Image Forensics from Chroma Subsampling of High-Quality JPEG Images

Benedikt Lorch
benedikt.lorch@fau.de
IT Security Infrastructures Lab
Friedrich-Alexander University Erlangen-Nürnberg (FAU)
Erlangen, Germany

Christian Riess
christian.riess@fau.de
IT Security Infrastructures Lab
Friedrich-Alexander University Erlangen-Nürnberg (FAU)
Erlangen, Germany

ABSTRACT

The JPEG compression format provides a rich source of forensic traces that include quantization artifacts, fingerprints of the container format, and numerical particularities of JPEG compressors. Such a diverse set of cues serves as the basis for a forensic examiner to determine origin and authenticity of an image.

In this work, we present a novel artifact that can be used to fingerprint the JPEG compression library. The artifact arises from chroma subsampling in one of the most popular JPEG implementations. Due to integer rounding, every second column of the compressed chroma channel appears on average slightly brighter than its neighboring columns, which is why we call the artifact a "chroma wrinkle". We theoretically derive the chroma wrinkle footprint in DCT domain, and use this footprint for detecting chroma wrinkles. The artifact is detected with more than 90% accuracy on images of JPEG quality 75 and above. Our experiments indicate that the artifact can also be used for manipulation localization, and that it is robust to several global postprocessing operations.

CCS CONCEPTS

• **Theory of computation → Data compression**; • **Computing methodologies → Image processing**; **Image compression**; • **Information systems →** *Multimedia content creation.*

KEYWORDS

Image forensics, JPEG compression, library fingerprinting, forgery localization.

ACM Reference Format:
Benedikt Lorch and Christian Riess. 2019. Image Forensics from Chroma Subsampling of High-Quality JPEG Images. In *ACM Information Hiding and Multimedia Security Workshop (IH&MMSec '19), July 3–5, 2019, TROYES, France.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3335203.3335722

1 INTRODUCTION

The goal of digital image forensics is to develop tools for validating origin and authenticity of digital images. These tools are typically intended for users in law enforcement, journalism, and private corporations that use images as part of their business model. These users operate on images with large variations in content and quality, ranging from high-quality photographic source data in court to downsampled and strongly compressed content in internet forums. Such diverse requirements from these users can not be satisfied by a single forensic tool. Thus, researchers developed a set of methods with varying specializations. An overview on existing methods can be found in textbooks on image forensics, e.g., [10, 21].

Among these works, a family of methods focuses on the analysis of JPEG compression artifacts, because JPEG is probably the most widely used storage format for images from consumer devices. It achieves much of its storage efficiency by removing high-frequency image content. This change of the image content is oftentimes visually imperceptible, but it can nevertheless provide valuable cues for forensic examiners.

Compression artifacts can be used for example to estimate the compression history of an image, and in particular to distinguish image areas that are compressed once from areas that are compressed twice. To this end, many works distinguish whether the JPEG blocks of the first and second compression are assumed to be aligned [17, 20] or not aligned [3]. Both tasks can also be combined into one approach, which either involves an exhaustive search over the possible block alignments [4, 9, 24] or a statistical model with a capacity that is sufficiently large to model the appearance of all possible shifts [2]. It is even possible to analyze more than two subsequent compression steps, if adequate assumptions are met. Block convergence can be used to estimate how many times an image has undergone JPEG compression [7, 15]. Pasquini *et al.* proposed a method to identify up to three aligned JPEG compressions and quality factors using Benford-Fourier coefficients [18].

JPEG artifacts can also be used specifically for manipulation detection. For example, retouching or splicing may lead to inconsistent blocking artifacts [16, 23].

Closely related is also the extensive literature on steganalytic methods for images, which search for very subtle signal embeddings in JPEG images. For example, Fridrich and Kodovský proposed the "rich models" [11] as a statistical descriptor for classification of unnatural image content, and extended this approach to the DCT domain of JPEG images [14]. Similar features capturing intrablock and interblock correlations between DCT coefficients were shown to be effective for steganalysis and image forensics tasks [8, 13].

Recently, researchers started to investigate differences in implementations of JPEG libraries. The goal of these works is to extract a library fingerprint. Such a cue provides insights on the software platform and configuration of the system, which helps associating

IH&MMSec '19, July 3–5, 2019, TROYES, France

ACM ISBN 978-1-4503-6821-6/19/06…$15.00
https://doi.org/10.1145/3335203.3335722

an image to a source. Additionally, such a cue can indicate manipulations if it can be recovered on a sufficiently small local image region. For example, Agarwal and Farid characterize different JPEG implementations by their integer rounding operators during quantization [1]. The choice between floor and ceiling operators manifests itself as a single darker or brighter pixel in each 8×8 pixel block, which they call "dimples". Bonettini *et al*. recompress an input image with a known compressor in order to characterize differences in JPEG implementations [5]. This descriptor differentiates between JPEG images compressed with Photoshop and the Python Imaging Library (PIL) even if the same quantization matrix is used.

In this work, we report about a new artifact from the JPEG library implementation `libjpeg`. The artifact occurs as a high-frequency periodic pattern in chroma subsampling, and hence we call this artifact *chroma wrinkles*. It can best be observed in high-quality JPEG images, where least of the high-frequency content is cut off. We propose a straightforward detector that correlates a template of the artifact with the coefficients of the discrete cosine transform (DCT) of the image. We evaluate the detector's robustness to typical postprocessing operations including double compression. The presence or absence of the artifact can provide a hint which software library was used in the last compression step. Specific combinations of libraries even allow to deduce information about earlier compression steps. As such, chroma wrinkles can be a useful tool to distinguish images from different sources or processing histories. Additionally, local inconsistencies in the chroma wrinkles can indicate image manipulations. We demonstrate forgery localization for the case of image inpainting and two cases of image splicing when either host or donor, or when both contain the artifact.

The paper is organized as follows. In Sec. 2, we provide background information on differences in JPEG library implementations. In Sec. 3, we present the origin of the chroma wrinkles, and a detection algorithm. In Sec. 4, we evaluate the detector for a range of different JPEG qualities, study the artifact under various postprocessing operations, and investigate its prevalence in images from mobile devices. The work is concluded in Sec. 5.

2 CHROMA SUBSAMPLING IN LIBJPEG

Many software packages use the JPEG implementation in `libjpeg` or one of its forks for processing JPEG images. Notable forks of `libjpeg` are `libjpeg-turbo` and `mozjpeg`, which optimize on computational efficiency and file size, respectively.

All these implementations offer to subsample the chroma components for higher compression rates. Downsampling is oftentimes done by a factor of 2, where 4:2:2 denotes horizontal downsampling, and 4:2:0 denotes horizontal and vertical downsampling.

JPEG implementations differ in the exact realization of the downsampler and its associated upsampler. We refer to the classical scaling as *simple scaling*. Here, neighboring pixels are averaged prior to the DCT on 8×8 pixels. Upsampling is done by replicating each pixel after the inverse DCT. A variant of this upsampler linearly interpolates chrominance pixels weighted by proximity [7]. This approach is called *fancy upsampling*.

From `libjpeg v7` on, *DCT scaling* became the default scaling operation. Here, a DCT is performed on the full 16×16 macro-block, and only the 8×8 DCT coefficients corresponding to the lowest frequencies are stored. Upsampling is performed by zero-padding of the missing DCT coefficients prior to the inverse DCT. Interestingly, `libjpeg-turbo` did not adopt this change. In particular, the default downsampler uses *simple scaling* and the default upsampler uses *fancy upsampling*.

```
1  METHODDEF(void) h2v2_downsample (...)
2  {
3     ...
4     inptr0 = input_data[inrow];
5     inptr1 = input_data[inrow+1];
6     bias = 1;
7     /* bias = 1,2,1,2,... for successive samples */
8     for (outcol = 0; outcol < output_cols; outcol++) {
9        *outptr++ = (JSAMPLE) (
10           (GETJSAMPLE(*inptr0)
11         + GETJSAMPLE(inptr0[1])
12         + GETJSAMPLE(*inptr1)
13         + GETJSAMPLE(inptr1[1])
14         + bias) >> 2);
15       bias ^= 3;      /* 1=>2, 2=>1 */
16       inptr0 += 2; inptr1 += 2;
17    }
18    inrow += 2;
19    outrow++;
20    ...
21 }
```

Listing 1: *Simple scaling* is implemented in `jcsample.c`. The bias calculation avoids a shift of intensity values, but introduces a periodic pattern.

In this work, we demonstrate that *simple scaling* leaves behind a detectable forensic trace. This trace allows to distinguish images that are compressed with *simple scaling* from *DCT scaling*. Because *simple scaling* is the default option in `libjpeg-turbo`, the artifact is introduced by a number of software packages and operating systems that use this library, for example Android, Ubuntu Linux, and Debian.

3 ORIGIN AND DETECTION OF CHROMA WRINKLES

In this section, we first show the origin of the chroma wrinkles artifact in *simple scaling*. Then, we show its footprint in the DCT coefficients and propose for detection a template-matching algorithm in DCT domain.

3.1 Artifact origin

In `libjpeg` and its variants, *simple scaling* is implemented in the methods `h2v2_downsample` and `h2v1_downsample` for 4:2:0 and 4:2:2 subsampling, respectively. Chroma wrinkles are introduced in both variants, but due to space constraints, we only show the case for 4:2:0 subsampling. 4:2:0 *simple subsampling* averages 2×2 pixels by calculating the sum and performing a division by 4 via two bit shifts. This bit-shifting operation always rounds to the next lower integer. To counter a potential loss of intensity, a scalar bias is added. The bias toggles between the values one and two in horizontal direction, which normalizes the intensity range, but introduces a periodic pattern. Listing 1 shows the location in the code of `libjpeg` for 4:2:0 subsampling where the artifact is introduced. The bias is added in line 14 of the listing, and toggled in line 15.

3.2 Artifact footprint in DCT domain

Suppose a 1-D signal $f(x) \in \mathbb{Z}^8$ that is a sum of the image signal $s(x)$ and chroma wrinkles $w(x)$, i.e.,

$$f(x) = s(x) + w(x) \quad . \tag{1}$$

The linearity of the DCT yields the same relation in DCT domain,

$$\mathrm{DCT}(f(x)) = \mathrm{DCT}(s(x)) + \mathrm{DCT}(w(x)) \quad , \tag{2}$$

which allows to isolate the DCT footprint of the chroma wrinkles $\mathrm{DCT}(w(x))$. Thus, we seek the DCT of $w(x)$, which is

$$\mathrm{DCT}(w(x)) = \alpha(u) \sum_{x=0}^{N-1} w(x) \cos\left(\frac{\pi(2x+1)u}{2N}\right) \quad , \tag{3}$$

where $N = 8$, $0 \le u \le 7$, $\alpha(u) = \sqrt{1/N}$ for $u = 0$ and $\alpha(u) = \sqrt{2/N}$ otherwise, and $w(x) = [1, 2, 1, 2, 1, 2, 1, 2]$ as illustrated in the previous subsection.

We rearrange the terms into pixels at odd and even positions,

$$\begin{aligned}\mathrm{DCT}(w(x)) = \alpha(u)\Big(&\sum_{x=0,2,4,6} w(x) \cos\left(\frac{\pi(2x+1)u}{2N}\right) + \\ &\sum_{x=1,3,5,7} w(x) \cos\left(\frac{\pi(2x+1)u}{2N}\right)\Big) \quad .\end{aligned} \tag{4}$$

Since $w(0) = w(2) = w(4) = w(6)$ and $w(1) = w(3) = w(5) = w(7)$, $w(x)$ can be moved in front of the sums,

$$\begin{aligned}\mathrm{DCT}(w(x)) = \alpha(u)\Big(&w(0)\big[\cos(\tfrac{u\cdot\pi}{16}) + \cos(\tfrac{5u\cdot\pi}{16}) + \\ &\cos(\tfrac{9u\cdot\pi}{16}) + \cos(\tfrac{13u\cdot\pi}{16})\big] + \\ &w(1)\big[\cos(\tfrac{3u\cdot\pi}{16}) + \cos(\tfrac{7u\cdot\pi}{16}) + \\ &\cos(\tfrac{11u\cdot\pi}{16}) + \cos(\tfrac{15u\cdot\pi}{16})\big]\Big)\end{aligned} \tag{5}$$

For even values of u, the sums of cosines in the square brackets are always 0, and hence $\mathrm{DCT}(w(x))$ is 0. For odd values of u, non-zero values are obtained. Thus, the solution for the AC coefficients of Eqn. 5 is

$$\mathrm{DCT}(w(x)) = (w(0) - w(1)) \cdot [0.25, 0, 0.30, 0, 0.45, 0, 1.28] \tag{6}$$

$$= [-0.25, 0, -0.30, 0, -0.45, 0, -1.28] \quad . \tag{7}$$

Consequently, the periodic bias translates to a periodic DCT pattern.

Extension to 2-D: The same consideration holds for the case of a 2-D pixel grid. Let $f(x, y) \in \mathbb{Z}^{8\times8}$ denote a subsampled block of pixels that contains wrinkles $w(x, y) \in \mathbb{Z}^{8\times8}$. The wrinkles are again additive to the image content $s(x, y)$, i.e., $f(x, y) = s(x, y) + w(x, y)$. Chroma wrinkles only occur in horizontal direction, and wrinkles from different rows are identical. Thus, we seek to transform a signal of the form

$$w(x, y) = \begin{bmatrix} 1 & 2 & 1 & 2 & 1 & 2 & 1 & 2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 2 & 1 & 2 & 1 & 2 & 1 & 2 \end{bmatrix} \quad . \tag{8}$$

We separate the 2-D DCT into one 1-D DCT in horizontal direction, and another 1-D DCT in vertical direction. The horizontal transformation leads for each row to the result of Eqn. 7. Transforming these identical rows in vertical direction only preserves the vertical DC component. The resulting coefficients are

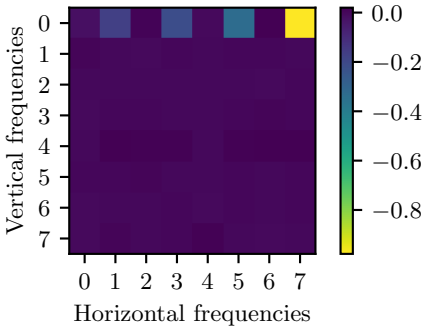

Figure 1: Average difference of DCT coefficients of the Cb channel of two images compressed with *simple scaling* and *DCT scaling*.

$$\mathrm{DCT}(w(x, y)) = \begin{bmatrix} * & -0.72 & 0 & -0.85 & 0 & -1.27 & 0 & -3.62 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad . \tag{9}$$

Thus, the periodic pattern of the bias in Sec. 3.1 translates to a periodic pattern in the first row of a DCT block. Figure 1 shows an experimental validation of this calculation. Here, we calculated the average difference of DCT coefficients of two images compressed with *simple scaling* and *DCT scaling*. The artifact occurs at horizontal frequencies for the odd coefficients. The artifact strength increases with frequency.

3.3 Detection

In practice, a chroma wrinkle detector has to determine the presence of the wrinkle $w(x, y)$ in presence of the actual image signal $s(x, y)$. We propose to test for the presence of chroma wrinkles directly in DCT domain. To do so, we create an 8×8 template from the DCT footprint of the chroma wrinkles in Eqn. 9.

To test an 8×8 image block $g(x, y)$ for chroma wrinkles, we first retrieve its chroma DCT coefficients from the compressed image. These coefficients are dequantized by multiplication with their respective quantization factor. The DC component is also removed, analogously to the template. The test itself is a zero-mean normalized cross-correlation on the AC coefficients, namely

$$\mathrm{zncc}(t(x, y), g(x, y)) = \frac{1}{63} \sum_{\substack{0\le x,y\le 7 \\ x\neq 0 \wedge y\neq 0}} \frac{\bar{t}(x, y) \cdot \bar{g}(x, y)}{\sigma_t \sigma_g} \quad , \tag{10}$$

where $t(x, y) = \mathrm{DCT}(w(x, y))$ denotes the template, $\bar{t}(x, y)$ and $\bar{g}(x, y)$ are mean-free versions of $t(x, y)$ and $g(x, y)$, and σ_t, σ_g denote the standard deviations over $t(x, y)$ and $g(x, y)$, respectively.

A high correlation indicates that the block contains the artifact, a low correlation indicates that the artifact is not present. The score for a single image can be computed by averaging the scores over all blocks. The decision threshold can either manually be set, or

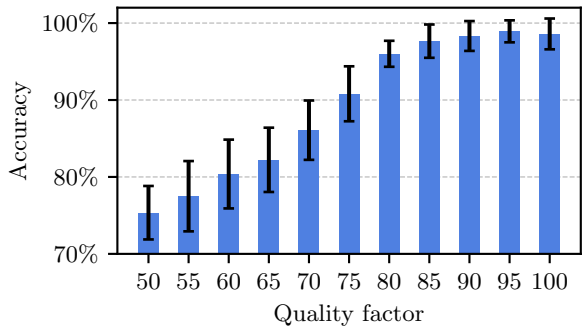

Figure 2: Accuracy to distinguish *simple scaling* and *DCT scaling* with a linear SVM on block correlations. The detection accuracy decreases with lower JPEG quality factors.

determined from the data. We propose to use a simple yet effective support vector machine (SVM) with linear decision boundary that is trained on the matching scores for the Cb and Cr channels. It is interesting to note that our decision boundary does not weight correlations in the Cb and Cr channel equally. For the majority of images, the correlation score in the Cb channel is higher than in the Cr channel, which we hypothesize is due to differences in in-camera processing for the red and blue channels. We also considered correlating only a subset of the AC coefficients but found the full 63 AC coefficients to give the best results.

4 EVALUATION

All evaluations are performed on 1,491 RAW images from the Dresden database [12]. We convert the images to JPEG using `dcraw` as well as `cjpeg` and `djpeg` from `libjpeg v9a`. Command line switches allow to generate images both with *DCT scaling* and with *simple scaling*. For *fancy upsampling* we use `djpeg` from `libjpeg-turbo v2.0.1`. In all experiments, care was taken that the type of subsampling is the only difference in the processing of the images. All experiments are performed with 4:2:0 chroma subsampling, but the same artifact is analogously part of 4:2:2 chroma subsampling.

4.1 Detection under varying JPEG quality

Chroma wrinkles are a high-frequency artifact, and as such susceptible to strong JPEG compression that discards high-frequency content. To investigate the robustness of the proposed artifact, we compress the RAW images with JPEG quality factors between 50 and 100, once using *DCT scaling* and once using *simple scaling* (all other compression parameters are fixed). We distinguish both scaling variants with SVMs, where one SVM is trained for each quality factor. The training is performed separately for each quality factor on 90% of the images using 10-fold cross-validation. Testing is performed on the remaining images.

Figure 2 shows the obtained accuracies per quality factor. For quality factors 100, 95, and 90 the accuracies range around 98%. With decreasing JPEG quality, the classifier's effectiveness drops. For JPEG quality 75, accuracy is still above 90%, for JPEG quality 50, accuracy is at about 75%.

Table 1: Effect of recompression on chroma wrinkles. Repeated *simple downsampling* with intermediate *DCT/fancy upsampling* enhances the artifact, while *simple/fancy upsampling* followed by *DCT downsampling* attenuates it. The remaining variations barely affect the artifact.

1st compression	Decomp.	2nd comp.	Cb correlation
simple	-	-	0.081 ± 0.032
	simple	simple	0.080 ± 0.031
		DCT	0.071 ± 0.028
	DCT	simple	0.130 ± 0.038
		DCT	0.080 ± 0.031
	fancy	simple	0.130 ± 0.029
		DCT	0.041 ± 0.019

4.2 Interplay of up- and downsampling

In many practical cases, it is expected that an image has been compressed at least twice. This compression likely occurs on different systems, e.g., inside a camera and on a social media site. Let us assume that the first compression uses *simple scaling*, which introduces chroma wrinkles. Then, we show that the wrinkle detecability is affected by the combination of the decompressor and the second compressor. To this end, we first compress the RAW images from the Dresden database using *simple scaling* with quality 100. Then, these images are decompressed using *simple upsampling*, *DCT upsampling*, or *fancy upsampling* and subsequently recompressed by using either *simple downsampling* or *DCT downsampling* with quality 100. Table 1 lists the correlation scores and standard deviations for these four combinations for the Cb channel (the Cr channel behaves analogously). The results show that identical methods for up- and downsampling do not affect the correlation. Correlations are slightly attenuated with *simple upsampling* followed by *DCT downsampling*. *Fancy upsampling* followed by DCT downsampling further decreases the correlations. The correlations are increased by *DCT upsampling* or *fancy upsampling* followed by *simple downsampling*, since this sequence enhances the wrinkles.

4.3 Resilience to global operations

We evaluate the persistence of chroma wrinkles under four common postprocessing operations, namely recompression, gamma adjustment, additive noise, and image scaling. Finally, we discuss (without evaluation) that image cropping does not impact detectability.

For each scenario, the 1,491 RAW images from the Dresden database are first converted to JPEG with *simple scaling* and quality 100. We then apply *DCT upsampling*, perform the post-processing operation in image space, and recompress the resulting image with *DCT scaling* and quality factor 100. As shown in the previous section, this compressor configuration does not impact the correlations, and therefore allows to study the respective post-processing operations.

4.3.1 Recompression. We study JPEG compression with lower quality factors. To this end, we used a quality factor between 80 and 100 for the second compression. At quality factor 100, the average correlation is 0.08 (cf. the fifth row of Tab. 1). Figure 3 (left) shows that correlations quickly approach zero below JPEG quality 95.

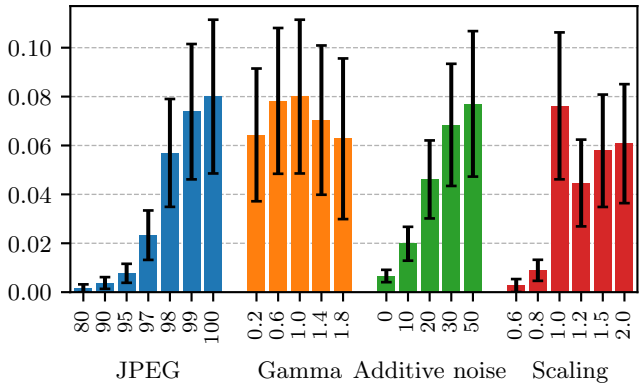

Figure 3: Correlation scores in Cb channel after applying one of four common post-processing operations: JPEG compression (quality factor), gamma correction (gamma), corruption by additive noise (SNR in dB), scaling (scale factor).

4.3.2 Gamma correction. The second plot in Fig. 3 shows that the chroma wrinkles are barely affected by gamma adjustments in the image within a wide range of gamma factors from 0.2 to 1.8.

4.3.3 Additive noise. The third plot in Fig. 3 shows that the chroma wrinkles are relatively resilient to additive zero-mean Gaussian noise. The correlations are clearly different from 0 for signal-to-noise ratios (SNR) of 10 dB and above.

4.3.4 Image resizing. The rightmost plot in Fig. 3 shows the impact of scaling on chroma wrinkles. We perform this evaluation by scaling the image by a fixed scaling factor using bilinear interpolation, store the image in JPEG format, uncompress it again, and scale it back to the original resolution. It turns out that upsampling barely affects the wrinkles, while downsampling is a quite destructive operation, which is in line with recent fundamental results on resampled signals [19].

4.3.5 Image cropping. Cropping an image does not change detectability of chroma wrinkles. If an odd number of columns is cropped from the left boundary of the image, the sign of the chroma wrinkle artifacts flip. Mathematically, $w(0)$ and $w(1)$ swap values in Eqn. 6, which also flips the sign of the correlations, but the magnitude of the correlation is unchanged.

4.4 Manipulation localization

Inconsistencies in the correlations of local image patches can be used to detect areas that have been edited. We demonstrate this with three examples in Fig. 4. In the top row of Fig. 4, the background image is compressed with *DCT scaling*, while the inserted airplane wreck is compressed with *simple scaling*. The ground truth mask for the inserted area is shown in the middle. We calculate chroma wrinkle correlations on patches of 128×128 pixels, which gives the Cb correlation map on the right. Although the correlations in the inserted regions have an overall magnitude of only about 0.1, they can be well distinguished from the background.

The middle row of Fig. 4 illustrates the case where both background and inserted image contain wrinkles, but the wrinkles in the inserted area are horizontally shifted by one pixel relative to the background. As a consequence, background and inserted area both show non-zero correlations. However, the signs of these correlations are different, which indicates the spliced region.

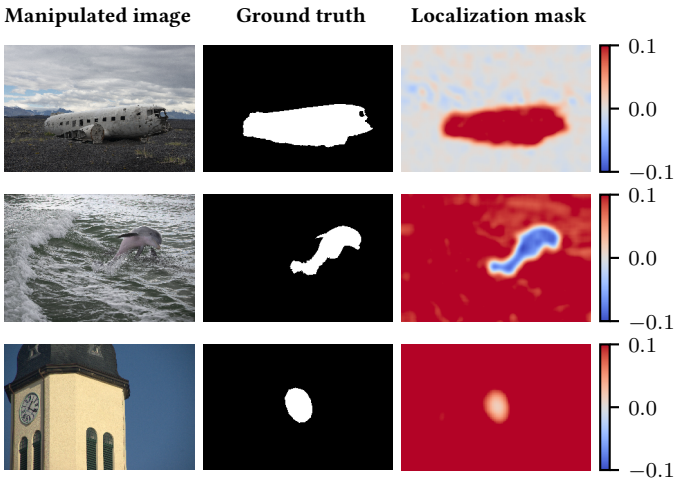

Figure 4: Three examples for image manipulation localization. Top: Image splicing localization when only the donor image contains chroma wrinkles. The background landscape is compressed with *DCT scaling* and the donor uses *simple scaling*. Middle: Image splicing localization where both host and donor contain the artifact, however, the artifacts in the inserted region are desynchronized with the host. Bottom: Inpaintaing localization when the original image contains the chroma wrinkles. The church tower image is compressed with *simple scaling*. The clock in the front was removed in Photoshop using content-aware fill.

The bottom row of Fig. 4 illustrates the case that the background image contains wrinkles that are removed by local editing. More specifically, a tower clock is removed using Photoshop's content-aware fill, and the result is saved again as JPEG image with quality factor 100. To detect this relatively small area, we additionally suppress the scene content by transforming the DCT coefficients back into pixel domain, and apply a 3×3 Wiener filter. The noise residual is transformed back into DCT coefficients for computing the correlations. As shown in the localization mask, the manipulated area exhibits a correlation of zero.

Similar cases like the examples above can be created from several library constellations. The widely used `libjpeg-turbo` introduces the artifact by default, while recent versions of `libjpeg` do not. High-quality compressions of `mozjpeg` and Adobe Photoshop do not subsample the chroma channels, and therefore also do not introduce chroma wrinkles in that case. Thus, image splicings between images from mixed libraries or editing within `libjpeg-turbo`-compressed images can exhibit such inconsistencies in chroma wrinkles. However, further research is required to detect local inconsistencies in chroma wrinkles "in the wild". In particular, scene content affects the analysis. We empirically found that noise residuals from Wiener filtering oftentimes improve but sometimes also weaken the detection.

4.5 Prevalence in mobile devices

We study 35 mobile devices from the VISION dataset [22]. All images with chroma subsampling steps other than 4:2:0 were excluded, which left us with 10,867 original, 7,478 WhatsApp, and 15,130 Facebook images. Since the estimator is sensitive to compression, we first estimate the quality factor from the image under analysis. To this end, we use the quantization tables of `libjpeg v9a` in quality steps of 5, and select the quantization table with the closest least squares distance to the chroma table of the input image. We use that quantization table to create a training set on the Dresden database, and train a linear SVM. Among the 35 devices, we found two of them to predominantly produce images with chroma wrinkles. These smartphones are the Huawei EVA-L09 and the LG D290 with 99.7% and 77.2% of the images, respectively. It is interesting to note that some of the LG D290's images seem to contain the artifacts and others do not.

Additionally, the WhatsApp and Facebook images in the VISION dataset were analyzed. As most WhatsApp images in the dataset use the same quantization table, we generate training images with that particular quantization table to more closely align training and test images. WhatsApp images with other quantization tables are excluded from the evaluation. We observe a surprisingly diverse mixture of present and absent chroma wrinkles in this dataset. In total, chroma wrinkles are detected on 71.8% of the WhatsApp and 63.6% of the Facebook images. We hypothesize that these results are due to different processing pipelines within these platforms, which has also been reported in dedicated works on social network provenance [6]. While this property makes it difficult to use chroma wrinkles for fingerprinting of social media data, it may be useful for detecting local inconsistencies for spliced images where background and inserted images stem from within the same network.

5 CONCLUSIONS

We present a new periodic artifact that arises from chroma subsampling in JPEG compression libraries. The artifact is a slight periodic chroma variation in horizontal direction, which is why we call it chroma wrinkle. We theoretically derive the footprint of chroma wrinkles in DCT domain. This footprint serves as a template for detecting the wrinkles, by correlating the template with dequantized DCT blocks of an image.

Chroma wrinkles can serve as forensic cue on the software library that was used to compress the image, most notably for compression with the widely used `libjpeg-turbo` in its default configuration. In principle, chroma wrinkles can also be used for manipulation localization. In future work, it may be interesting to investigate advanced denoising methods for detection of local manipulations.

ACKNOWLEDGMENTS

The authors would like to thank Shruti Agarwal and Hany Farid for their comments and suggestions.

This material is based on research sponsored by the Air Force Research Laboratory and the Defense Advanced Research Projects Agency under agreement number FA8750-16-2-0204. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory and the Defense Advanced Research Projects Agency or the U.S. Government.

REFERENCES

[1] Shruti Agarwal and Hany Farid. 2017. Photo forensics from JPEG dimples. In *2017 IEEE Workshop on Information Forensics and Security (WIFS)*. 1–6.

[2] Mauro Barni, Luca Bondi, Nicolò Bonettini, Paolo Bestagini, Andrea Costanzo, Marco Maggini, Benedetta Tondi, and Stefano Tubaro. 2017. Aligned and non-aligned double JPEG detection using convolutional neural networks. *J. Visual Communication and Image Representation* 49 (Nov. 2017), 153–163.

[3] Tiziano Bianchi and Alessandro Piva. 2011. Analysis of non-aligned double JPEG artifacts for the localization of image forgeries. In *2011 IEEE International Workshop on Information Forensics and Security*. 1–6.

[4] Tiziano Bianchi and Alessandro Piva. 2012. Image Forgery Localization via Block-Grained Analysis of JPEG Artifacts. *IEEE Transactions on Information Forensics and Security* 7, 3 (June 2012), 1003–1017.

[5] Nicolò Bonettini, Luca Bondi, Paolo Bestagini, and Stefano Tubaro. 2018. JPEG Implementation Forensics Based on Eigen-Algorithms. In *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*. 1–7.

[6] Roberto Caldelli, Rudy Becarelli, and Irene Amerini. 2017. Image origin classification based on social network provenance. *IEEE Transactions on Information Forensics and Security* 12, 6 (June 2017), 1299–1308.

[7] Matthias Carnein, Pascal Schöttle, and Rainer Böhme. 2015. Forensics of high-quality JPEG images with color subsampling. In *2015 IEEE International Workshop on Information Forensics and Security (WIFS)*. 1–6.

[8] Chunhua Chen and Yun Q. Shi. 2008. JPEG image steganalysis utilizing both intrablock and interblock correlations. In *ISCAS*. IEEE, 3029–3032.

[9] Hany Farid. 2009. Exposing digital forgeries from JPEG ghosts. *IEEE Transactions on Information Forensics and Security* 4, 1 (2009), 154–160.

[10] Hany Farid. 2016. *Photo Forensics*. The MIT Press.

[11] Jessica Fridrich and Jan Kodovský. 2012. Rich Models for Steganalysis of Digital Images. *IEEE Transactions on Information Forensics and Security* 7, 3 (March 2012), 868–882.

[12] Thomas Gloe and Rainer Böhme. 2010. The 'Dresden Image Database' for Benchmarking Digital Image Forensics. In *Proceedings of the 2010 ACM Symposium on Applied Computing (SAC '10)*. ACM, New York, NY, USA, 1584–1590.

[13] Zhongwei He, Wei Lu, Wei Sun, and Jiwu Huang. 2012. Digital Image Splicing Detection Based on Markov Features in DCT and DWT Domain. *Pattern Recognition* 45, 12 (Dec. 2012), 4292–4299.

[14] Jan Kodovský and Jessica Fridrich. 2012. Steganalysis of JPEG images using rich models. In *Media Watermarking, Security, and Forensics*, Vol. 8303.

[15] ShiYue Lai and Rainer Böhme. 2013. Block convergence in repeated transform coding: JPEG-100 forensics, carbon dating, and tamper detection.. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013*. IEEE, Vancouver, BC, Canada, 3028–3032.

[16] Weiqi Luo, Zhenhua Qu, Jiwu Huang, and Guoping Qiu. 2007. A Novel Method for Detecting Cropped and Recompressed Image Block. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 217–220.

[17] Babak Mahdian and Stanislav Saic. 2009. Detecting double compressed JPEG images. In *3rd International Conference on Imaging for Crime Detection and Prevention (ICDP 2009)*. 1–6.

[18] Cecilia Pasquini, Giulia Boato, and Fernando Pérez-González. 2014. Multiple JPEG compression detection by means of Benford-Fourier coefficients. In *2014 IEEE International Workshop on Information Forensics and Security (WIFS)*. 113–118.

[19] Cecilia Pasquini and Rainer Böhme. 2019. Information-Theoretic Bounds for the Forensic Detection of Downscaled Signals. *IEEE Transactions on Information Forensics and Security* 14, 7 (July 2019), 1928–1943.

[20] Alin C. Popescu and Hany Farid. 2004. Statistical Tools for Digital Forensics. In *Proceedings of the 6th International Conference on Information Hiding (IH'04)*. Springer-Verlag, Berlin, Heidelberg, 128–147.

[21] Husrev Taha Sencar and Nasir Memon (Eds.). 2013. *Digital Image Forensics*. Springer, New York, NY.

[22] Dasara Shullani, Marco Fontani, Massimo Iuliani, Omar Al Shaya, and Alessandro Piva. 2017. VISION: a video and image dataset for source identification. *EURASIP Journal on Information Security* 2017, 1 (03 Oct. 2017).

[23] Shuiming Ye, Qibin Sun, and Ee-Chien Chang. 2008. Detecting Digital Image Forgeries by Measuring Inconsistencies of Blocking Artifact. In *2007 International Conference on Multimedia & Expo(ICME)*, Vol. 00. 12–15.

[24] Fabian Zach, Christian Riess, and Elli Angelopoulou. 2012. Automated Image Forgery Detection through Classification of JPEG Ghosts. In *Pattern Recognition — Joint 34th DAGM and 36th OAGM Symposium*. 185–194.