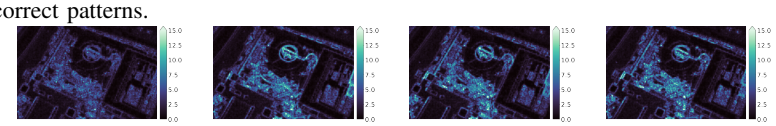
(a) Input image, HA-demosaiced in the $\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}$ pattern.



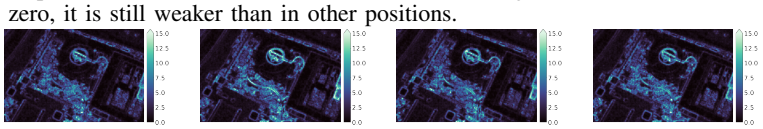
(b) Residuals when the input image is demosaiced again with the same algorithm (HA) in the four positions, from left to right $\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}$ (correct pattern), $\begin{smallmatrix}\mathrm{B}&\mathrm{G}\\\mathrm{G}&\mathrm{R}\end{smallmatrix}$, $\begin{smallmatrix}\mathrm{G}&\mathrm{R}\\\mathrm{B}&\mathrm{G}\end{smallmatrix}$, $\begin{smallmatrix}\mathrm{G}&\mathrm{B}\\\mathrm{R}&\mathrm{G}\end{smallmatrix}$. The residual is zero when the correct pattern is used, making the pattern identification easy.



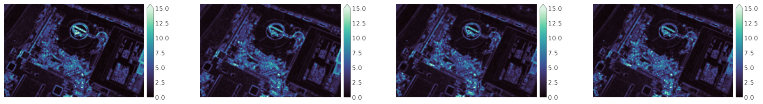
(c) Here, the input image is analysed with a different algorithm (LMMSE). The residual in the correct pattern (left) is no longer zero, but is still weaker than on the incorrect patterns.



(d) The input image is JPEG-compressed ($Q = 90$ before the second demosaicing), with the same (HA) algorithm. Again, although the correct pattern's residual is not zero, it is still weaker than in other positions.



(e) The input image is JPEG-compressed ($Q = 90$ before the second demosaicing, with a different (LMMSE) algorithm. The residual is still weaker with the correct pattern.



(f) Here, the (uncompressed) input image is analysed with yet another algorithm (ARI).

Fig. 2: An image [2] was demosaiced with HA and analysed with several algorithms. When the image is not compressed between the two demosaicing operations, and if the algorithm used is the same, the residual will be zero in the correct mosaic pattern. When the image is compressed, the residual is no longer zero in the correct pattern, but is still weaker than in the other patterns as long as the compression is moderate. If a different algorithm is used for analysis, results vary: Analysing the image with LMMSE yields a lower residual in the correct pattern, but this is no longer the case when using ARI, which is not suited to analyse HA-demosaiced images. Residuals are blurred ($\sigma = 1.7$) for better visualization.

ing to the pattern. $\mathcal{M}\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}S$ is the observed image on the sensor, of size $(2X, 2Y)$, mosaiced in the $\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}$ pattern. The image is demosaiced with algorithm A, yielding the image $I \triangleq \mathcal{D}^A\mathcal{M}\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}S$ of size $(2X, 2Y, 3)$, which we observe and analyse. The goal is to detect this image was demosaiced with the $\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}$ pattern.

Demosaicing interpolates missing colour values from the image, but it does not modify the already-known values. What happens then if we artificially recreate a mosaic on I, then demosaic it again? Mosaicing $\mathcal{D}^A\mathcal{M}\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}S$ in its original $\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}$ position yields the original mosaic; as a consequence, when performing the remosaicing-demosaicing operation with the original CFA and algorithm A, the image is unchanged:

$$\mathcal{D}^A\mathcal{M}\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}I = \mathcal{D}^A\mathcal{M}\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}\mathcal{D}^A\mathcal{M}\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}S = \mathcal{D}^A\mathcal{M}\begin{smallmatrix}\mathrm{R}&\mathrm{G}\\\mathrm{G}&\mathrm{B}\end{smallmatrix}S = I. \tag{1}$$

On the other hand, if we are using the wrong CFA pattern, the originally-sampled values are lost, and the final demosaicing will reconstruct a slightly different image.

Assuming the demosaicing algorithm A is known, it is thus easy to find the original CFA pattern of an image, by remosaicing and demosaicing it again on the four possible patterns, and looking for the pattern with which the image was unchanged. Based on this, Kirchner [3] proposed to find the CFA pattern of an image by demosaicing in the four patterns with bilinear interpolation, and looking at the pattern with the lowest residual. We follow on this idea by using multiple demosaicing algorithms instead of just bilinear interpolation. Indeed, although the assumption of bilinear demosaicing was reasonable when the article was published, most common demosaicing methods nowadays are much more advanced.

Figure 2 shows a visual example of one image in different scenarios. When the original demosaicing and analysis algorithms are the same, the residual is zero on the correct

pattern, yielding easy detection. In the compressed case, the residual is no longer zero in the correct position, but it is still weaker than in other positions, and the detection is still possible. When using different algorithms for demosaicing and analysis, detection is still possible as long as one is using a close enough demosaicing method – hence the necessity of trying several algorithms to select the best-matching one.

In this article, we propose to **analyse and extend double demosaicing** to adapt to different demosaicing algorithms. Automatic selection of the **most suited algorithm** enables a more robust understanding of the studied image. Local analysis of the results on an image can then be used to **detect inconsistencies** and thus forgeries. An *a contrario* paradigm is then used to **automatically** detect forgeries without need for expert analysis. Our studies show that this double demosaicing scheme can be used to analyse an image, even in compressed scenarios, to understand its original processing as well as to detect its inconsistencies and potential forgeries.

II. Related Works

In a pioneer paper on demosaicing analysis, Popescu and Farid [4] jointly estimate a linear model for the demosaicing algorithm and detect sampled pixels. In demosaiced regions, the locations of detected pixels are 2-periodic, the local absence of this periodic component hints at the absence of demosaicing in a region. Ferrara [5] also look for the local absence of demosaicing traces. A fixed predictor looks at the difference of variance between the two lattices of supposedly-sampled and interpolated pixels: in the presence of demosaicing, the variance should be significantly higher on the former. Kirchner [6], whose work we extend, performs bilinear demosaicing in the four possible patterns to identify the true sampling pattern as the one where the residual is lowest. Similarly, Milani et al. [7] applies demosaicing with several algorithms on the supposed pattern to identify the image's true demosaicing algorithm. Choi et al. [8], [9] notice that interpolated pixels are more likely to be intermediate values than their neighbours. They propose to compare the counts of intermediate values in each lattice to estimate the correct pattern. Bammey [10] makes use of positional training to make the network implicitly replicate the underlying mosaic pattern of an image. Very recently, Park [11] avoids using reinterpolation, so as not to be subject to the suitability of the hard-to-estimate kernel. Instead, a channel-wise SVD removes background information to yield a more reliable residual.

While the most recent methods overcome to some extent the variety of non-linear, interchannel-transfers-rich demosaicing algorithms, they still struggle to make detections on compressed images, even at a very high quality. Although it is likely impossible to detect demosaicing artefacts in strongly-compressed images, some degree of robustness is desirable for applicability in many fields where compressed but high-quality images are found, such as criminal investigations.

III. Methodology

Two inconsistencies can be used as evidence of forgeries: a difference in the detected pattern of a region, and a difference in the best algorithm that can be used to approximate said region. We analyse both features in parallel with an *a contrario* paradigm. Most forgery detection methods do not perform automatic detection. Instead, they yield a heatmap of regions that appear to be forged, and let the user decide on whether the image is indeed forged. However, the user trying to know whether an image is forged does not necessarily have the knowledge to interpret the results. Furthermore, visual analysis of all images is not possible if many images are to be inspected. To make the detection truly automatic, *a contrario* analysis [12] can prove useful. It has already been successfully applied to other forgery detection methods [13], [14]. Based on Gestalt theory, this detection paradigm computes automatic thresholds on the heat map by controlling an upper bound of the number of false alarms (NFA) one might expect. This is obtained by a threshold on the p-value deviance to a background hypothesis. This threshold is chosen so as to limit the expected number of images which would wrongly be detected as forged while under the background hypothesis. The NFA of a detection belongs in $(0, +\infty)$, with scores closer to 0 corresponding to more significant detections. An NFA of 10^{-3}, means that under the background hypothesis, we can expect at most one false detection every 1000 images.

To do this, we aggregate the image into 2×2 blocks, so as to neglect the variance caused by the different results on different positions of a mosaic. Indeed, one could expect different accuracy and bias where the pixel is sampled in different colours, so aggregating the 2×2 blocks removes this volatility. Each block is then made to vote for the best algorithm that can approximate it, the pattern on which the lowest residual is achieved, and the diagonal on which it is achieved ($\begin{smallmatrix}\cdot & \mathrm{G}\\ \mathrm{G} & \cdot\end{smallmatrix}$ if the pattern is $\begin{smallmatrix}\mathrm{R} & \mathrm{G}\\ \mathrm{G} & \mathrm{B}\end{smallmatrix}$ or $\begin{smallmatrix}\mathrm{B} & \mathrm{G}\\ \mathrm{G} & \mathrm{R}\end{smallmatrix}$, $\begin{smallmatrix}\mathrm{G} & \cdot\\ \cdot & \mathrm{G}\end{smallmatrix}$ otherwise). Separation between full and diagonal patterns is used to capture the many cases where the diagonal can be estimated correctly, but not the full pattern, as is classical in demosaicing analysis [8], [15], [16].

The globally best algorithm, diagonal and full pattern, is the mode of all 2×2 block votes. However, one wants to know whether this detection is indeed significant, or could come from random-like block votes. This can be easily checked; in the absence of demosaicing one could expect those votes to be distributed uniformly, and the count of votes for one grid would follow a binomial distribution. The frequency at which one could obtain such a significant detection in the absence of demosaicing is thus bounded from above by the number N of possible votes (4 if voting for the patterns, 2 for the diagonals, and the number of algorithms when voting for the algorithm), multiplying the survival function of the binomial distribution, $N \operatorname{Binom}_{sf}\left(k, n, \frac{1}{N}\right)$, where k is the number of votes for the best value and n the total number of blocks. Thresholding on this value enables automatic detections while keeping control over the expected number of false detection in noise.

If the global detection is significant, we declare to know

which algorithm is globally best, or the best pattern or diagonal for the image. We then try to detect local inconsistencies over these values. In each overlapping window, of size $W \times W$, if the locally-best value is different from the global one, then we again use the binomial distribution to decide on the significance of the detection. To fully control the number of false alarms, we further multiply the result by the number of windows. This enables one to keep control, not over the number of blocks that would be falsely detected in authentic images, but more directly over the frequency at which *one image* would be detected as forged. The resulting number of false alarms for the detection is thus

$$NFA = n_{\text{windows}} N \operatorname{Binom}_{sf}\left(k, W^2, \frac{1}{N}\right). \qquad (2)$$

Thresholding over this yields an automatic decision. We threshold the outputs at 10^{-3}, so we can expect one false rejection of the background hypothesis every thousand images.

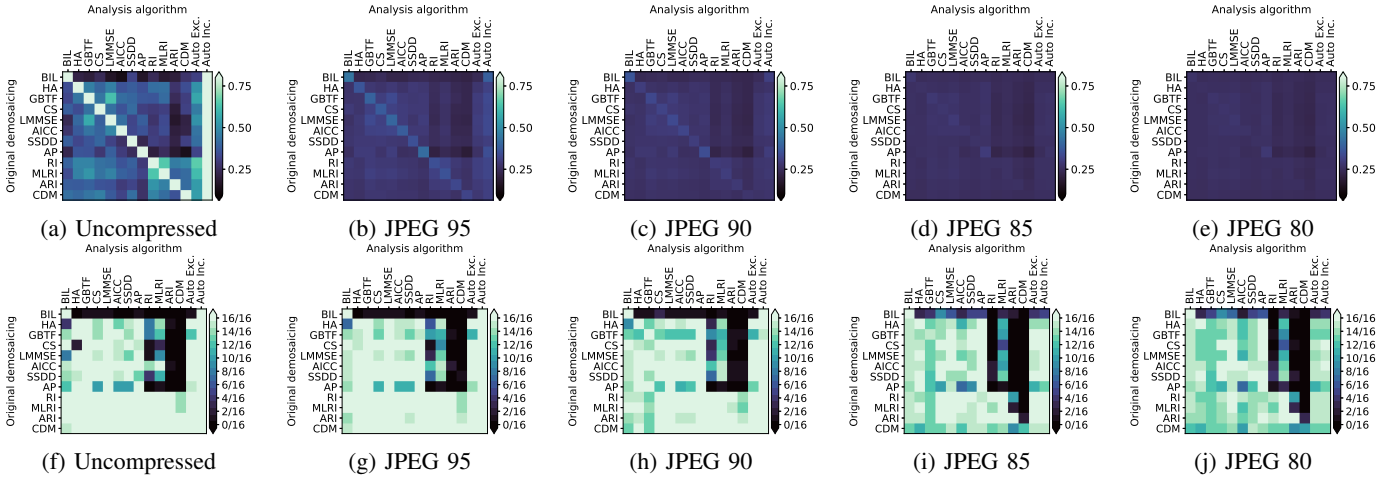
IV. Experiments

The authors of [2] have provided 18 images clean of any demosaicing traces, with varying levels of textures. We recreate a demosaiced version of those images with several algorithms, on a single position: bilinear demosaicing, contour stencils (CS) [17], Hamilton-Adams (HA) [18], Linear Minimum Mean-Square-Error Estimation (LMMSE) [19], Alternating Projections (AP) [20], self-similarity-driven demosaicing (SSDD) [21], CDM-CNN (CDM) [22], gradient-based threshold-free (GBTF), residual interpolation (RI) [23], minimized-Laplacian residual interpolation (MLRI) [24], and adaptive residual interpolation (ARI) [25]. We then demosaic the image a second time in all four positions with the array of algorithms, and check whether the lowest residual is found on the correct pattern. We make this check with each individual pair of algorithms, as well as with the closest-found algorithm, whose selection is done both taking into account the original demosaicing method and excluding it. This corresponds to testing the method in scenarios where the image was originally demosaiced with an algorithm that either belongs in the list of tested methods, or not. To check robustness against JPEG compression, we include an optional compression step between the two demosaicing operations. We look at the proportion of 2×2 blocks that are correctly detected, as well as the number of images where most 2×2 blocks are detected on the correct pattern. Results can be seen in Fig. 3. It is unsurprisingly easier to detect the correct pattern when the demosaicing algorithm is known and the image uncompressed. While the results are poorer on more difficult scenarios, automatic search of the best algorithm enables detection of the correct pattern at a better rate than random, even on JPEG-80 images where the original algorithm is excluded from the tests. While this is not enough for analysis at a very local scale, it enables the detection of the pattern on a more global scale. Surprisingly, analysing bilinear-demosaiced images with other algorithms yields poor results. This is probably due to the fact that other methods have a very different behaviour from bilinear demosaicing in many cases, so as to provide a better image. More globally, images demosaiced with complex algorithms such as ARI can easily be analysed with simpler ones, whereas said complex algorithms perform poorly to analyse other images. This leads us to only use a selection of six algorithms for the rest of the study: bilinear demosaicing, CS, AP, LMMSE, HA and GBTF. While those are not the best demosaicing methods, they form a basis that is shown to be very good at analysing most images, including those demosaiced with SOTA algorithms.

	Grid	Alg	Alg (J95)	Alg (J90)	runtime (s)
Ours	0.401	0.415	<u>0.135</u>	<u>0.102</u>	7.2
Ours, `ALG`	0.003	0.244	0.098	0.087	7.2
Ours, `GRID`	0.422	0.325	0.065	0.039	7.2
4Point [26]	**0.709**	**0.523**	**0.307**	**0.151**	642
Bammey [10]	<u>0.682</u>	<u>0.501</u>	0.005	0.003	40.6
Shin [16]	0.104	0.085	0.001	0.001	5.4
Choi [8], [9]	0.603	0.420	0.108	0.049	5.7
Ferrara [5]	0.071	0.218	0.005	0.000	5.6
Dirik [27]	-0.002	0.001	0.000	0.001	36.4
Park [11]	0.116	0.152	0.003	0.001	48.3
Noiseprint [28]	-0.001	0.066	0.013	0.005	32.5
Splicebuster [29]	0.003	0.075	0.015	0.002	23.9

TABLE I: Comparative results of the method on the Trace [30] database, with the CFA Grid (Grid) and CFA Algorithm (Alg) datasets. We use the Matthews Correlation Coefficient (MCC) as metric. The score varies from -1 to 1, with random baseline at 0 and perfect detection at 1. The CFA Algorithm dataset is further tested with JPEG compression at quality factors 95 (J95) and 90 (J90). The proposed method is tested in full, as well as when looking separately at detected algorithm discrepancies (`ALG`) or mosaic pattern shifts, both in the full patterns and diagonals (`GRID`). While pattern inconsistencies are easier to spot than demosaicing algorithm inconsistencies on uncompressed images, this order is reversed on JPEG-compressed images, where algorithm detections make our method the most robust against JPEG compression. Only two other methods feature robustness to JPEG compression, Choi [8], [9] and 4Point [26]. Although 4Point yields better results than all other methods, including the proposed one, it requires fine-tuning a CNN on every image to be tested, a costly process that cannot be carried on large amounts of images. Furthermore, both 4Point and Choi are solely based on detecting mosaic pattern shifts, which leads us to believe that our detections are complementary in nature. Runtime average for one 2474×1640 image, 12-cores intel i7 G8, Nvidia Quadro P3200.

We now study the ability of double demosaicing to detect inconsistencies of the CFA pattern. The Trace [30] database introduces invisible forgery traces in the form of pipeline inconsistencies. We use two of its datasets; the "CFA grid" dataset, in which the forged region is demosaiced in a different pattern; and the "CFA algorithm" dataset, in which the forged region is demosaiced with a different algorithm, and a new demosaicing pattern is selected, which thus has a $\frac{3}{4}$ chance to be different from the authentic region. On the CFA algorithm dataset, we further study our robustness to JPEG compression



(a) Uncompressed (b) JPEG 95 (c) JPEG 90 (d) JPEG 85 (e) JPEG 80

(f) Uncompressed (g) JPEG 95 (h) JPEG 90 (i) JPEG 85 (j) JPEG 80

Fig. 3: Percentage of 2×2 blocks whose residual is lowest in the correct pattern depending on the original demosaicing and the algorithm used for analysis (top), and number of images on which most 2×2 blocks are detected in the correct pattern (bottom). On compressed data, `Auto Exc.` (resp.`Inclusive`) refers to selecting the pattern with the lowest residual across all analysis demosaicking algorithms, excluding (resp. including) the algorithm that was used in the original demosaicing process. It is unsurprisingly easier to detect the correct pattern when the demosaicing algorithm is known and the image uncompressed. While the results are poorer on more difficult scenarios, automatic search of the best algorithm enables detection of the correct pattern at a better rate than random, even on JPEG-80 images where the original algorithm is excluded from the tests (`Auto Exc.`). This enables the detection of the pattern on a global scale (the tested images have a size of 704×469 pixels).

by compressing all images before testing.

The experiments were run with automatic selection of the algorithm. Note that while 15% of the images in the Trace database use bilinear demosaicing, none of the remaining demosaicing algorithms are used in our method. This selection provides a challenging opportunity to further test how the method generalizes to unseen algorithms.

Results on the Trace [30] database, with the CFA Grid (Grid) and CFA Algorithm (Alg) datasets, are presented in Table I. We compare our method with six demosaicing-based forensic tools, Bammey [10], Shin [16], Choi [8], [9], Ferrara [5], Dirik [27], Park [11], and with two generic forgery detection methods Splicebuster [29] and Noiseprint [28]. The original Choi and Shin methods [8], [16] only detect mosaic patterns, but do not detect forgeries.[9] extends [8] to find discrepancies, we also port this extension to [16]. The implementations for Ferrara [5] and Dirik [27] are taken from [31]. Bammey [10], Park [11], Splicebuster [29] and Noiseprint [28] are used directly with the authors' implementations.

To measure detections, we provide scores with the Matthews Correlation Coefficient (MCC), whose values lie between 1 (perfect detection) and -1 (its complementary). Any input-independent method has an expected score of 0. Although our method provides automatic, binary detections, other tested methods provide a continuous heatmap: we thus use the best threshold over the dataset. Note that this provides a scoring advantage to those methods, compared to the proposed work which automatically thresholds the data.

Against uncompressed images, our method comes third on the CFA Grid dataset, after Bammey [10] and Choi [8], [9], and second on the CFA Algorithm dataset, after Bammey [10]. On compressed images, however, our method comes first by a large margin, with Choi being the only other method providing some robustness to JPEG compression.

As an ablation, we tested our method when only using detection separately from algorithm inconsistencies, and from shifts in the mosaic, both with the full pattern and diagonal. Results are presented on the same Table I. On uncompressed images, mosaic shifts are quite easy to detect and most of the score comes from these. This situation is reversed on compressed images; pattern shifts are much harder to detect with our method, while discrepancies over the detected demosaicing keep more robustness. We conclude that our results on compressed images mainly stem from algorithm discrepancies, and are thus complementary to Choi's results, a method based solely on pattern detection.

V. Discussion

In this article, we have studied the possibility of using double demosaicing to detect the pattern of an image. When an image has been through demosaicing in an unknown pattern, a second demosaicing can be applied in all four patterns. No information is lost when the initial pattern is used for the second demosaiced. As a consequence, the residual is lower or even zero in the correct pattern than in the other positions. A simple strategy is enough to select the best-matching demosaicing algorithm among a list.

Overall, this method can be used to analyse the image at a global scale, helped by its surprisingly good robustness to JPEG compression. We then applied an *a contrario* scheme to automatically detect and localize image forgeries. While some of the robustness to JPEG compression is inevitably lost

at a local scale, the proposed method is still quite robust to compression. Furthermore, double demosaicing analysis can provide useful information on the original demosaicing of an image, and is thus a valuable tool in reverse-engineering the image processing pipeline used by a given image, which could in turn be used for image authentication.

Demosaicing detection methods now perform quite well on uncompressed images, it is now necessary to focus on improving their robustness to JPEG compression. While it is unlikely that detecting demosaicing inconsistencies on low-quality images could be possible, performing robust detection at a quality factor above 90 now seems possible, if not easy, and is of critical security importance to analyse images in contexts such as police investigations, where images are often of a high enough quality. Even if demosaicing analysis cannot suffice on such images, it can serve as an excellent complement to other tools, given that all existing non-demosaicing-specific forensic tools are mostly blind to demosaicing traces.

References

[1] P. Korus and J. Huang, "Evaluation of random field models in multi-modal unsupervised tampering localization," in *IEEE WIFS*, 2016.

[2] M. Colom, *Noise-free test images dataset*. [Online]. Available: http://mcolom.info/pages/no_noise_images/.

[3] M. Kirchner and R. Böhme, "Synthesis of color filter array pattern in digital images.," *Media Forensics and Security*, vol. 7254, p. 72 540, 2009.

[4] A. C. Popescu and H. Farid, "Exposing digital forgeries in color filter array interpolated images," *IEEE Transactions on Signal Processing*, 2005.

[5] P. Ferrara, T. Bianchi, A. De Rosa, and A. Piva, "Image forgery localization via fine-grained analysis of cfa artifacts," *IEEE TIFS*, vol. 7, no. 5, 2012.

[6] M. Kirchner and J. Fridrich, "On detection of median filtering in digital images," in *MFS II*, 2010.

[7] S. Milani, P. Bestagini, M. Tagliasacchi, and S. Tubaro, "Demosaicing strategy identification via eigenalgorithms," in *ICASSP*, 2014.

[8] C.-H. Choi, J.-H. Choi, and H.-K. Lee, "CFA pattern identification of digital cameras using intermediate value counting," in *MM&Sec*, ACM, 2011.

[9] Q. Bammey, R. Grompone von Gioi, and J.-M. Morel, "Image Forgeries Detection through Mosaic Analysis: the Intermediate Values Algorithm," *IPOL*, 2021.

[10] Q. Bammey, R. G. von Gioi, and J.-M. Morel, "An adaptive neural network for unsupervised mosaic consistency analysis in image forensics," in *CVPR*, 2020.

[11] C. W. Park, Y. H. Moon, and I. K. Eom, "Image tampering localization using demosaicing patterns and singular value based prediction residue," 2021.

[12] A. Desolneux, L. Moisan, and J.-M. Morel, "Meaningful alignments," *IJCV*, 2000.

[13] T. Nikoukhah, J. Anger, T. Ehret, M. Colom, J.-M. Morel, and R. Grompone von Gioi, "JPEG grid detection based on the number of DCT zeros and its application to automatic and localized forgery detection," in *CVPRW*, 2019.

[14] M. Gardella, P. Musé, J.-M. Morel, and M. Colom, "Noisesniffer: A fully automatic image forgery detector based on noise analysis," in *IWBF*, IEEE, 2021.

[15] M. Kirchner, "Efficient estimation of CFA pattern configuration in digital camera images," in *MFS*, 2010.

[16] H. J. Shin, J. J. Jeon, and I. K. Eom, "Color filter array pattern identification using variance of color difference image," *Journal of Electronic Imaging*, 2017.

[17] P. Getreuer, "Image Demosaicking with Contour Stencils," *IPOL*, 2012. DOI: 10.5201/ipol.2012.g-dwcs.

[18] J. F. Hamilton Jr and J. E. Adams Jr, *Adaptive color plan interpolation in single sensor color electronic camera*, US Patent 5,629,734, May 1997.

[19] P. Getreuer, "Zhang-Wu Directional LMMSE Image Demosaicking," *IPOL*, 2011.

[20] B. K. Gunturk, Y. Altunbasak, and R. M. Mersereau, "Color plane interpolation using alternating projections," *IEEE TIP*, DOI: 10.1109/TIP.2002.801121.

[21] A. Buades, B. Coll, J. M. Morel, and C. Sbert, "Self-similarity Driven Demosaicking," *IPOL*, vol. 1, pp. 51–56, 2011. DOI: 10.5201/ipol.2011.bcms-ssdd.

[22] R. Tan, K. Zhang, W. Zuo, and L. Zhang, "Color image demosaicking via deep residual learning," in *ICME*, 2017, pp. 793–798.

[23] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, "Residual interpolation for color image demosaicking," in *ICIP*, 2013, pp. 2304–2308.

[24] ——, "Minimized-laplacian residual interpolation for color image demosaicking," in *Digital Photography X*, International Society for Optics and Photonics, vol. 9023, 2014, p. 90230L.

[25] Y. Monno, D. Kiku, M. Tanaka, and M. Okutomi, "Adaptive residual interpolation for color image demosaicking," in *ICIP*, IEEE, 2015, pp. 3861–3865.

[26] Q. Bammey, R. G. von Gioi, and J.-M. Morel, "Forgery detection by internal positional learning of demosaicing traces," in *WACV*, Jan. 2022, pp. 328–338.

[27] A. E. Dirik and N. Memon, "Image tamper detection based on demosaicing artifacts," in *ICIP*, 2009.

[28] D. Cozzolino and L. Verdoliva, "Noiseprint: A cnn-based camera model fingerprint," *IEEE TIFS*, 2020.

[29] D. Cozzolino, G. Poggi, and L. Verdoliva, "Splicebuster: A new blind image splicing detector," in *WIFS*, 2015.

[30] Q. Bammey, T. Nikoukhah, M. Gardella, R. G. von Gioi, M. Colom, and J.-M. Morel, "Non-semantic evaluation of image forensics tools: Methodology and database," in *WACV*, Jan. 2022, pp. 3751–3760.

[31] M. Zampoglou, S. Papadopoulos, and Y. Kompatsiaris, "Large-scale evaluation of splicing localization algorithms for web images," *Multimedia Tools and Applications*, vol. 76, no. 4, pp. 4801–4834, 2017.