FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Image Forensics from Chroma Subsampling in High-Quality JPEG Images

Benedikt Lorch and Christian Riess
IT Security Infrastructures Lab
Friedrich-Alexander University Erlangen-Nürnberg
July 4, 2019

Forensic cues from JPEG implementations

Camera manufacturers and software engineers fine-tune their JPEG implementation to suit their tastes and needs. [Farid 2016]

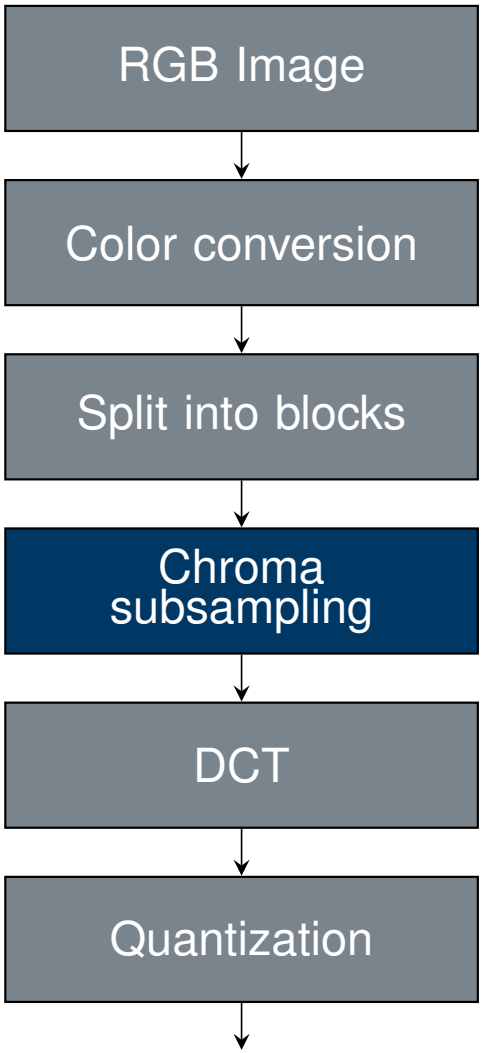
JPEG implementation characteristics provide cues for forensic tasks:

- estimate compression history [Bianchi 2012, Pasquini 2019]
- detect manipulations [Luo 2007]
- extract library fingerprint [Agarwal 2017, Bonettini 2018]

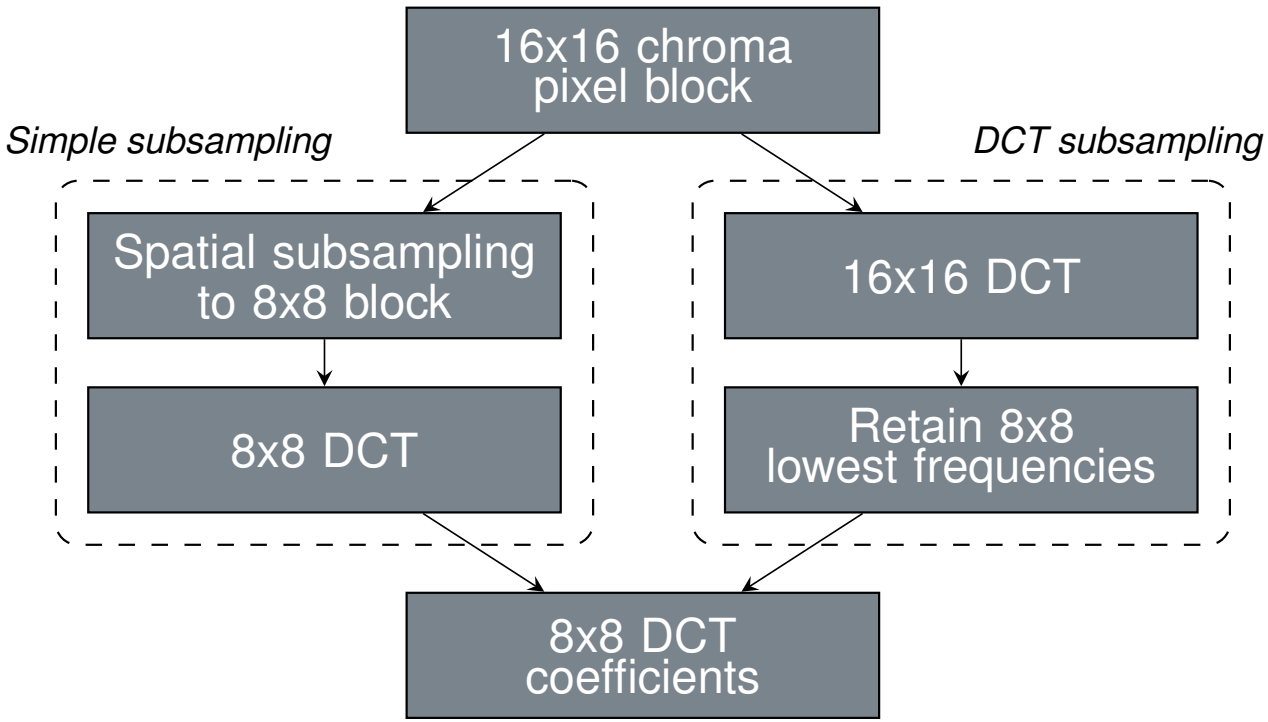
Our work: Fingerprint JPEG implementation from chroma subsampling

- Novel artifact arising from chroma subsampling
- Integer rounding lets every second column appear brighter than neighboring columns (**chroma wrinkles**)

Chroma subsampling in JPEG



Chroma subsampling variants

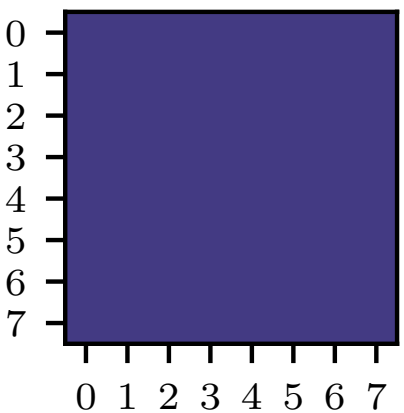


Simple vs. DCT subsampling

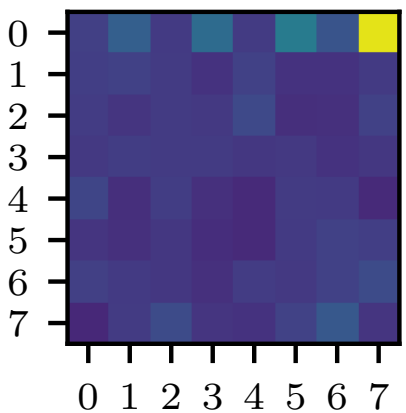
Simple subsampling

DCT subsampling

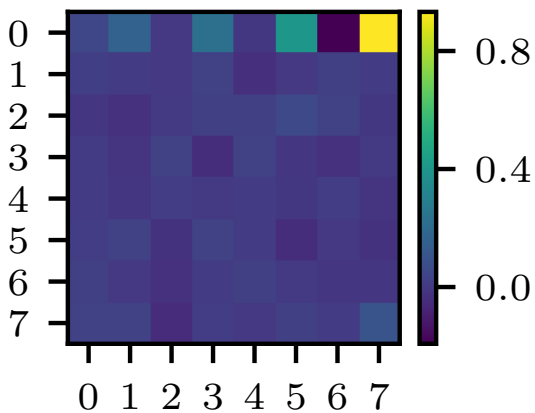
DCT difference in Y



DCT difference in Cb



DCT difference in Cr



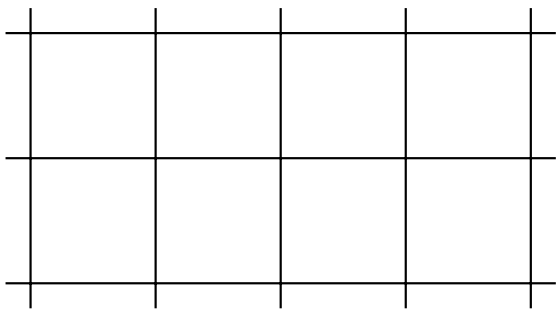
Artifact origin

Simple subsampling as implemented in `jcsample.c`

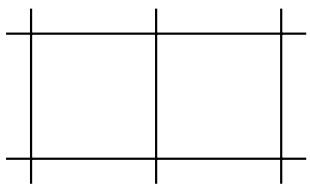
$$\mathbf{S}_{x,y} = \left\lfloor \frac{\mathbf{I}_{2x,2y} + \mathbf{I}_{2x+1,2y} + \mathbf{I}_{2x,2y+1} + \mathbf{I}_{2x+1,2y+1} + \mathit{bias}}{4} \right\rfloor$$

$$\mathit{bias} = \begin{cases} 1 & \text{for } x \bmod 2 = 0 \\ 2 & \text{for } x \bmod 2 = 1 \end{cases}$$

Input channel **I**



Subsampled channel **S**



Artifact origin

Simple subsampling as implemented in `jcsample.c`

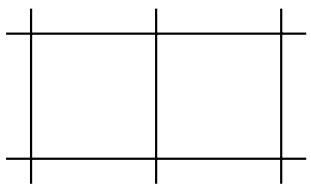
$$\mathbf{S}_{x,y} = \left\lfloor \frac{\mathbf{I}_{2x,2y} + \mathbf{I}_{2x+1,2y} + \mathbf{I}_{2x,2y+1} + \mathbf{I}_{2x+1,2y+1} + \mathit{bias}}{4} \right\rfloor$$

$$\mathit{bias} = \begin{cases} 1 & \text{for } x \bmod 2 = 0 \\ 2 & \text{for } x \bmod 2 = 1 \end{cases}$$

Input channel **I**

2	3	2	3
3	2	3	2

Subsampled channel **S**



Artifact origin

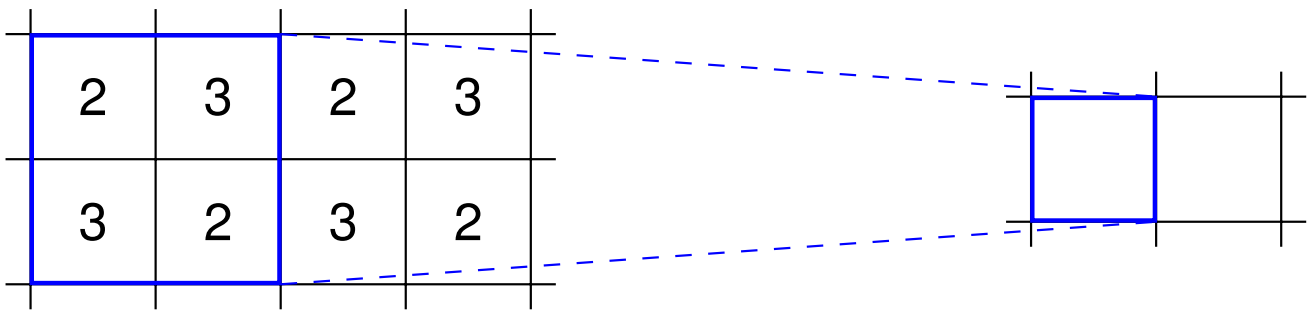
Simple subsampling as implemented in `jcsample.c`

$$\mathbf{S}_{x,y} = \left\lfloor \frac{\mathbf{I}_{2x,2y} + \mathbf{I}_{2x+1,2y} + \mathbf{I}_{2x,2y+1} + \mathbf{I}_{2x+1,2y+1} + \mathit{bias}}{4} \right\rfloor$$

$$\mathit{bias} = \begin{cases} 1 & \text{for } x \bmod 2 = 0 \\ 2 & \text{for } x \bmod 2 = 1 \end{cases}$$

Input channel **I**

Subsampled channel **S**

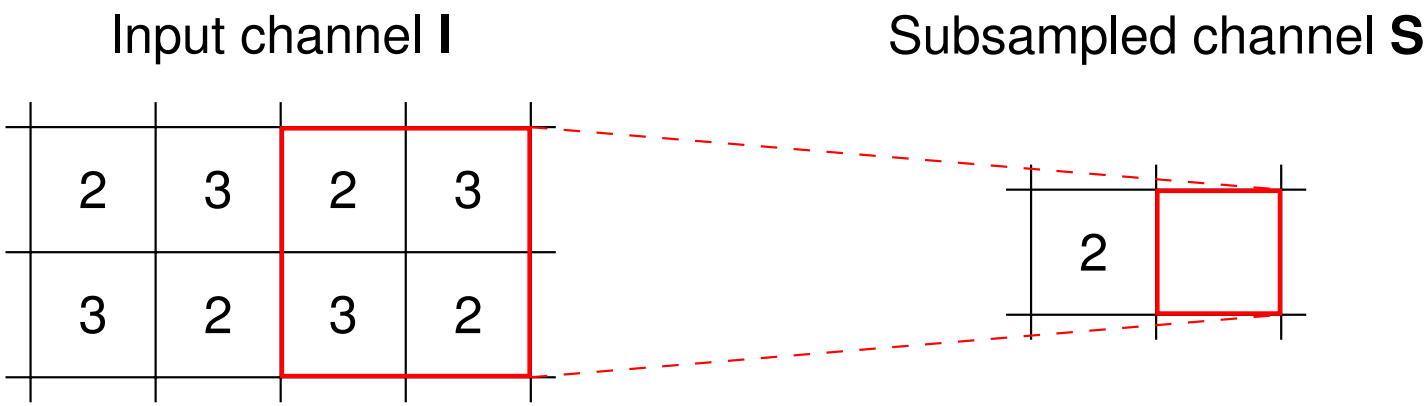


Artifact origin

Simple subsampling as implemented in `jcsample.c`

$$\mathbf{S}_{x,y} = \left\lfloor \frac{\mathbf{I}_{2x,2y} + \mathbf{I}_{2x+1,2y} + \mathbf{I}_{2x,2y+1} + \mathbf{I}_{2x+1,2y+1} + \mathit{bias}}{4} \right\rfloor$$

$$\mathit{bias} = \begin{cases} 1 & \text{for } x \bmod 2 = 0 \\ 2 & \text{for } x \bmod 2 = 1 \end{cases}$$



Artifact origin

Simple subsampling as implemented in `jcsample.c`

$$\mathbf{S}_{x,y} = \left\lfloor \frac{\mathbf{I}_{2x,2y} + \mathbf{I}_{2x+1,2y} + \mathbf{I}_{2x,2y+1} + \mathbf{I}_{2x+1,2y+1} + \mathit{bias}}{4} \right\rfloor$$

$$\mathit{bias} = \begin{cases} 1 & \text{for } x \bmod 2 = 0 \\ 2 & \text{for } x \bmod 2 = 1 \end{cases}$$

Input channel **I**

2	3	2	3
3	2	3	2

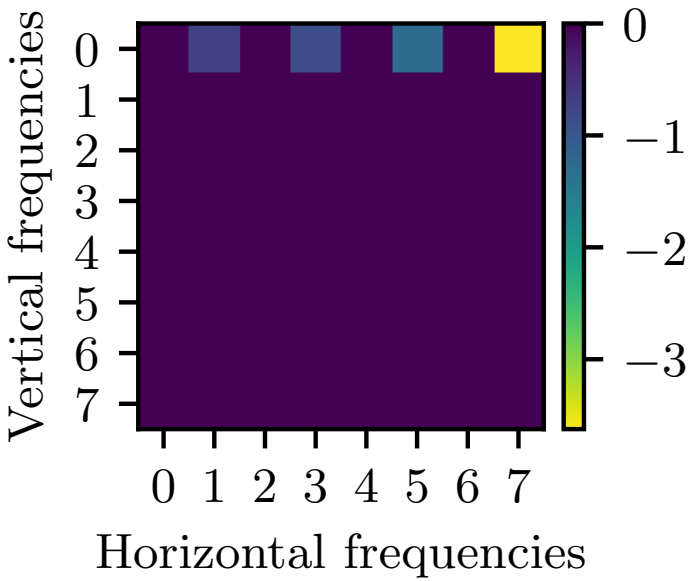
Subsampled channel **S**

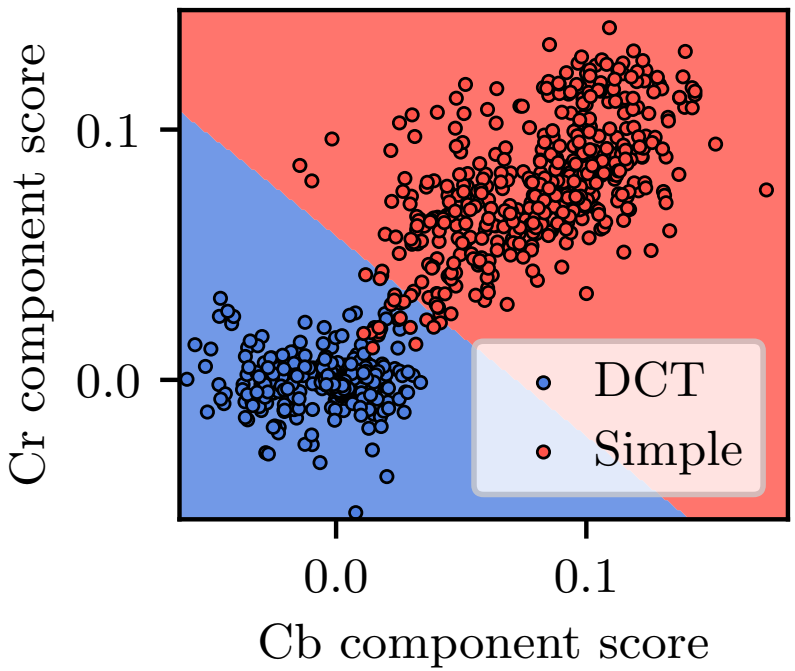
2	3

Bias introduces periodic artifact in horizontal direction (**chroma wrinkle**)

Correlation-based detector

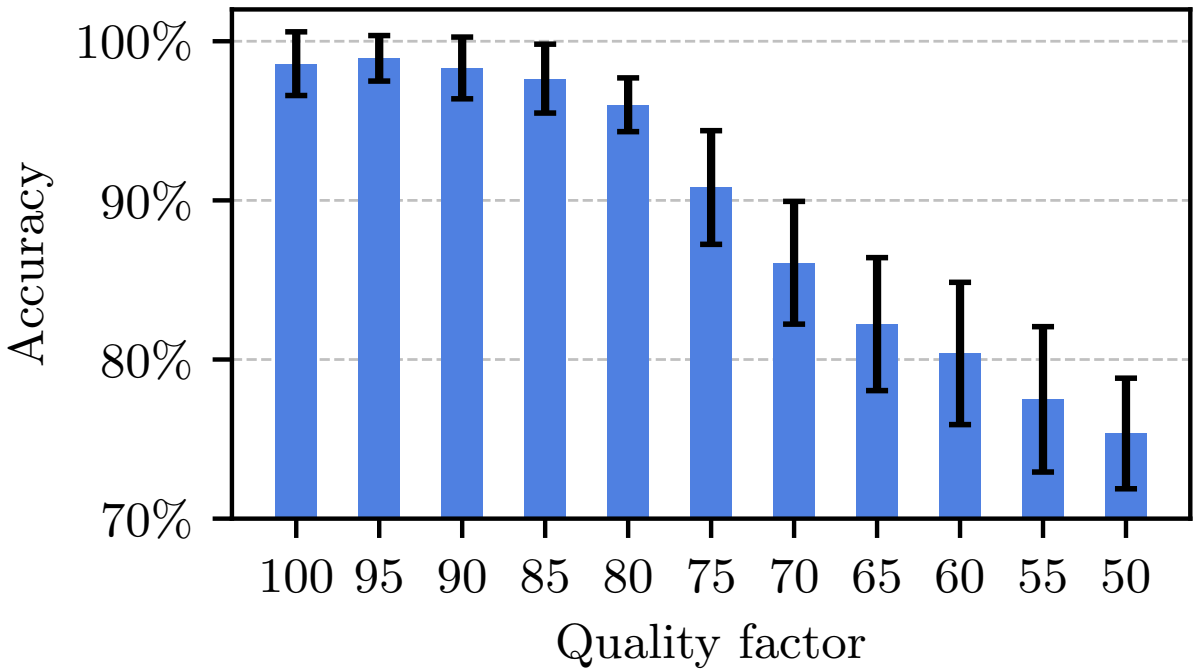
- Zero mean normalized cross-correlation with template in DCT domain
- Average correlations over all blocks
- Linear SVM on average correlation





Detection accuracy: Simple vs. DCT subsampling

Classification with linear SVM on block correlations



Correlation as simple yet effective detector for higher quality factors

Repeated up- and downsampling

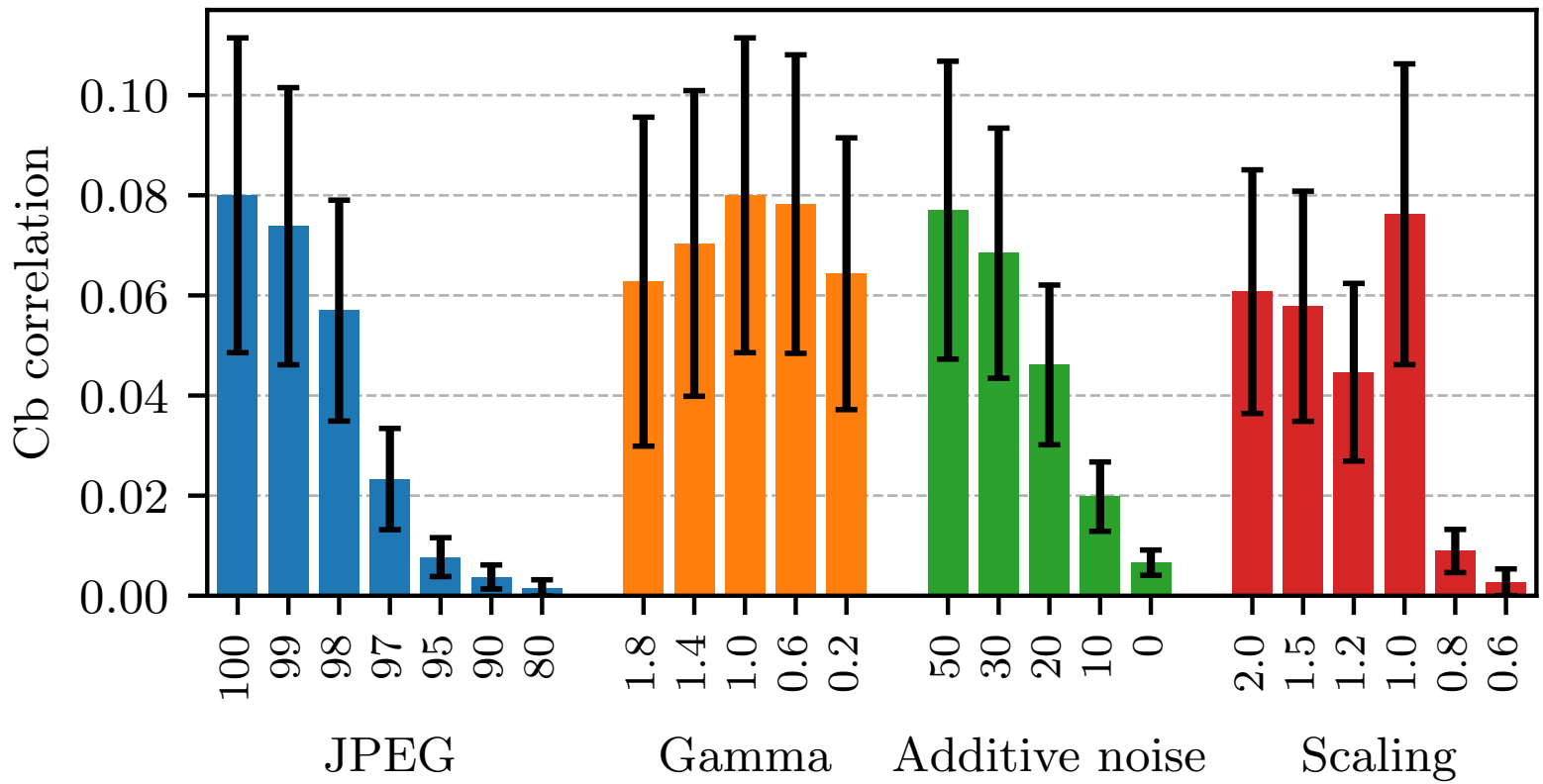
Effect of recompression on chroma wrinkles with quality factor 100

1^{st} compression	Decompression	2^{nd} compression	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

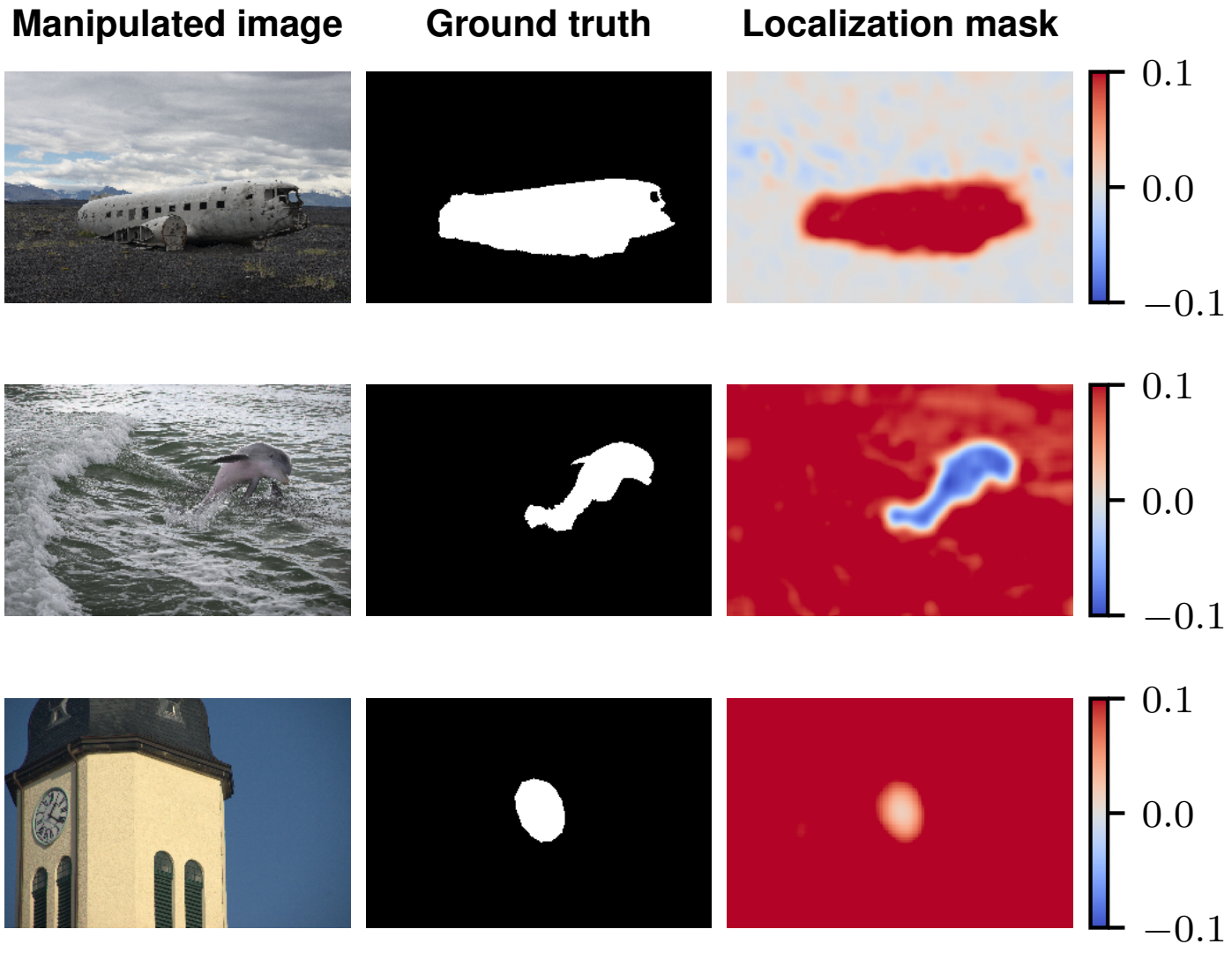
- Interplay of *DCT up-* and *simple subsampling* increases artifact strength
- Hint about previous compression

Resilience to global post-processing

Correlation scores in Cb channel after applying one of four common post-processing operations



Manipulation localization



Prevalence in software with default configuration

libjpeg and forks

Library	Chroma wrinkles?
libjpeg 6b	**Yes**
libjpeg 7	No
libjpeg 8 (and a, b, c, d)	No
libjpeg 9 (and a, b, c)	No
libjpeg-turbo 2.0.1	**Yes**
mozjpeg 3.3.1	**Yes**, for quality factors < 90

Editing software

Software	Chroma wrinkles?
Adobe Photoshop CC 2019	No for quality levels 7 through 12
Gimp 2.8.22	**Yes**, depends on OS[1]

[1] *libjpeg-turbo* is used or provided by Android, Debian, Ubuntu, among others.

Summary on chroma wrinkles

- Periodic artifact arising from chroma subsampling in *libjpeg-turbo*
- Correlation in DCT domain as simple yet effective detector

Applications

- Fingerprint JPEG compression library
- Detect double compression
- Localize manipulations

Future work

- Surpress scene content to isolate artifact from scene
- More effective detector against higher compression rates

Thank you

References

- Hany Farid, *Photo Forensics*, The MIT Press, 2016.
- Tiziano Bianchi and Alessandro Piva, *Image forgery localization via block-grained analysis of JPEG artifacts*, TIFS, 2012.
- Cecilia Pasquini and Rainer Böhme, *Information-theoretic bounds for the forensic detection of downscaled signals*, TIFS, 2019.
- Weiqi Luo, Zhenhua Qu, Jiwu Huang, Guoping Qiu, *A novel method for detecting cropped and recompressed image block*, ICASSP, 2007.
- Shruti Agarwal and Hany Farid, *Photo forensics from JPEG dimples*, WIFS, 2017.
- Nicolò Bonettini, Luca Bondi, Paolo Bestagini, Stefano Tubaro, *JPEG implementation forensics based on eigen-algorithms*, WIFS, 2018.