# Towards a reliable detection of forgeries based on demosaicing

Jérémie Dentan
jeremie.dentan@polytechnique.org
École Polytechnique
France

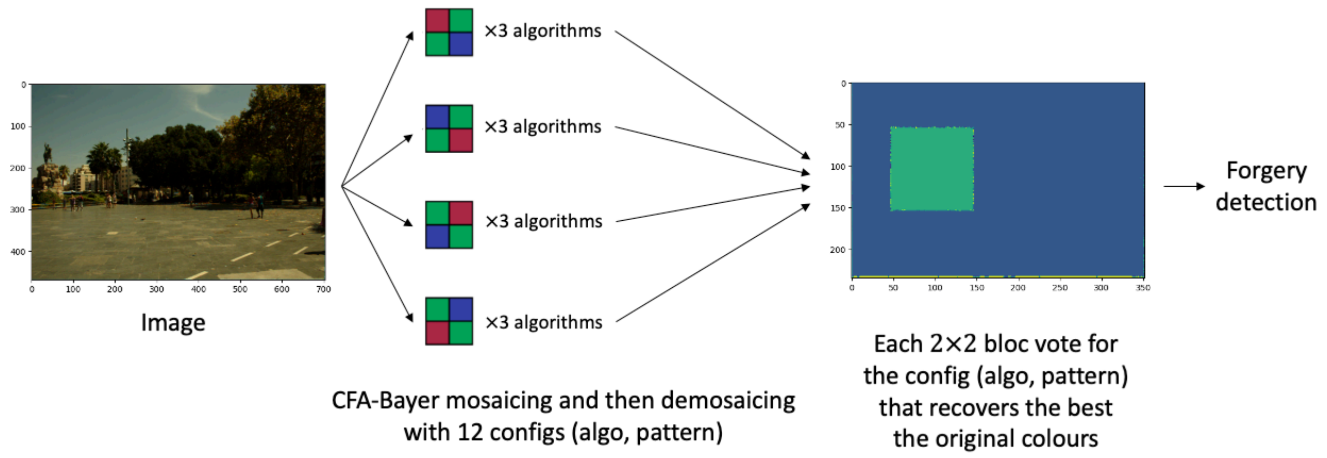
Image
×3 algorithms
×3 algorithms
×3 algorithms
×3 algorithms
CFA-Bayer mosaicing and then demosaicing with 12 configs (algo, pattern)
Each 2×2 bloc vote for the config (algo, pattern) that recovers the best the original colours
Forgery detection

**Figure 1: General overview of the forgery detection pipeline.** The method is based on double demosaicing: an image to be tested is demosaiced using several algorithms and patterns; if several areas of the image are better reconstructed with two different configurations, we can then deduce that one of these areas has been faked.

## ABSTRACT

This is a review of the paper "Demosaicing to Detect Demosaicing and Image Forgeries" written by Quentin Bammey, Rafael Grompone von Gioi, and Jean-Michel Morel [1], which proposes a demosaicing-based forgery detection method.

More than the general performance of the method proposed by the author, our review mainly focus on the reliability of the detection performed, especially in terms of false detections. Thus, our main objective has been to verify that the assumptions made by on the a-contrario model are valid and lead to reliable detections, quantified by a robust Number of False Alarms (NFA).

The results of our experiments show that despite its good performances in the absence of compression, this forgery detection method leads to an abnormally high number of false alarms compared to the theoretical guarantees of the a-contrario model, when the images are compressed. This proves that the assumptions of this model are not verified. Admittedly, we used a smaller number of demosaicing algorithms for our experiments, and older algorithms. However, at no point does the paper discuss either the assumptions made about these algorithms, or their impact on the reliability of the detection.

Thus, our study shows the importance of empirical or theoretical validation of the set of demosaicing algorithms used for forgery detection, a necessary condition to obtain reliable and industrially usable detections. In particular, we show that the set of algorithms used must (1) present similar performances for any level of image compression and (2) present similar performances for any type of textures on natural images.

## KEYWORDS

Demosaicing, Forgery Detection, A-contrario model, Number of False Alarm (NFA)

## 1 INTRODUCTION

This technical report comes with several resources:

- A GitHub repository that implements all the experiments made for this review, available here
- Two datasets: one provided by [3] available here; one provided by the author of this review [4] and available here
- Two notebooks that simply illustrate the methods and assumptions made by this review, and can be run independently, available here and here

### 1.1 Context

The trustworthiness of images is a significant issue, as digital images can now be easily and convincingly altered using editing tools, making it difficult to assume their accuracy. This creates a pressing need for image analysis in various fields, such as police investigations, fact-checking, and journalism, to detect forgeries. The solution proposed by the authors involves analyzing the traces left by the camera during demosaicing, an initial stage of image formation. Disruption of these traces when an object is added or displaced on an image can leave behind forgery clues.

Most modern cameras cannot detect colors directly, which is why they have a Colour Filter Array (CFA) in front of their sensor. Thus, each pixel perceives only the color corresponding to its filter, and the other colors are interpolated from the values of the neighboring pixels. The Bayer CFA filter is the most widely used nowadays, and that's why it focuses all the attention of the demosaicing detection research.

For this paper review, we will adopt the following definitions:

- **Mosaicing:** The action of applying a Bayer CFA to an existing image. This steps tries to reproduce a mosaiced image as it would be right after the sensors of a camera. Depending on the offset of the CFA, the mosaicing can be done in one of the four patterns (cf. below).
- **Demosaicing:** Given a mosaiced image, interpolate the RGB colour values of every pixels using the values of the neighboring pixels. The pattern used for the demosaicing should be the same as the one used for the mosaicing.
- **Demosaicing algorithm:** An algorithm that performs demosaicing. For this paper review, we will use the three demosaicing algorithms that are available in the Python `colour-demosaicing` library: `bilinear`: simple bilinear interpolation; `malvar`: an algorithm proposed by Malvar et al. in 2004 [7]; `menon`: an algorithm proposed by Menon et al. in 2007 [8].
- **Pattern:** one of the four following offset of the CFA: $\begin{smallmatrix}R&G\\G&B\end{smallmatrix}$, $\begin{smallmatrix}B&G\\G&R\end{smallmatrix}$, $\begin{smallmatrix}G&B\\R&G\end{smallmatrix}$, $\begin{smallmatrix}G&R\\B&G\end{smallmatrix}$.
- **Configuration:** A tuple (demosaicing algorithm, pattern). With our settings, there are 12 of them.
- **Diagonal:** The orientation of the green diagonal in a pattern: 'up' for $\begin{smallmatrix}R&G\\G&B\end{smallmatrix}$and $\begin{smallmatrix}B&G\\G&R\end{smallmatrix}$, and 'down' for $\begin{smallmatrix}G&B\\R&G\end{smallmatrix}$and $\begin{smallmatrix}G&R\\B&G\end{smallmatrix}$. Indeed, sometimes the diagonal is easier to detect than the precise pattern.
- **Forge:** The action of enforcing the demosaicing configuration of an image. This is done by a mosaicing step immediately followed by a demosaicing step, using the pattern and the demosaicing algorithm we want to enforce. An optional JPEG compression can be done after the forging.
- **Fake:** The action of adding a distortion in the demosaicing of an image (forged or not). This is done by selecting an area in the image, and forging it with a configuration (that is different from the one of the full image, if it has one).

## 1.2 The method of the article

The article describes a method for detecting the demosaicing artefacts in an image. More precisely, there are two types of detections that are discussed in the article, and the different sections of this review discuss separately those detections, even though they are interconnected:

- The detection of the demosaicing configuration that was used during the caption of an image. The a-contrario model for this is discussed in section 2.3, and we evaluate this detection with the experiment described in section 3.1.
- The detection of a region in the image that have different demosaicing artefacts than the other, suggesting that the image is fake. The a-contrario model for this is discussed in section 2.4, and we evaluate this detection with the experiment described in section 3.2.

The principle of the method proposed by the article, and described in figure 1, is the following:

- We have as input an image, and we want to detect if some areas of the image are fake
- For the 12 different configuration, we forge the image, resulting in 12 different images of the same shape as the original one
- We divide the pixels of the original image in blocks of shape $2 \times 2$. Each block 'votes' for the configuration that approximates it the best, i.e. for the forged image for which the mean absolute difference between the four pixels (the reconstruction error) is the lowest. The idea behind this is that the reconstruction error should be the lowest for the true configuration, i.e. the one that was really used during the caption of the image.
- In case of equality, the block votes randomly for one of the configuration that achieves the minimum reconstruction error.
- Then, we decide that the image is fake if there is a region in the image that have a lot of votes that are different from what we observe elsewhere in the image (cf. section 2 for more details). The idea behind this is that for a real natural image, the majority of votes should vote for the real configuration that was used during the caption of the image, so there should be no region where the vote are significantly different. On the opposite, if a piece of image is inserted or displaced, it is likely to have either different pattern of a different demosaicing algorithm.

## 1.3 Existing literature

For a more complete overview of the existing literature, we invite you to directly refer to the original paper [1].

Some early work have been done to detect, given an image, the pixel whose colours are interpolated, using either the periodicity of the CFA grid or the variance of the pixel values where the interpolation is done [5, 11]. Then an important paper have been published in 2010 [6], whose work is extended by the paper [1] we are reviewing. This paper used a double demosaicing with the four CFA patterns to detect which one have been used for the image. However, this paper only uses bilinear demosaicing, contrary to [1]. Then, [9] applied several demosaicing algorithms using a pattern already detected to detect which one was originally used.

One of the greatest difficulties faced by these different methods is the robustness to compression and in particular JPEG compression [13], which is still extremely used today. As explained in the article we are reviewing, very few forgery detection methods based on demosaicing are robust to compression. According to the benchmark proposed in the article, only the 4Point method [2] is robust to compression. It is a method proposed the same year by the same authors, which uses a simple neural network to learn the pattern used during the demosaicing of an image. This method, which is very efficient (according to the same benchmark, it obtains the best performance in all categories), is however very long to execute (about 100 times longer than the approach discussed in this review). This is the reason why it is interesting to explore other numerical methods, lighter, whose performances and robustness, although inferior, are close to those of the 4Point method.

### 1.4 Scope of this review

As said previously, our review mainly focus on the reliability of the detection performed, especially in terms of false detections. Thus, we will not analyze in detail the benchmark comparing the performance of this method with other state-of-the-art methods. Thus, the main issues we will address in this review concern the automatic detection of fake images using the a contrario framework:

- What are the assumptions behind the a contrario model, and are they verified?
- Does the number of false alarms (NFA) stated in the article really give confidence in the detection results?
- What are the cases where the method fails, and those where it remains effective?

## 2 THE A CONTRARIO FRAMEWORK

### 2.1 Introduction

In this section, we will present the a contrario model used by the paper for detecting image forgery. Our discussion will be based on the definition of *a contrario detection* and *number of false alarm* as presented in [10].

As explained in the paper, the goal of the a contrario framework is to automate the process of deciding whether an image is fake or not, without the need for an expert. Indeed, the fake images that are detected using demosaicing analysis are often not detectable at all with the naked eye. For example, let's take the fake image in figure 2, and let's refer to the Gestalt theory [10, 14] to decide what is detectable and what is not. In this image, there is no color discontinuity, no line break or change in the perspective, no shape similarity, nothing that could suggest us that some word have been removed. This difficulty to assess if a image is fake or not makes it crucial to have an automatic pipeline, that does not requires human control and provide a reliable metric to quantify the likelihood of an image to be fake.

### 2.2 The a contrario model

*A contrario detection* is a detection method whose purpose is to automate the decision that, based on observations, decides whether an event has taken place or not, and to quantify the degree of certainty with which these observations allow us to make this decision. This quantification is made with a metric called *number of false alarm* (NFA).

For our task, the a contrario introduced by the authors is the following (cf. section 1.2 and figure 1 for a more general presentation of the method):

- The image is forged using the 12 configurations (this number of 12 configuration is not the same in the original paper, but the idea remains the same), and we compute the mean reconstruction error for each pixel among the 3 colour channels
- Every $2 \times 2$ pixel block votes for the configuration for which the reconstruction error is the lowest
- In case of equality, the block votes for randomly for one of the configuration that achieved the minimum. This technical precision is not clearly stated in the original paper, however it is needed for the hypotheses of the a contrario model to be true

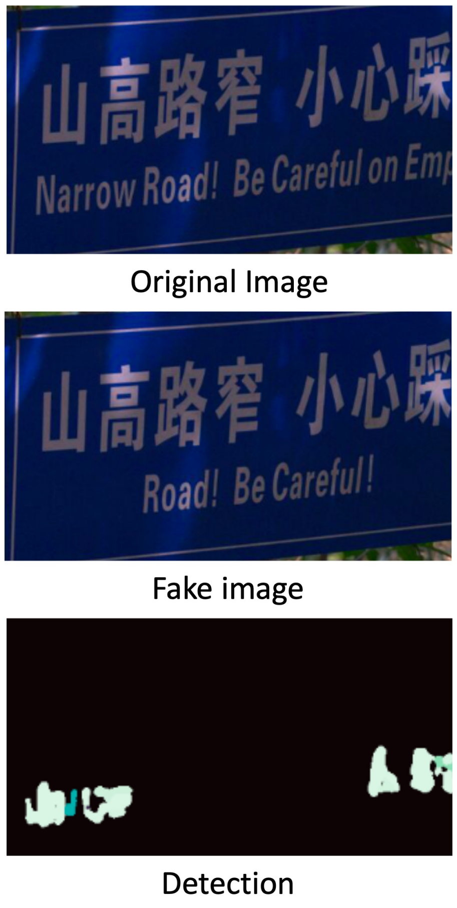

**Figure 2: Figure adapted from a figure in [2]: An example of demosaicing-based detection of a fake image.** Here, the forgery is not visible at all with the naked eye. The detection is made using the 4Point method, which is a bit different from the one discussed here, yet the general principle remains the same.

### 2.3 Detecting the global configuration of the image

Thus, in the absence of any demosaicing artefacts, we make the hypothesis that each of the 12 configuration is equally likely. Thus, if we count the votes for each of the 12 configurations, we should roughly count 1/12 of the total votes for each configuration.

More precisely, let's quantify the probability that the configuration that gets the most votes gets at least $k$ votes. For this purpose, let's denote by $N_{cfg}$ the number of configuration (here, 12), $N_{block}$ the number of $2 \times 2$ blocks (roughly the number of pixels divided by 4), and $\text{Binom}_{sf}$ the survival function of the binomial distribution. Then, for each block, there is a probability of 1/12 to observe a given configuration. Thus, the probability that we observe at least $k$ times this given configuration is:

$$g(k) = \text{Binom}_{sf}(k, N_{block}, 1/N_{cfg})$$

Then, let's take an image and denote by $(X_i)_{1 \leq i \leq N_{cgf}}$ the number of votes for each configuration. If we take $f(i,k) = N_{cfg} \times g(k)$, using proposition 4 of section 2.5 in [10], we obtain that $f$ is a Number of False Alarm (NFA) with respect to the definition 2 section 2.5 of [10]. Thus, we will call $NFA_{global}(k)$ the following quantity, which can be interpreted as the probability that a image with no demosaicing artefacts lead to a maximal number of votes for a configuration of $k$ or more:

$$NFA_{global}(k) = N_{cfg} \times \text{Binom}_{sf}(k, N_{block}, 1/N_{cfg})$$

Thus, for a given image, if we observe $k$ votes for the majority vote, and if $NFA_{global}(k)$ is really low (typically, $10^{-6}$ or less), we can say that this image is very unlikely to be free of any demosaicing artefacts, and on the contrary the majority votes is very likely to represent the true configuration used during the caption of the image.

## 2.4 Detecting the configuration in small windows of the image

What we just did for the whole image, we can do for a subpart of the image, for example a window of size $W \times W$ blocks. Thus, for each window we have the detection of a configuration. It can happen that the configuration detected for one of the windows is not the same configuration as the one detected for the whole image.

More precisely, we will perform the following series of tests: for each window of size $W \times W$, we count the number of votes for each configuration different from the global configuration of the image. Then, we take the maximum number of votes among these different tests. As before, for each of these tests, the probability of observing a number of votes greater than or equal to $k$ is given by $\text{Binom}_{sf}(k, W^2, 1/N_{cfg})$. Given that there are $N_{win} \times (N_{cfg} - 1)$ tests, where $N_{win}$ is the number of windows, the NFA associated to the maximum number of votes is:

$$NFA_{fake}(k) = N_{win} \times (N_{cfg} - 1) \times \text{Binom}_{sf}(k, W^2, 1/N_{cfg})$$

Note here that it is not necessary for the windows to be disjointed. Thus, in the case of overlapping windows, each test is not really independent, and the previous formula can therefore be seen as an upper bound on the number of false alarms, and not an exact expression.

Thus, as noted in the original article, this NFA value 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 fake.

Moreover, we note here that our formula is different to the one of the original article: we used $(N_{cfg} - 1)$ instead of $N_{cfg}$ in the number of tests, because we considered that we do count the votes for the configuration that gets the majority in the full image.

## 2.5 Doing detection on algorithm, pattern, or diagonal

The same principle can be used, replacing votes for the best approaching configuration, by votes for the best algorithm, pattern, or diagonal. To do this, we take the votes of every $2 \times 2$ block, and give it to the algorithms involved in this configuration, or pattern, or diagonal.

The rest of the reasoning, as well as the NFA formulas presented in the previous sections, remain the same, replacing $N_{cfg}$ either by 3, 4, or 2 depending on whether it is for algorithm, pattern, or diagonal.

## 3 EXPERIMENTS TO TEST THE METHOD

In this section, we will discuss the validity of the hypothesis made on the demosaicing algorithms by the original paper. To do so, we first present two types of experiment we conducted, in order to verify the validity of the NFA computed in section 2, which are similar to the ones of the original paper:

- First, we conducted experiments on images that were forged *a priori* with various configuration of algorithm and pattern, using various JPEG compression level, to see if the method proposed by the author lead to a correct *a posteriori* detection of the configuration previously used.
- Second, we conducted experiments to evaluate the detection of fake images: we took a thousand of images, and for half of them we artificially forged a part of the image with another demosaicing configuration than the rest of the image. Then, we computed the true detection and false detection rates.

We also discuss why the NFA provided by the authors seem invalid with our experiments, and what should be done to obtain a robust and trustworthy fake detection method based on demosaicing analysis.

## 3.1 Experiments 1: detecting the configuration used during the caption of an image

To illustrate this experiment, we have prepared a notebook that can be executed as a standalone, and that highlights the important points of this section. This notebook is available here.

*3.1.1 Experimental protocol.* First, we wanted to evaluate the effectiveness of the double demosaicing technique proposed by the authors. To do so, we first used a dataset that was also used in the original paper, which is provided by [3]. This dataset contains 16 high-quality images, which have really low noise since their pixel values were smoothed with a $5 \times 5$ mean pooling. Given the $5 \times 5$ mean pooling of the images, this dataset is claimed to be free of any demosaicing traces by [1]. Then we did the following:

- For each of the 16 images of the dataset; for each of the 12 configuration (algorithm, pattern) plus an additional configuration where nothing is done; for each of the JPEG compression level 100, 95, 90 (level 100 means no compression); we forged the image. This aim at reproducing and controlling what would be the true configuration used during the caption of the images.

- Then, for each of those 702 images, we performed several detections with various NFA thresholds: $t = 1$, $t = 0.01$, $t = 10^{-8}$:
  - Detection of the event "no demosaicing is done": for this task, the detection happens when no configuration is detected with NFA smaller than $t$. For this task, there can only be true or false detection (there cannot be no detection).
  - Detection of the event "demosaicing was done with algorithm A" for A equal to `bilinear`, `malvar`, `menon`: for this task, the detection happens when the algorithm is correctly detected with NFA smaller than $t$; there is no detection if no algorithm is detected with NFA smaller than $t$, and there is a false detection if the wrong algorithm is detected with NFA smaller than $t$.
  - Detection of the patterns $\begin{smallmatrix} R & G \\ G & B \end{smallmatrix}$, $\begin{smallmatrix} B & G \\ G & R \end{smallmatrix}$, $\begin{smallmatrix} G & R \\ B & G \end{smallmatrix}$, $\begin{smallmatrix} G & B \\ R & G \end{smallmatrix}$, with similar definition as for the algorithm.
  - Detection of the diagonals up ("0" in the figures) and down ("1" in the figures), with similar definitions as for the algorithm.

*3.1.2 Results on the no_noise_images dataset.* The result of those simulations using dataset [3] are presented in figures 3, 4 and 5.

**Lecture of the figures:** For each of the groups (e.g. "BGGR" or "0"), the left column is without JPEG compression, the middle one is with JPEG95, and the right one with JPEG90. Each column have been normalized to 1, and represent the proportion of true, false, and no detection for this setting (e.g. for group "0", true, false and no detection of diagonal "up" during the detection of the diagonal).

*3.1.3 Interpretation.* We can make the following observations from figures 3, 4 and 5.

First, let's analyse the six groups at the right, i.e. the ones that corresponds to the detection of the pattern or of the diagonal.

- As one could expect, we always observe that the performance decrease (i.e. the number of false detection or no detection improves) as the JPEG compression quality decreases. This is visible with red and blue parts of the bars getting bigger when going to the right in a group of three bars. However, as mentioned in the original paper, we observe that the performance remains quite high, and for the detection of the pattern and the diagonal, there is at least 60% of true detection, even for JPEG90 compression.
- Moreover, we observe that when the NFA threshold decreases (i.e. going from figure 3 to 4 to 5), more and more false detection become no detection, and more and more true detection become no detection. This is exactly what we expect from a NFA: the smaller the threshold is, the more confident we want to be on our detections, which removes some false detection, at the cost of loosing some true ones.
- More precisely, the number of false detection is coherent with the theoretical values of the NFA. Indeed, as shown in table 1, the proportion of false detection is equal or inferior to the NFA. For example, with NFA threshold set at $10^{-2}$, we observe precisely 1% of false alarm. And with 1152 tests, we observe no false detection with threshold $10^{-8}$, as expected.

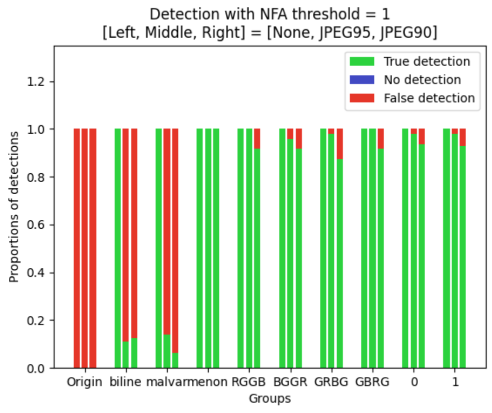

**Figure 3: Detection of the configuration used during the capture of the images, NFA = 1, using dataset no_noise_images [3]**

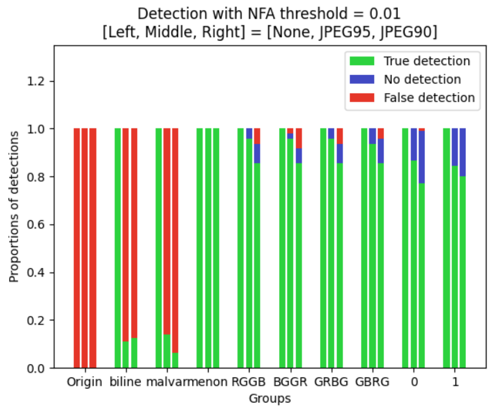

**Figure 4: Detection of the configuration used during the capture of the images, NFA = 0.01, using dataset no_noise_images [3]**

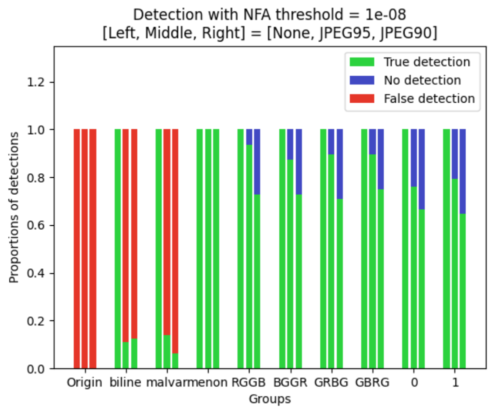

**Figure 5: Detection of the configuration used during the capture of the images, NFA = 10-8, using dataset no_noise_images [3]**

Thus, the theory developed in section 2 seems coherent with the experiments we conducted for pattern and diagonal detection (the six groups at the right).

| NFA threshold | False detection (1152 tests) | Proportion |
|---|---|---|
| $10^0$ | 37 | $3.2 \times 10^{-2}$ |
| $10^{-2}$ | 12 | $1.04 \times 10^{-2}$ |
| $10^{-8}$ | 0 | 0 |

**Table 1: Evolution of the proportion of false detection when decreasing the NFA threshold for the detection.** The 1152 tests correspond only to the detection of the pattern or the diagonal (6 right columns on figures 3, 4 and 5). These experiments confirm that the NFA theoretically computed in section 2 are indeed upper bounds on the expectations of the number of false alarms. This table is adapted from a more complete table available in the `output` directory of the repository here.

However, what we observe with the four groups at the left (i.e. detection of the absence of demosaicing or detection of the algorithm) is totally incoherent with the theory developed in section 2:

- The number of false detection is much greater than the NFA: there is a total of 624 tests for those four columns, and a least 276 false detection (about 44%) for every NFA threshold, even $10^{-8}$!
- For every image that was not demosaiced, no matter the JPEG compression or the NFA threshold, there is always a false detection of a demosaicing configuration.

There are several ways to interpret this:

- First, to explain that a demosaicing is always detected, even when nothing is done, we can make the hypothesis that the images of the dataset [3] are not free of any demosaicing traces. However, this hypothesis does not stand, for several reasons:
  - First, those images are obtained after a $5 \times 5$ mean pooling. Given that the CFA grid is 2-periodic, every pixel value of the final images comes from many true and interpolated pixel values, making it really unlikely that demosaicing artefacts remain after the pooling.
  - Second, we have conducted the same experiments with images that are for sure free of any demosaicing artefacts because they are computer-generated (cf. section 3.1.4), so no CFA grid was never preset during the creation of the image. And with those computer generated images, the observed the same rate of false detection.
- Second, we can make the hypothesis that some demosaicing algorithms are intrinsically better than others in some situation, leading to their false detection. This hypothesis contradicts the a-contrario model, because this model makes the hypothesis that in the absence of demosaicing artefacts, every configuration is equally likely to get the vote of a $2 \times 2$ block. This could explain all our observation of abnormally high false detection rate:
  - In the absence of any demosaicing artefacts (first group at the left), given that some algorithms are intrinsically better than others, many blocks of $2 \times 2$ vote for them, leading to their false detection with very low NFA value.
  - For any of the three groups `bilinear`, `malvar`, `menon`, there is 100% of true detection without JPEG compression, but the problem of false detection appears with the JPEG compression. This can be explained with the hypothesis we just made: when the image is compressed, the demosaicing artefacts are less visible, so the true algorithm is less favored for the votes. Thus, a lot of $2 \times 2$ blocks vote for the most performant algorithm rather than one that left some hardly-detectable demosaicing artefacts.
  - With our observation, it is likely that algorithm `menon` is more performant than the other, because it is not really concerned by this apparition of false detection when compression increases.

*3.1.4 Results on the dall_e_images dataset.* As said above, to refute the hypothesis that the images of the no_noise_dataset [3] contain some demosaicing artefact, we performed the same experience on images that are truly free of any demosaicing artefacts since they are computer-generated. To do so, we generated 18 images using DALL-E software [12]. For more simplicity, the results are not presented here, however they are available in the `output` directory of the repository here.

The results of those experiments are clear: we observe the same type of result as with the no_noise_images dataset, refuting this hypothesis.

*3.1.5 Hardware and computation time.* For this experiment, we used an Intel Xeon W-1290P 3.70GHz. To process the 16 images of dataset [3] using a single thread, this experiment took about 8min, i.e. about 30s per image for all the tests (the images have shape $704 \times 469$).

*3.1.6 Conclusion on experiment 1.* The method described by the original article seems to work really well for the detection of the demosaicing patter, even when the images are compressed. For those detections, the numbers of false detection are coherent with the NFA we computed theoretically, which are thus empirically validated.

However, for the detection of the demosaicing algorithm or the detection of the absence of demosaicing, the method lead to a really high proportion of false detection, making it unreliable. To explain those false detections, we retain our second hypothesis, that explains all our observations : **some demosaicing algorithms are intrinsically better than other,** which contradicts the uniform distribution of the a contrario model, resulting in much higher false detection that what we expected based on our NFA computations. This issue has no consequence on the detection of the pattern and of the diagonal, but misleads both the detection of the absence of demosaicing, and the detection of a given demosaicing algorithms. Obviously, this will have consequences on the performance of the forgery detection pipeline.

## 3.2 Experiment 2: detecting if an image is fake or not

To illustrate this experiment, we have prepared a notebook that can be executed as a standalone, and that highlights the important points of this section. This notebook is available here.

*3.2.1 Experimental protocol.* After evaluating the performance of demosaicing detection, we conducted a second group of experiments with the objective of evaluating the detection of fake images based on demosaicing detection.

Of course, the problems described above that affect demosaicing detection, especially in the absence of demosaicing artifacts, are still present and have repercussions on the detection of fakes.

We conducted the following experiments:

- We have generated 3 times 1000 random fake images, one for every compression factor equals to 100, 95, or 90, using the following method:
  - We randomly sample an integer between 0 and 17, and take the image with the corresponding index in the dataset
  - We randomly sample an integer between 0 and 11, and forge the full image with the corresponding (algorithm, pattern) configuration
  - We randomly sample the position of a 200×200 window where we will imitate a fake
  - We randomly sample an integer between 0 and 2, and with this integer we select one of the 3 pattern that was not used for the full image. Then, we forge the windows with this pattern, and with the same algorithm as the one used for the full image. The fact that we only change the pattern comes from the difficulty to detect the algorithm, as discussed for our first experiment in section 3.1.
  - We apply the JPEG compression at the end if needed
- We have generated 3 times 500 random non-fake image, one for every compression factor equals to 100, 95, or 90, using the following method. With this method, the images are not fake, however they are forged to imitate some natural demosaicing artefacts that would be left by a camera.
  - We randomly sample an integer between 0 and 17, and take the image with the corresponding index in the dataset
  - We randomly sample an integer between 0 and 11, and forge the full image with the corresponding (algorithm, pattern) configuration
  - We apply the JPEG compression at the end if needed

Then for every JPEG compression level, we deploy the fake image detection pipeline presented in section 2, and count the true detection, false detection, and no detection. More precisely:

- For fake image, there is either a true detection or a no detection
- For an non-fake image, there is either a false detection or a true detection

*3.2.2 Results of the simulation.* The results of those experiments are presented below in figures 6, 7 and 8.

**Lecture of the figures:** For each of the groups `fake` (called `forge` here due to an inconsistency in the naming) and `non-fake` (called `unforged` here due to an inconsistency in the naming), the left column is without JPEG compression, the middle one is with JPEG95, and the right one with JPEG90. Each column have been normalized to 1, and represents the proportion of true, false, and no detection for this setting.

*3.2.3 Interpretation of the results.* We can make the following observations from figures 6, 7 and 8:

- The method developed by the original article seems to be quite effective in detecting fake images (in more than 80% of cases, even 90% for uncompressed images). Moreover, in the absence of compression, the method does not make any false detection on the non-fake images.
- For the fake images, we can see that the number of non-detections is greater in the presence of compression than in its absence, as we would expect. However, it is surprising that this number is higher in JPEG95 than in JPEG90.
- The number of false detections of non-fake images is abnormally high as soon as the images are compressed, even in good quality. This is quite surprising, especially since the detection here only concerns the demosaicing pattern, which was however relatively well detected in JPEG95 and JPEG90 (cf. section 3.1).

One hypothesis that could explain this abnormally high number of false detections is that the colors of some textures are better interpolated with some patterns than with others. This would lead some regions of the image to be more likely to vote for patterns that do not have the majority vote for the whole image. However, we were not able to conduct an experiment to test this hypothesis, and we leave this to future work.

On the other hand, it is true that our experiments focus on images that are not genuine fakes generated in order to distort an image. It is therefore possible that this abnormally high number of false detections differs on real images. However, our fake image generation process is a priori easier to detect, because we directly and artificially introduced an inconsistency in the demosaicing artefacts. We therefore leave to future work the task of evaluating this method on a dataset of genuine fake images.

*3.2.4 Hardware and computation time.* For this experiment, we used an Intel Xeon W-1290P 3.70GHz. The generation of $3 \times 1000$ fake images with a single thread took about 227s (3min47, about 75ms per image). The generation of $3 \times 1000$ non-fake images with a single thread took about 213s (3min33, about 71ms per image). Then, still with a single process, it took about 5058s to do the detection on the 6000 images, i.e. about 843ms per image (the images have shape $704 \times 469$).

*3.2.5 Conclusion on experiment 2.* This experiment showed that the forgery detection method proposed by the paper is relatively efficient to detect fake images, even compressed. However, the abnormally high number of false detections makes the detections from this method unreliable. Our simulations therefore suggest that this method needs to be reworked before it can be qualified for use on real data.

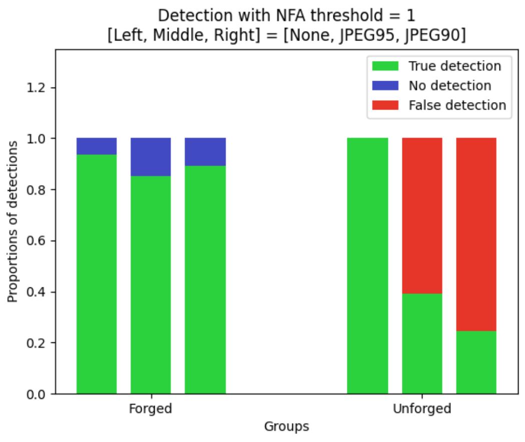

**Figure 6: Detecting if there is a fake area in the image, NFA = 1, using dataset no_noise_images [3]**

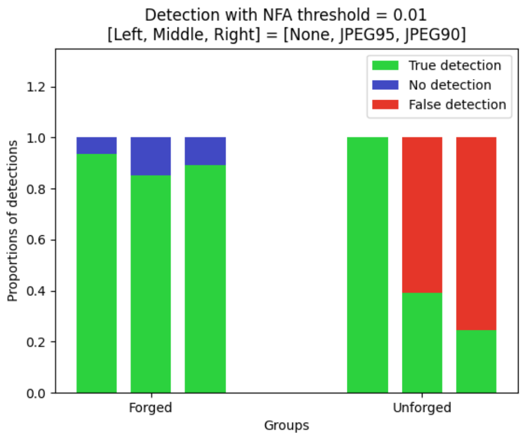

**Figure 7: Detecting if there is a fake area in the image, NFA = 1, using dataset no_noise_images [3]**

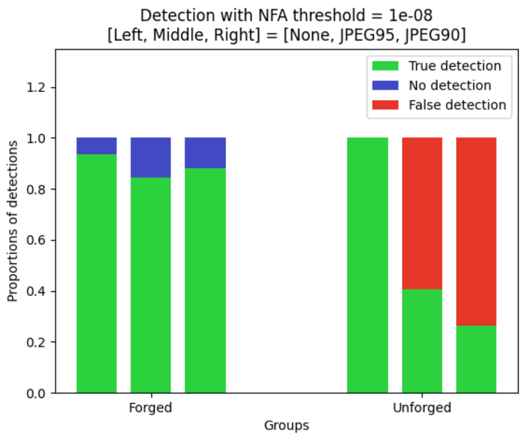

**Figure 8: Detecting if there is a fake area in the image, NFA = 1, using dataset no_noise_images [3]**

## 4 CONCLUSION

In conclusion, our experiments have shown that the method described in the article under review is promising, but that some aspects need to be improved to allow reliable and industrially usable detections.

First, the first experiment of our study focused on the detection of the demosaicing algorithm and pattern used during the caption of images. We observed that the proposed method worked very well for the detection of the demosaicing pattern, even in the presence of compression; and that it also worked well for the detection of the demosaicing algorithm, in the absence of compression. However, this experiment also showed that the better performances of some demosaicing algorithms can lead to an abnormally high number of false demosaicing detections with respect to the theoretically calculated NFA, when the images are compressed. Thus, in order to obtain a reliable method, it is necessary to validate the hypotheses of the a contrario model, which is not done in the article. This validation could take the form of an evaluation on a large set of images free of demosaicing artefacts (e.g. computer generated images or images obtained after a mean pooling), and for which the different algorithms must have similar performances.

Then, the second experiment of our study focused on the detection of fake images, i.e. images for which a distortion has been artificially added in the demosaicing pattern. We observed good performance in detecting fake images, even in the presence of compression, and no false detection in the absence of compression. However, the number of false detections in the absence of compression makes this method too unreliable until it is improved.

## REFERENCES

[1] Quentin Bammey, Rafael Grompone Von Gioi, and Jean-Michel Morel. 2022. Demosaicing to Detect Demosaicing and Image Forgeries. In *2022 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, Shanghai, China, 1–6. https://doi.org/10.1109/WIFS55849.2022.9975454

[2] Quentin Bammey, Rafael Grompone von Gioi, and Jean-Michel Morel. 2022. Forgery Detection by Internal Positional Learning of Demosaicing Traces. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. IEEE, Waikoloa, HI, USA, 1019–1029. https://doi.org/10.1109/WACV51458.2022.00109

[3] Miguel Colom. 2023. Noise free test images. https://mcolom.perso.math.cnrs.fr/pages/no_noise_images/

[4] Jérémie Dentan. 2023. Dataset dall-e-images. https://gist.github.com/DentanJeremie/21bfd925c5234afd15d854135b569bec

[5] Pasquale Ferrara, Tiziano Bianchi, Alessia De Rosa, and Alessandro Piva. 2012. Image Forgery Localization via Fine-Grained Analysis of CFA Artifacts. *IEEE Transactions on Information Forensics and Security* 7, 5 (Oct. 2012), 1566–1577. https://doi.org/10.1109/TIFS.2012.2202227 Conference Name: IEEE Transactions on Information Forensics and Security.

[6] Matthias Kirchner and Jessica Fridrich. 2010. On detection of median filtering in digital images, Nasir D. Memon, Jana Dittmann, Adnan M. Alattar, and Edward J. Delp III (Eds.). San Jose, California, 754110. https://doi.org/10.1117/12.839100

[7] Henrique Malvar, Li-wei He, and Ross Cutler. 2004. *High-quality linear interpolation for demosaicing of Bayer-patterned color images*. Vol. 3. https://doi.org/10.1109/ICASSP.2004.1326587 Journal Abbreviation: Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on Pages: 485 Publication Title: Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on.

[8] Daniele Menon, Stefano Andriani, and Giancarlo Calvagno. 2007. Demosaicing With Directional Filtering and a posteriori Decision. *IEEE Transactions on Image Processing* 16, 1 (Jan. 2007), 132–141. https://doi.org/10.1109/TIP.2006.884928

[9] Simone Milani, Paolo Bestagini, Marco Tagliasacchi, and Stefano Tubaro. 2014. Demosaicing strategy identification via eigenalgorithms. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2659–2663. https://doi.org/10.1109/ICASSP.2014.6854082 ISSN: 2379-190X.

[10] Jean-Michel Morel and Rafael Grompone von Gioi. 2021. *Detection Theory and Industrial Applications*. http://dev.ipol.im/~qbammey/anne/detection_theory_lecture_notes.pdf

[11] A.C. Popescu and H. Farid. 2005. Exposing digital forgeries in color filter array interpolated images. *IEEE Transactions on Signal Processing* 53, 10 (Oct. 2005), 3948–3959. https://doi.org/10.1109/TSP.2005.855406 Conference Name: IEEE Transactions on Signal Processing.

[12] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical Text-Conditional Image Generation with CLIP Latents. https://doi.org/10.48550/arXiv.2204.06125 arXiv:2204.06125 [cs].

[13] A. Skodras, C. Christopoulos, and T. Ebrahimi. 2001. The JPEG 2000 still image compression standard. *IEEE Signal Processing Magazine* 18, 5 (Sept. 2001), 36–58. https://doi.org/10.1109/79.952804 Conference Name: IEEE Signal Processing Magazine.

[14] Christian von Ehrenfels. 1890. *Über "Gestaltqualitäten"*. Reisland. https://phaenomenologica.de/wp-content/uploads/2019/02/CvEhrenfels_Gestaltqualitaeten.pdf

Received 01 January 2023