https://doi.org/10.2352/EI.2022.34.4.MWSF-339

Image montage detection based on image segmentation and robust hashing techniques

Martin Steinebach, Tiberius Berwanger, Huajian Liu; Fraunhofer SIT / ATHENE, Darmstadt, Germany

Abstract

We present a new method for detecting montages and, in general, recognizing images or parts of images. Image recognition is becoming increasingly important, for example, in detecting copyright infringement, disinformation that puts images in a different context, detecting child pornography in image collections. Numerous methods based on robust hashing and feature extraction, more recently also supported by machine learning, are already known for this purpose. Inverse image search solutions for users are also available here. In general, however, these methods are either only robust to a limited extent against changes such as rotation and cropping or they require a high data and computational effort. Especially when several images are copied into one another and montages are created, automated recognition has been difficult to achieve up to now.

Motivation

Even before the term *Fake News* was popular, there were already photo manipulations that contributed to disinformation. It is known that already since 1864 the first fake pictures were created for this context. [1] Various image manipulations are used to make images appear differently. Even a slight cropping of the image can be used to misrepresent an image. A popular form of image manipulation is to create image montages. This involves taking an image element, such as a person, from an existing image and inserting it into someone else's image. As an example, the Malaysian politician Jeffrey Wrong Su En created an image montage to increase his reputation among the population. In doing so, he cut Ross Brawn out of a photo of him being knighted by the Queen of England and replaced it with a picture of himself. This made it appear that Jeffrey Wong Su En was knighted [2] .

Due to the continuous development of image processing programs, it is becoming increasingly difficult to distinguish manipulated images with the human eye. Thus, automated recognition systems are needed to help detect image forgeries. Image manipulations such as rotations or changes in image illumination ratios, which are additionally applied to image montages, make computer-aided detection more difficult.

For these reasons, this work addresses image montage recognition. A previous work by us addresses the detection of image montages based on feature detection [3]. This achieves high detection scores and is robust against a variety of different manipulation techniques. However, the system created by [3] is of limited practical use. The demand on the disk space and memory grow with the number of stored feature descriptors extracted from the original images.

The goal of this work is to enable image montage detection while ensuring practical features. The approach is to classify image montages based on image segmentation and robust hashing techniques. The retrieval of an image for a manual test, which could be used for journalists, should take less than one second. Therefore the performance of the detection system is of special importance. In order to form a comparison to [3], the evaluation is done with the same manipulation techniques and their defined parameters.

Background

The task of recognizing images can be found in several sub-disciplines in the literature. Examples are person, car or face recognition.[4] [5] [6] Here, these areas are not concerned with the entire image, but rather on the objects that are contained in the image. An example of this would be person re-identification, in which the goal is to recognizing persons in different images. Basically, however, the recognition disciplines that focus on specific areas such as people can be summarized as object re-identification. This means that not the whole image is recognized but single parts of the image. Image re-identification is also used in image authentication, for example through content authentication. In the literature, the term *Near Duplicate Detection* is often used to describe image recognition. Near duplicates are image copies of original images that have been slightly manipulated, such as by light conditions or lossy compression [7].

In image recognition, a fundamental distinction must be made between feature-based and robust hash-based techniques. Both of them try to achieve robustness against conventional image manipulation.

Feature based approach The basic concept of feature detection is to find features from areas of interest. Subsequently, the areas are then extracted and described by a feature descriptor. This description can then be used for re-identification [8].

The basic method of an image re-identification by the feature based methods is as follows: at the beginning, an image database containing the original images is created. Then, feature descriptors are extracted from all images and stored in a feature vector database. This is then used as a reference. If a new image is to be identified, the feature descriptors are also extracted from this image and compared with the feature vectors from the reference database. If a match is found, the requested image is marked as a near duplicate image. The match is thereby based on a reference value, or threshold, which determines the minimum similarity of the vectors [7].

Hash based approach Hash-based algorithms are used in various application areas, such as image search, duplicate or near duplicate detection, or image authentication. [9] [10] [11] [12] Hash functions can be divided into the two categories of cryptographic hashes and robust hashes. Cryptographic hashes are very sensitive with respect to the input data. If only 1-bit changes in the source file, when the hash is regenerated, it results in a completely new and not similar hash to the original. With a lossy compression, the original and the compressed variant would give completely different hash results. It does not matter that both images do not differ much visually. Hash-based approaches in the image context are called robust hashes or perceptual hashes. These are to be distinguished from the conventional cryptographic hash algorithms such as the MD5 hashes. Robust hashes are not very sensitive to slight modifications such as lossy compression. Even with compression, the resulting hashes would be very similar. Thus, when identifying images, the use of robust hashes is more appropriate than cryptographic hashes.

Own Previous Work

The robust hash applied in the work is the ForBild block hash presented by us in [13]. It is the result of an evaluation of image hashing methods [14]. Based on this hash, we have added segmentation countermeasures based on face detection [5] and watershed image segmentation [12]. Beyond the recognition of images, we also addressed the possibility of combining privacy and robust hashing in [15]. As an alternative to robust hashing, we also evaluated feature-based montage detection utilizing SIFT and SURF in [3]

In this work image montage detection is done by recognizing image segments. The reasoning is that every montage is based on existing images. If these images are known and their usage is detected, one can reliably identify montages. An alternative to this is the forensic approach. Here image objects inserted into a background images are recognized by splicing detection, further discussed by us in [16]. Forensic approaches do not require image references, but their detection rates are significantly lower than image re-identification. Another alternative is the application of robust signature schemes as discussed in [17]. Here image montages could be recognized by a significant difference between reference and actual image. There are also digital watermarking concepts for detecting changes within images, especially fragile and feature-fragile algorithms. Here before image distribution a watermark is embedded as a security seal [18] [19] [20] [21].

Concept

Figure 1 shows the basic idea of an image montage: The triangle in the upper right is to be inserted in the image on the upper left. The lower left shows a simple montage where the triangle is simply inserted next to the other image objects with some scaling. The lower right shows an example with rotation, scaling and overlapping of objects.

To recognize montage objects like the triangle in the lower right of figure 1, we follow a simple concept: every image added to the collection of known references is first divided into image objects which can be seem as 'atoms' which would not be further truncated by creators of montages. These objects are placed against a black background. To counter potential rotation, their geometrical alignment is normalized. After normalization, a robust hash is created from the bounding box around the object. Figure 2 shows the first steps of this process.

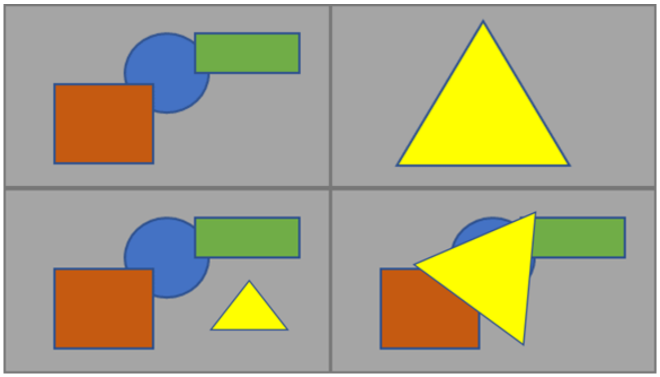


Figure 1: Abstract montage concept

Implementation

To implement the concept of the previous section, we implemented the following modules:

Image Segmentation

For segmentation, we use BlendMask which is based on machine learning. It is an instance-based segmentation, as a further development of semantic segmentation and object detection. Overlapping objects like persons are individually masked. However, unlike semantic segmentation, the entire image is not segmented, but only the objects found. The BlendMask model allows instance based segmentation of an image. [22]

Instance segments are formed from previously learned objects. For learning, the *Microsoft COCO train2017 instance segmentation* dataset was used. It is important to segment all relevant objects from an image. The specified *confidence-threshold* plays an important role. Depending on the specified value, different numbers of objects are recognized. The default value given by AdelaiDet is 0.5 and the value in the BlendMask publication is 0.35. For this work a value of 0.20 was chosen.

Rotation Normalization

All object segments are handled individually. This involves highlighting each instance segment from the image by setting the background or pixels around the object to white (255). A copy of the highlighted object segment is converted from the three-dimensional to the two-dimensional plane. This is necessary for the calculation of the orientation as the applied image orientation algorithm cannot handle three-dimensional images.

The resulting two-dimensional image is analyzed by the ScikitImage regionprops function. The object is discarded if it has a resolution smaller than 50 x 50, because it is assumed that such a small object has little relevance for a montage and rather an error of the object detection is present. If an object is large enough, the orientation is extracted and the image is rotated by the negative value of the orientation. Thus, the orientation of the object is normalized on the y-axis. The three-dimensional image is now used as the object to be rotated around which a bounding box is placed. For the rotation, it is important that the image is still displayed completely after the rotation, and is not cropped by the previous resolution ratios. The process is shown visually in Figure 2.

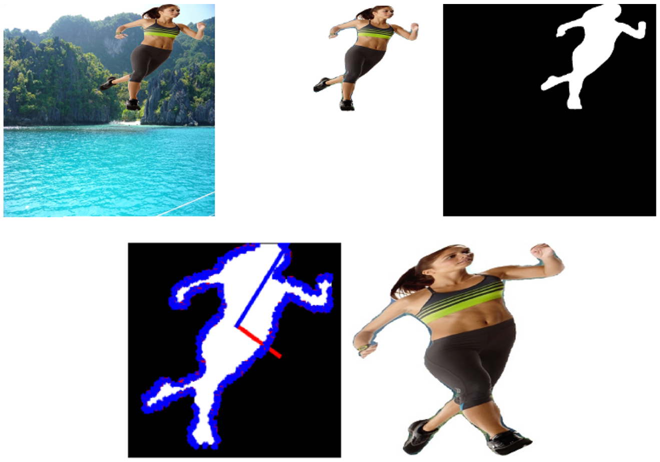


Figure 2: process of object normalization

Robust Hash Generation

Robust hash generation is provided by ForBild. It is a block hash algorithm based on down-scaling an arbitrary image size to a 16x16 pixel gray scale image and calculating the hash bit values by comparison of individual pixels to the median of all pixels. Various additional steps are taken to improve robustness [23].

Matching and Verification

Once all robust hashes have been computed, matching of the hashes is performed against the specified reference database. The Hamming distance, weighted Hamming distance, and *similarity score* are used for matching. Each object segment is checked in turn. A hash table is created and the image to be checked is stored as a key value. The source references found from the matching check are now assigned to the key value. If there are 2 or more different source references in the key value, this is stored in a list that represents the found montages. This list is output after all object segments have been checked.

Evaluation

For an exhaustive automated test, the required image sets were created using a montage creation script. For the construction of image montages, 2000 images are used as backgrounds and 1000 transparent images act as inserted objects. Another 1000 images are for the false positive evaluation. This results in a large test image set due to the different manipulation parameters and their different resolutions that need to be tested. Complex objects were also considered in the choice of objects. For simple objects like a dog the object recognition is almost 100% precise, but this would not be realistic. A montage consists of strictly one inserted object in order to be compliant with the evaluation from [3] on the one hand, but also to exclude potential object occlusions that can influence the evaluation.

The following components were used for the evaluations:

- Operating system: Ubuntu 20.04.2 LTS
- Graphics card: NVIDIA GeForce RTX 2080 Ti
- Processor: Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz
- Memory: 32 GB, DDR4
- Hard disk storage: HDD, SATA 6Gb/s

Two evaluations were performed for montage recognition. The first concerns background recognition and the second object recognition. Using the example from figure 1, background recognition would find the upper left image as the background used in the lower montage examples. Object recognition would find the triangle from the upper right image. Thus, there are potentially different results for the chosen metrics as far as background recognition and object recognition are concerned. For example, a montage may not be detected because the object is not detected but the background is detected correctly.

This would result in a 100% TPR for background detection and a 0% TPR for object detection. [3] Therefore, the metrics are each measured and listed separately. Due to the large number of different paramter options, it was necessary to settle on a set of parameter values. These values concern the *confidence threshold* for segmentation and the value of the *similarity score* of ForBild. Changing one value would mean a complete new evaluation of the entire image sets. The range of values for the *confidence threshold* of the segmentation and the *similarity score* of ForBild is 0.10-0.50 and 70-90, respectively. This results in an extremely high test effort. Thus, the values for the *similarity score* were set to 80 and for the *confidence threshold* to 0.20. Furthermore, an additional test was performed to evaluate the machine segmentation in combination with the ForBild robust hash for image recognition. For this purpose, the montage backgrounds were used for the image recognition test. Since the effort to test each image set manually would be too large, the MontageEvaluation script was created, which performs the evaluation automatically.

- Objects: To create a montage, objects are needed that are inserted into a background. The transparent objects used for the work are from *pngimg.com*. The website offers transparent PNG objects that are divided into classes. 30 object classes were selected to serve as objects. The selected classes are detectable by the trained BlendMask model. When selecting the classes, care was taken to ensure that they are recognizable on the one hand, but also complex. It makes little sense to use only a simple object class like dog. The detection would be thereby very easy, since the objects are not complex, however that would not be realistic. For this reason, classes were chosen that have complex shapes, such as the class bird. The objects are much more complex, because they have fine structures, which make an exact segmentation much more difficult, but realistic. These are different classes of objects, with people making up a large portion of the set of objects. These are also represented in different and complex poses, such as dancing. The images were downloaded by a self-written *crawler*, and checked for duplicates by the ForBild algorithm. The objects are inserted into the object image and montages.
- Background image set for object images:
  For the creation of the object images, the *The INRIA Holidays* dataset [24] was used as background. The same database was also used in [3] for the backgrounds of the object images.
- Background image set for montages:Ṫo create the montages, the Cityscape dataset [25] was used as background. The reason for this is that the images have objects like people or cars that are detectable by the model. Due to the fact that the model was trained over the *Microsoft Common Ob-*

jects in Context (COCO) dataset, only certain object types are available for detection. However, the model can be easily extended if, for example, the *google open image* database [26] is used for training. This should also improve the AP, since much more training data is available. Another reason that contributed to the selection of the Cityscape dataset is that the images are presented as a scene. Unlike the *Microsoft Common Objects in Context (COCO)* dataset, which represents images that have a focus on individual objects. This also makes the dataset not a good background option for montages. If this dataset would be used, a detection of the background would be very badly possible, because the objects in the background would be almost always cropped.

- Image set for the *false positive* Evaluation: For the evaluation of the *false positives* needed for background recognition and object recognition, the *Microsoft Common Objects in Context (COCO)* dataset was used. The reason for this is that the object types contained in the dataset were also used when training the BlendMask model. However, the recognition system does not know these images because the dataset is used for COCO 2017 validation. It would not be meaningful to use another arbitrary dataset that has no objects that can be recognized by the model. This would greatly reduce the FP, but would skew the results.

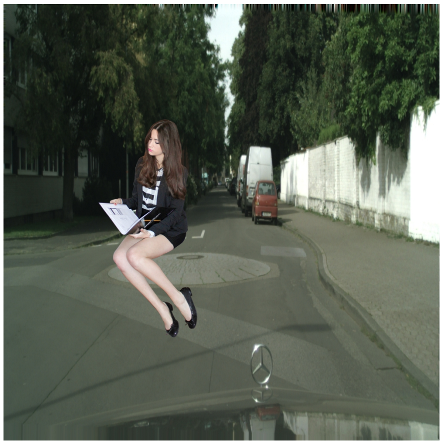


Figure 3: Example of automated montage

Results

The following results are only excepts of the analysis. The focus is on rotation robustness. We also provide results for added noise as an additional example.

Object rotation: object detection

The test is performed in 3 different image resolutions with 9 different rotation values. Without a rotation-countermeasure for almost all rotation values, a TPR of less than 8% is achieved. The exception is the 180° rotation value, which represents an inverted reflection, and is detected by the mirror robustness of the ForBild robust hash with a TPR of over 50%.

The graph in Figure 4 shows the detection rate at three different image resolutions with 9 different rotation values in which orientation detection was used by the Scikit regionprops method. The TPR remains above 70% for the rotation values from -20° to 30° at a resolution of 1000 x 1000 pixels. The TPR of a 50% object in a 1000 x 1000 pixel montage is at least 80%. This results in a recognition value of ≈87.5% for the realistic practical rotation values. The FPR is 19.3% and the Precision rounded down is 78%.

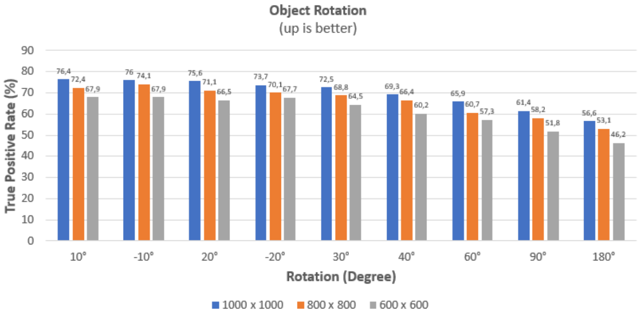


Figure 4: Rotation robustness by Scikit

Object rotation: background detection

The test refers to the background detection of the same montages as used in the object rotation test. The results are constant for all rotations and differ by a maximum of 2.1% at a resolution of 1000 x 1000 pixels. Thus, a TPR of over 77.4% is achieved. Based on the fact that the background has a resolution of 1000 x 1000 and a contained object of the object size of 50% of the resolution, and a recognition of 79.6% on average is achieved, this results in a recognition after the rotation modification of ≈98.4%. Thus, the scikit modification for orientation recognition biases the recognition by ≈1.6%. The FPR remains at 19.6 and the precision rounded down at 79%.

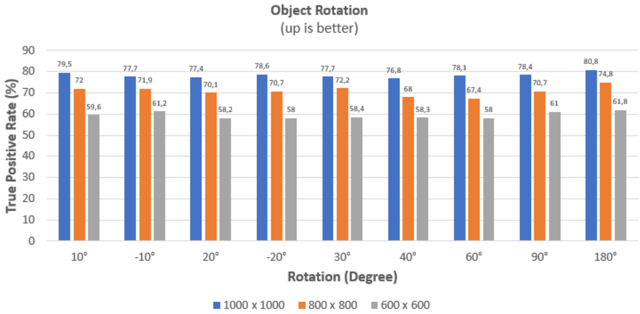


Figure 5: Object rotation results for background detection

Added noise: object detection

For the image noise test, the entire montage is overlaid with a Gaussian noise. The different noise level parameters are:

- noise 1: No noise
- noise 2: Weak noise, mean = (10,12,34)/3, variance = (1, 5, 25)/3
- noise 3: Medium noise, mean = (10,12,34)/3, variance = (1, 5, 25)/3
- noise 4: Strong noise, mean = (10,12,34)/3, variance = (1, 5, 25)/3

The method used to generate the noise is the *randn* method from

OpenCV. The same method was also used in [27]. An example is shown in figure 6

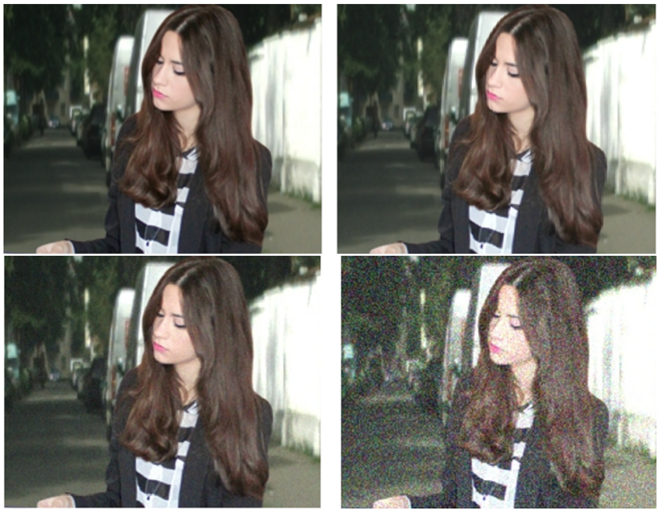
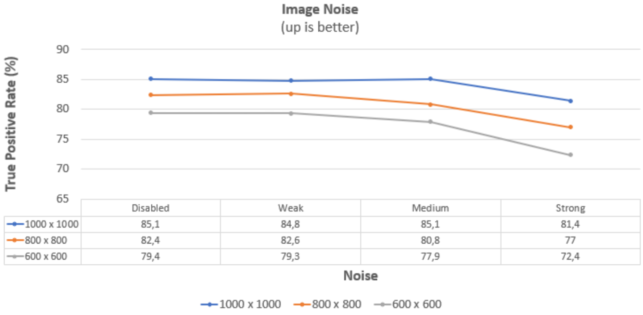


Figure 6: Noise addition. Upper left: noise 1, upper right: noise 2, lower left: noise 3, lower right: noise 4

The test is performed in 3 different image resolutions with the 4 noise values. The TPR is consistently above 80% at a resolution of 1000 x 1000 pixels. The FPR is 6.3% and the precision is 93% rounded off.



	Disabled	Weak	Medium	Strong
1000 x 1000	85,1	84,8	85,1	81,4
800 x 800	82,4	82,6	80,8	77
600 x 600	79,4	79,3	77,9	72,4

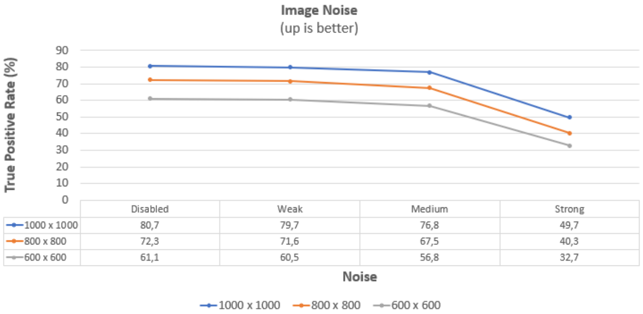
Figure 7: Image noise results for object detection

Added noise: background detection

The test refers to the background detection of the same montages as used in the image noise test. The results show that all resolution levels only deteriorate by a maximum of 1% in the case of weak image noise, but also drop sharply from the recognition rate in the case of strong noise. The background detection is affected by a stronger drop of the detection rate than the object detection. The TPR remains above 76.8% at a resolution of 1000 x 1000 pixels up to the medium noise level, which corresponds to a maximum loss of 4.1% compared to the non-manipulated montage. The FPR is 3.8% and the precision rounded off at 94%.

Summary and Conclusion

Compared to our feature-based [3], lower recognition rates are achieved. The runtime is also somewhat higher. However, in contrast to [3], the system also has robustness against mirroring. Practically, the custom-built system is significantly more applicable than the feature-based variant. The memory requirements are significantly lower, as only a fraction of the of the hard disk memory and the main memory is needed. Thus, the recognition system scales very well even for millions of images.



	Disabled	Weak	Medium	Strong
1000 x 1000	80,7	79,7	76,8	49,7
800 x 800	72,3	71,6	67,5	40,3
600 x 600	61,1	60,5	56,8	32,7

Figure 8: Image noise results for background detection

Parameters for segmentation and matching the hash values can still be optimized. The created system is modular. Thus instead of the BlendMask segmentation also another another segmentation model can be used instead of the BlendMask segmentation. However, the requirement for this is that compatibility with the Detectron2 framework is given. The detection rates are significantly influenced by the object detection and the segmentation. Nevertheless, the recognition system produces comparably good recognition results as long as the images have an image scaling of 1000 x 1000 pixels. The runtime is well below the defined target of one second and can, due to the low runtime can potentially also be used for automated recognition, e.g. in file uploads. Image recognition is performed using the same recognition system. An average recognition of over 85% across multiple tamper types is achieved, where the FPR drops to 0.1% by keeping the background. Based on these recognition values and the low FPR, the recognition system is classified as a Robust Image Recognition System.

Acknowledgment

This research work has been funded by BMBF and the Hessen State Ministry for Higher Education, Research and the Arts within their joint support of the National Research Center for Applied Cybersecurity ATHENE.

References

[1] Hany Farid. Photo tampering throughout history.

[2] Judith Redi, Wiem Taktak, and Jean-Luc Dugelay. Digital image forensics: A booklet for beginners. *Multimedia Tools Appl.*, 51:133–162, 10 2011.

[3] Martin Steinebach, Karol Gotkowski, and Hujian Liu. Fake news detection by image montage recognition. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, pages 1–9, New York, NY, USA, 2019. ACM.

[4] Mang Ye, Jianbing Shen, Gaojie Lin, Tao Xiang, Ling Shao, and Steven C. H. Hoi. Deep learning for person re-identification: A survey and outlook. *IEEE transactions on pattern analysis and machine intelligence*, PP, 2021.

[5] Martin Steinebach, Huajian Liu, and York Yannikos. Facehash: Face detection and robust hashing. In Pavel Gladyshev, Andrew Marrington, and Ibrahim Baggili, editors, *Digital Forensics and Cyber*

Crime, volume 132 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 102–115. Springer International Publishing, Cham, 2014.

[6] Sultan Daud Khan and Habib Ullah. A survey of advances in vision-based vehicle re-identification. *Computer Vision and Image Understanding*, 182:50–63, 2019.

[7] K. K. Thyagharajan and G. Kalaiarasi. A review on near-duplicate detection of images using computer vision techniques. *Archives of Computational Methods in Engineering*, 28(3):897–916, 2021.

[8] Ali Ismail Awad and Mahmoud Hassaballah, editors. *Image Feature Detectors and Descriptors: Foundations and Applications*, volume 630 of *Studies in Computational Intelligence*. Springer International Publishing, Cham and s.l., 1st ed. 2016 edition, 2016.

[9] Andrea Drmic, Marin Silic, Goran Delac, Klemo Vladimir, and Adrian S. Kurdija. Evaluating robustness of perceptual image hashing algorithms. In *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 995–1000. IEEE, 2017.

[10] Dat Tien Nguyen, Firoj Alam, Ferda Ofli, and Muhammad Imran. Automatic image filtering on social networks using deep learning and perceptual hashing during crises.

[11] Ling Du, Anthony T.S. Ho, and Runmin Cong. Perceptual hashing for image authentication: A survey. *Signal Processing: Image Communication*, 81:115713, 2020.

[12] Martin Steinebach, Huajian Liu, and York Yannikos. Efficient cropping-resistant robust image hashing. In *2014 Ninth International Conference on Availability, Reliability and Security*, pages 579–585. IEEE, 2014.

[13] Martin Steinebach. Robust hashing for efficient forensic analysis of image sets. In Pavel Gladyshev and Marcus K. Rogers, editors, *Digital Forensics and Cyber Crime*, volume 88 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 180–187. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[14] Christoph Zauner, Martin Steinebach, and Eckehard Hermann. Rihamark: perceptual image hash benchmarking. In Nasir D. Memon, Jana Dittmann, Adnan M. Alattar, and Edward J. Delp III, editors, *Media Watermarking, Security, and Forensics III*, SPIE Proceedings, page 78800X. SPIE, 2011.

[15] Uwe Breidenbach, Martin Steinebach, and Huajian Liu. Privacy-enhanced robust image hashing with bloom filters. In Melanie Volkamer and Christian Wressnegger, editors, *ARES 2020: The 15th International Conference on Availability, Reliability and Security, Virtual Event, Ireland, August 25-28, 2020*, pages 56:1–56:10. ACM, 2020.

[16] Raphael Antonius Frick, Huajian Liu, and Martin Steinebach. Detecting double compression and splicing using benfords first digit law. In *Proceedings of the 15th International Conference on Availability, Reliability and Security*, pages 1–9, 2020.

[17] Martin Steinebach, Sebastian Jörg, and Huajian Liu. Checking the integrity of images with signed thumbnail images. *Electronic Imaging*, 2020(4):118–1, 2020.

[18] Stefan Katzenbeisser, Huajian Liu, and Martin Steinebach. Challenges and solutions in multimedia document authentication. In *Handbook of Research on Computational Forensics, Digital Crime, and Investigation: Methods and Solutions*, pages 155–175. IGI Global, 2010.

[19] Stefan Thiemert, Hichem Sahbi, and Martin Steinebach. Using entropy for image and video authentication watermarks. In *Security, Steganography, and Watermarking of Multimedia Contents VIII*, volume 6072, page 607218. International Society for Optics and Photonics, 2006.

[20] Hongliang Cai, Huajian Liu, Martin Steinebach, and Xiaojing Wang. A roi-based self-embedding method with high recovery capability. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1722–1726. IEEE, 2015.

[21] Huajian Liu and Martin Steinebach. Digital watermarking for image authentication with localization. In *2006 International Conference on Image Processing*, pages 1973–1976. IEEE, 2006.

[22] Hao Chen, Kunyang Sun, Zhi Tian, Chunhua Shen, Yongming Huang, and Youliang Yan. Blendmask: Top-down meets bottom-up for instance segmentation.

[23] Martin Steinebach, Huajian Liu, and York Yannikos. Forbild: Efficient robust image hashing. In *Media Watermarking, Security, and Forensics 2012*, volume 8303, page 83030O. International Society for Optics and Photonics, 2012.

[24] Herve Jegou. The inria holidays dataset, 2006.

[25] Cityscape. Dataset overview – cityscapes dataset, 2021-09-08.

[26] OpenImage. Open images v6 - download, 2021-06-25.

[27] Karol Gotkowski. *Erkennung von Montagen zur Bekämpfung von Fake News*. B. sc. thesis, Technische Universität Darmstadt, Darmstadt, 2019.

Author Biography

Prof. Dr. Martin Steinebach is the manager of the Media Security and IT Forensics division at Fraunhofer SIT. In 2003 he received his PhD at the Technical University of Darmstadt for this work on digital audio watermarking. In 2016 he became honorary professor at the TU Darmstadt.

Tiberus Berwanger worte his thesis on the paper topic as a master student of TU Darmstadt at Fraunhofer SIT.

Huajian Liu received his B.S. and M.S. degrees in electronic engineering from Dalian University of Technology, China, in 1999 and 2002, respectively, and his Ph.D. degree in computer science from Technical University Darmstadt, Germany, in 2008. He is currently a senior research scientist at Fraunhofer Institute for Secure Information Technology (SIT). His major research interests include information security, digital watermarking, robust hashing and digital forensics.