A Comparative Study on Image Hashing for Document Authentication [*][†]

Dominik Klein[‡]
Bundesamt für Sicherheit in der Informationstechnik, 53133 Bonn
`dominik.klein@bsi.bund.de`

Jan Kruse[§]
Hochschule Emden/Leer, 26723 Emden Stadt
`kruse.jan.g@gmail.com`

January 5, 2016

Abstract

A digital seal is a cryptographically signed 2D barcode printed on a document to verify the document's authenticity and integrity. In order to secure a printed image against tampering, a compact *image hash* can be stored inside the barcode. We investigate and experimentally evaluate various methods from the areas of image hashing and face verification w.r.t. its appropriateness for such an image hash that is resistant to a print-scan transformation. Exhaustive experiments with genuine security paper show that compressed local binary patterns (LBPs) and feature extraction by DCT-II perform best in this setting.

1 Introduction

We are concerned with a very specific and practical scenario of image hashing to verify the authenticity of a document. Consider a paper document which has important information printed on it and is usually protected against forgery by physical properties of the paper. Such documents are used in various domains, here we focus on visa stickers as an example. A simple, yet highly effective way to ensure integrity and authenticity of the printed information is the use of a

[*]Portions of the research in this paper use the FERET database of facial images collected under the FERET program, sponsored by the DOD Counterdrug Technology Development Program Office

[†]This is an author version without proof reading. The final publication is available at `http://dx.doi.org/10.1109/BIOSIG.2015.7314592`

[‡]Corresponding author

[§]The research presented here is based on the second author's Bachelor thesis *Authentifizierung von Lichtbildern durch Image Hashing, 2015.*

digital seal[1]. A cryptographic signature of the printed information is created using a private key, and this cryptographic signature is encoded into a two-dimensional barcode and printed on the visa itself. The corresponding public key is distributed, and any interested party can verify the printed information on the visa sticker by reading the barcode and verifying the signature with the public key.

To prevent impersonation attacks (person B uses a visa issued to person A), a facial image of the visa holder is usually printed on the sticker. Barcodes can store up to approximately 3000 bytes, but the usable capacity depends on the available physical size, and practically it is impossible to store a photo inside. One solution to detect tampering of the printed image is to create a *perceptual image hash* of the image. Such a hash is much smaller in size compared to the original image and can thus be stored inside the cryptographically signed barcode. To verify, one can scan or photograph the image from the document, compute its image hash, and compare that image hash against the digitally signed image hash of the enrollment image that is stored inside the barcode. Note that this scenario is both *offline* and *local*, and hence distinct from any client-server based method for tamper detection, where information about, or the document itself, is stored in a central database.

The desired property of an image hash function is to always map perceptually similar images to similar hash values, whereas perceptually different images should be always mapped to different hash values. Two hash values are then compared w.r.t. some measure, and considered to be the same, if their distance is below a threshold. Our notion of perceptual difference is inspired by the above scenario: We consider two images to be perceptually similar, if both are the same, or one image is the result of printing and then re-scanning the other one. Two images are considered perceptually different, otherwise. To construct image hashing functions in practice, image invariant features are extracted from the image, and a difference measure compares these extracted features.

Typical tampering is cutting out and replacing the image from a genuine visa. To prevent that, the hash ideally has second preimage resistance and collision resistance. Even if this is practically infeasible, by just ensuring that the number of collisions is low and the image can no longer be replaced by another *arbitrary* one, typical fraud scenarios are prevented.

The contribution of this paper is: 1) We survey different feature extraction methods and distance measures and identify those suitable for the considered scenario, and 2) we perform exhaustive experiments to compare and evaluate their performance under practical conditions. Our implementation is freely available[2]. This paper is structured as follows: In Section 2, related work and foundations for our experiments are identified. The experimental setup, including feature extraction methods and distance measures, is described in Section 3, and results are provided in Section 4. We conclude the discussion in Section 5.

[1]BSI TR-03137: Optically Verifiable Cryptographic Protection of non-electronic Documents (Digital Seal)

[2]`http://www.github.com/d-klein`

2 Foundations and Related Work

Two main areas of research are related to our setting: The first area is *(perceptual) image hashing.* The goal is to construct a perceptual hash function that takes an image as input, and maps it to a compact hash value such that perceptually similar images always map to similar values, and to different values if the images are perceptually different. To construct such a function, usually different image invariant features are extracted, for example by mapping to the frequency domain, by using the Radon Transform [2, 12], or by using Zernike Moments [7]. It is open whether these approaches are suitable here, since robustness of hashes is often measured only w.r.t. moderate digital modifications, whereas in our setting artifacts are quite severe (cf. Figure 1). Often, key-based hash functions (akin to MACs) are considered, where the input of the hash function is not only an image, but also a secret key. However, secure distribution of a symmetric secret key to all interested third parties for verification is practically impossible in our scenario.

The second area is face recognition and face verification [3]. Here, one usually tries to extract image invariant features from images with a specific focus on capturing the essential features of faces. In most methods, the size of the extracted features by far exceed the capacity of a barcode, and it is non-trivial to shrink the size of extracted features without severely affecting performance. In our setting we do not strive for face recognition, but only consider tamper-detection of documents; hence lightning, pose and expression will be the same in both the enrollment image and the scan; only the print-scan transformation introduces a measurable difference. Procedures that classify two different poses of the same person as different are no problem in our setting, and in fact desirable as this implies the ability to detect even slightest tampering with the document.

3 Experimental Setup

For our setup, we prepared sets of facial images ("enrollment images") and compute their hashes by a feature extraction method. Next, we print and scan the images, optionally pre-process the scans to enhance the quality, and compute hash values of the scans. Next, we compare hash values of the enrollment images and the scans by some distance measure.

3.1 Image Sets and Processing of Scans

We took two image sets of front images from the FEI Face Database[3] and the Color FERET database [6, 5]. Both image sets do not satisfy requirements set by ICAO for biometric face images – but in practice, these requirements are often not adhered to anyway. Thus from the FEI Face Database the subset frontal

[3]`http://fei.edu.br/~cet/facedatabase.html`

images, manually aligned, was used, and from FERET subset `fa` and subset `fb` those frontal images were selected that meet best the above mentioned requirements, especially w.r.t. lightning. Image contrast was then enhanced using IrfanView's[4] auto-enhancement, the images were printed on genuine Schengen-visa stickers using an off-the-shelf inkjet printer, and then re-scanned at 600dpi using a Kyocera TASKalfa 420i office scanner. After sorting out misprints, we were left with 368 images from FEI from 184 persons (one smiling, one with a neutral expression), and 768 images from FERET from 768 persons (one image per person).

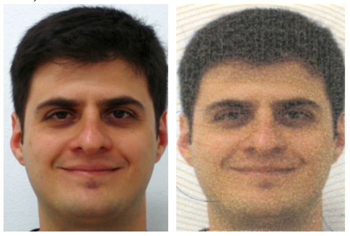
Figure 1: FEI, Original and Scan

	Accuracy	Size
$\mathrm{DCT}(8x8)^{\mathrm{mh}}_{\mathrm{raw}}$	0.922	500
$\mathrm{DCT}(11x11)^{\mathrm{mh}}_{\mathrm{raw}}$	0.921	1000
$\mathrm{DCT\ (blocks)}^{\mathrm{mh}}_{\mathrm{raw}}$	0.885	2000
$\mathrm{LBP}^{\mathrm{fb},\chi^2}_{\mathrm{raw}}$	0.961	1350
$\mathrm{LBP}^{\mathrm{binary},\chi^2}_{\mathrm{raw}}$	0.906	100

Table 1: Accuracy vs. Size (∼ Bytes)

An example of FEI is displayed in Figure 1: Printing and scanning introduces excessive noise, changes the aspect-ratio (due to a processing step in the visa-printing software) and adds a different background. Also, random guilloche lines reach over the image for security.

Having scanned the images, our experimental setup consists of the following steps: First, the face from the print-scan of the whole visa-sticker is extracted using a Haar cascade classifier, and automatically cropped to resemble the enrollment image. Optionally, some preprocessing of the images is applied to remove artifacts. Next, a feature vector is extracted from each image. Then, given one feature vector from an enrollment image, and one feature vector from a print-scanned image, the two feature vectors are compared by some metric to yield a score value.

Depending on the classification task, the score value is used either to classify the scanned image as being a print-scan of the enrollment image, or as a a different image (image verification), or the score value is used to identify the corresponding print-scanned image among a set of scans (image recognition).

Feature extraction methods have different resistance to artifacts. Given a scan, we consider the following additional steps before extracting features of scanned images: a) No preprocessing, b) a slight Gaussian blur (as this effectively removes visible guilloche lines without loosing too much sharpness), and c) the complete image pre-processing pipeline that has been proposed in [8]. After that, the image is resized to dimension NxN and converted to grayscale. The result is the input of the feature extraction algorithm.

[4]`http://www.irfanview.com`

3.2 Feature Extraction and Hash Comparison

The following feature extraction methods were used in the experiment. None depends on a symmetric key, and all are suitable to create very compact hash values.

Discrete Cosine Transform Switching from the spatial domain to the frequency domain is a technique frequently applied in various areas of image processing, and the DCT-II has been used for image hashing in the open-source tool `pHash`[5]. Similar, we take as the hash value the first $(8 * 8) - 1$ values (lower frequencies) of the DCT-II of a given image.

Radon Transform: RASH The radial variance based hash (RASH) was introduced by De Roover et al. in [2]. We directly apply their method, but do not threshold at the end and instead take the resulting real values as the hash.

Radon Transform: Method by Wu et al. For RASH, an approximation for the projection line is used to compute a hash inspired by the Radon transform. The method by Wu et al. [12] uses the radon transform more directly. Some implementation details are left out or are ambiguous in their description of the algorithm. In our implementation we use 40*10 blocks and employ the radon transform approximation of `skimage`[6].

Zernike Moments (ZM) Zernike Moments are image-invariant features that are especially robust to rotation, which makes them appealing in our setting. Their use was explored for image recognition in [9, 4]. **Pseudo Zernike Moments (PZM)** are adapted from Zernike Moments and less sensitive to noise [9]. In both cases, the absolute values of moments upto some chosen boundary are taken as the hash value.

Local Binary Patterns The use of local binary patterns (LBPs) for face recognition has been pioneered in [1]. Since taking raw LBPs as hash values is not suitable due to their large size, we adapt them here by reducing dimensionality with: (1) Histograms with fewer bins: We interpret each local binary pattern as an unsigned integer value. Then, in a somewhat ad-hoc approach, the number of bins in each local LBP histogram is reduced. (2) PCA: We take a training set of face images, compute the LBP hash for each image to get a high dimension feature vector, and perform principal component analysis (PCA) to compute a *static* PCA basis for LBPs of face images. For an arbitrary image (not related to the training set) the hash is computed by mapping its LBPs into PCA space.

Eigenfaces (EF) For LBPs, we mentioned how PCA can be used to reduce the dimensionality of a feature vector. The original Eigenface approach [10] used the pixel values directly as features to perform PCA. Whereas Eigenfaces are nowadays not competitive in arbitrary face verification [3], the appealing point of this approach is the low-dimensionality, and hence low storage requirements of the hash. Also, the weaknesses w.r.t. arbitrary face recognition (different facial expressions, different lightning etc.) become less important in our setting, since those parameters do not change for the print-scan compared to the enrollment image. To handle *both* image verification and image recognition we adapt PCA

[5]`http://www.phash.org`
[6]`http://scikit-image.org`

	FERET	FEC	combined		FERET	FEC	combined
$\mathrm{DCT}^{\mathrm{mh}}_{\mathrm{raw}}$	0.935	0.899 (0.967)	0.922 (0.944)	$\mathrm{RASH}^{\mathrm{mh}}_{\mathrm{raw}}$	0.641	0.609 (0.698)	0.620 (0.643)
$\mathrm{DCT}^{\mathrm{pcc}}_{\mathrm{raw}}$	0.951	0.891 (0.957)	0.929 (0.951)	$\mathrm{RASH}^{\mathrm{pcc}}_{\mathrm{raw}}$	0.704	0.802 (0.883)	0.715 (0.739)
$\mathrm{DCT}^{\mathrm{euc}}_{\mathrm{raw}}$	0.927	0.878 (0.957)	0.909 (0.936)	$\mathrm{RASH}^{\mathrm{euc}}_{\mathrm{raw}}$	0.482	0.242 (0.342)	0.388 (0.438)
$\mathrm{DCT}^{\mathrm{pcc}}_{\mathrm{gauss}}$	0.951	0.891 (0.957)	0.932 (0.949)	$\mathrm{RASH}^{\mathrm{pcc}}_{\mathrm{gauss}}$	0.721	0.813 (0.889)	0.732 (0.757)
$\mathrm{DCT}^{\mathrm{pcc}}_{\mathrm{tt}}$	0.928	0.832 (0.910)	0.893 (0.918)	$\mathrm{RASH}^{\mathrm{pcc}}_{\mathrm{tt}}$	0.831	0.633 (0.701)	0.750 (0.768)
$\mathrm{WU}^{\mathrm{mh}}_{\mathrm{raw}}$	0.747	0.840 (0.899)	0.767 (0.786)	$\mathrm{ZM}^{\mathrm{mh}}_{\mathrm{raw}}$	0.673	0.829 (0.867)	0.717 (0.729)
$\mathrm{WU}^{\mathrm{pcc}}_{\mathrm{raw}}$	0.876	0.883 (0.938)	0.872 (0.890)	$\mathrm{ZM}^{\mathrm{pcc}}_{\mathrm{raw}}$	0.746	0.774 (0.807)	0.742 (0.752)
$\mathrm{WU}^{\mathrm{euc}}_{\mathrm{raw}}$	0.724	0.829 (0.883)	0.745 (0.761)	$\mathrm{ZM}^{\mathrm{euc}}_{\mathrm{raw}}$	0.408	0.785 (0.837)	0.521 (0.537)
$\mathrm{WU}^{\mathrm{pcc}}_{\mathrm{gauss}}$	0.891	0.899 (0.946)	0.886 (0.901)	$\mathrm{ZM}^{\mathrm{pcc}}_{\mathrm{gauss}}$	0.755	0.764 (0.796)	0.748 (0.759)
$\mathrm{WU}^{\mathrm{pcc}}_{\mathrm{tt}}$	0.888	0.894 (0.954)	0.884 (0.902)	$\mathrm{ZM}^{\mathrm{pcc}}_{\mathrm{tt}}$	0.693	0.582 (0.628)	0.639 (0.654)
$\mathrm{PZM}^{\mathrm{mh}}_{\mathrm{raw}}$	0.531	0.772 (0.807)	0.596 (0.607)	$\mathrm{EF}^{\mathrm{mh}}_{\mathrm{raw}}$	0.711	0.867 (0.948)	0.709 (0.753)
$\mathrm{PZM}^{\mathrm{pcc}}_{\mathrm{raw}}$	0.697	0.750 (0.791)	0.688 (0.701)	$\mathrm{EF}^{\mathrm{pcc}}_{\mathrm{raw}}$	0.538	0.840 (0.946)	0.586 (0.616)
$\mathrm{PZM}^{\mathrm{euc}}_{\mathrm{raw}}$	0.247	0.736 (0.791)	0.391 (0.408)	$\mathrm{EF}^{\mathrm{euc}}_{\mathrm{raw}}$	0.464	0.818 (0.935)	0.549 (0.584)
$\mathrm{PZM}^{\mathrm{pcc}}_{\mathrm{gauss}}$	0.712	0.742 (0.780)	0.703 (0.715)	$\mathrm{EF}^{\mathrm{mh}}_{\mathrm{gauss}}$	0.837	0.867 (0.943)	0.780 (0.816)
$\mathrm{PZM}^{\mathrm{pcc}}_{\mathrm{tt}}$	0.634	0.595 (0.639)	0.592 (0.605)	$\mathrm{EF}^{\mathrm{mh}}_{\mathrm{tt}}$	0.784	0.736 (0.867)	0.776 (0.811)
$\mathrm{LBP}^{\mathrm{fb},\chi^2}_{\mathrm{raw}}$	**0.975**	0.938 (0.967)	**0.961 (0.970)**	$\mathrm{LBP}^{\mathrm{pca,mh}}_{\mathrm{raw}}$	0.811	0.791 (0.829)	0.792 (0.808)
$\mathrm{LBP}^{\mathrm{pca,pcc}}_{\mathrm{raw}}$	0.652	0.742 (0.772)	0.618 (0.624)	$\mathrm{LBP}^{\mathrm{pca,euc}}_{\mathrm{raw}}$	0.658	0.679 (0.712)	0.667 (0.674)
$\mathrm{LBP}^{\mathrm{pca},\chi^2}_{\mathrm{gauss}}$	0.962	**0.962 (0.986)**	0.961 (0.969)	$\mathrm{LBP}^{\mathrm{fb},\chi^2}_{\mathrm{tt}}$	0.958	0.913 (0.962)	0.944 (0.960)

Table 2: Evaluation for Image Recognition (Recognition Rate).

by selecting a small set of images to learn an overall *fixed* PCA basis for faces. These training images are not part of the set of enrollment images and scans. A hash for an image is then generated by projecting a query image into PCA space.

Distance Measures for Hash Comparison It is sometimes suggested (e.g. [13]) to further compact the hash by thresholding over the mean and comparing by Hamming distance. This however resulted in poor performance in our setting. Instead we consider the following distance measures to compare hashes: manhatten distance, euclidian distance, and peaks of cross correlation (PCC, c.f. [13]). These measures are not suitable for LBPs, since those consist of concatenated histograms. Hence, we use the χ^2 distance (c.f. [1]) and a weight of 1 for each region.

4 Experimental Results

First we consider the *image recognition problem* to get a notion on which feature extraction method works best in our context, which distance measure is optimal, and which pre-processing steps are best in reducing the artifacts introduced by printing and scanning. Given the hash of an enrollment image, we query the set of scans for the corresponding image. If correctly identified, we record a hit, otherwise a miss. The results are depicted in Table 2. Here experiments were run for the FERET dataset, for the FEC dataset, and both combined. For eigenfaces and PCA-based LBPs, we used 200 images from FEC as training data for FERET, 200 images from FERET as training data for FEC, and we split the combined set of 1136 images into 200 for training, and 936 for recognition.

Results in brackets denote the recognition score when one does not distinguish between images of the same person. For example if the hash originates from an image with a person smiling, and the lowest distance is to an image of that same person not smiling, we still count that as a hit. For preprocessing, *raw* denotes none, *gb* denotes a slight gaussian blur, and *tt* the pipeline of [8]. For distances, *mh* is manhatten distance, *euc* euclidian distance, and *pcc* peaks of cross correlation. For LBPs, *fb* means fewer bins, and *pca* size reduction by PCA. To keep the number of combinations feasible, we first identify the best distance measure for each method without preprocessing. Then taking the best beforming measure, we test w.r.t. the best performing preprocessing. Abbreviations for feature extractions are defined in Subsection 3.2.

Clearly, local binary patterns outperform any other feature extraction method. Somewhat surprising is that the very simple and straight-forward transformation into frequency domain by DCT-II is the second best feature. The method by Wu et al. performs adequate if an appropriate distance measure (PCC) is chosen. However their method based on the radon transform, as well all other methods, were apparently designed to cope with rather minor distortions compared to our setting, and perform much worse than LBPs and DCT.

Each of these feature extraction methods results in a different hash size. It would be desirable to measure their performance for equal hash sizes, but for a given method it is non-trivial to arbitrarily decrease the resulting size, and trivially increasing the size needs not to result in better performance. We investigate accuracy vs. size for the best performing methods LBP and DCT-II: Increasing the compact DCT-II is done (a) by taking 11x11-1 = 120 DCT-II coefficients instead 63, and (b), akin to JPEG compression, by dividing images into 4x4 blocks, computing 63 DCT-II coefficients for each block, and concatenating the result. To generate a smaller hash from LBPs, we thresholded each resulting local histogram by its mean to generate a binary stream, and used hamming distance to compare two hashes. Results are depicted in Table 1. Considering higher frequency information by taking more DCT-II information apparently picks up more noise and distortions, and does not lead to more accuracy. On the other hand, LBPs have much discriminatory power and there seems to be much potential to create a more compact hash.

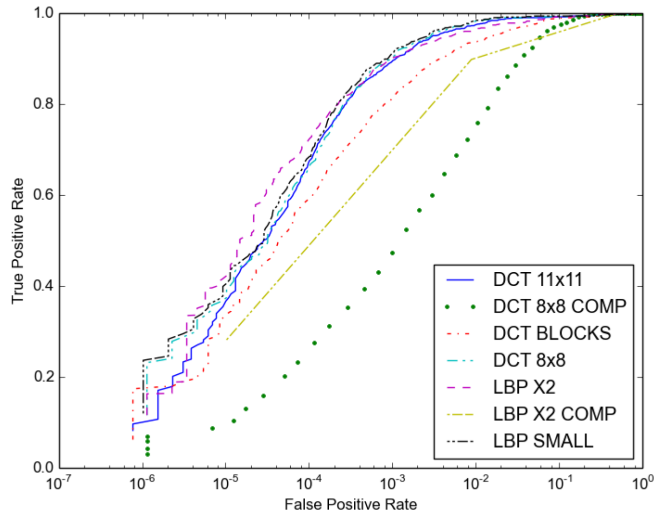

Figure 2: ROC for Image Verification

For all methods from Table 1, receiver operating characteristics (ROC) are depicted in Figure 2. It is not easy to turn the comparatively good results for image recognition into a good performance for *image verification*, i.e. distance values are such that it is possible to identify the corresponding image among a set by lowest distance, but difficult to derive a numerical global threshold for the image *verification* problem (given an arbitrary pair, decide whether they belong together or not). A powerful method is *One Shot Similarity* [11]: Given two arbitrary images x and y, just compare them w.r.t. to a set of images B for which it is known that each element of B shows a different person than both x and y, and train two classifiers based on this idea. Instead of training classifiers, we adopted this idea in a somewhat ad-hoc manner here (marked COMP in Figure 2), by splitting the combined test set into training and recognition data, and measuring distance between an enrollment image and a scan by first computing the minimal distance between the enrollment image and all training samples. The final distance is then the difference between the distance of scan and image minus the computed minimal distance to the training images. Unfortunately, this ad-hoc approach does not result in a better TPR/FPR ratio.

5 Conclusion and Future Work

We identified several potentially suitable hash methods for tamper detection. Security implications were only briefly touched by considering a set with smiling and non-smiling faces, and hence thorough evaluation of the security of the hashes is future work. We anticipate that OSS [11] can be adapted for better performance for image verification.

References

[1] T. Ahonen, A. Hadid, and M. Pietikinen. Face recognition with local binary patterns. In *Proc. ECCV*, volume 3021 of *LNCS*, pages 469–481, 2004.

[2] C. De Roover, C. De Vleeschouwer, F. Lefebvre, and B. Macq. Robust image hashing based on radial variance of pixels. In *Proc. ICIP*, pages 3,77–80, 2005.

[3] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, Univ. of Massachusetts, 2007.

[4] A. Khotanzad and Yaw H. H. Invariant image recognition by zernike moments. *Pattern Analysis and Machine Intelligence*, 12(5):489–497, 1990.

[5] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss. The FERET evaluation methodology for face-recognition algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(10):1090–1104, 2000.

[6] P. J. Phillips, H. Wechsler, J. Huang, and P. J. Rauss. The FERET database and evaluation procedure for face-recognition algorithms. *Image Vision Comput.*, 16(5):295–306, 1998.

[7] J. Revaud, G. Lavoué, and A. Baskurt. Improving zernike moments comparison for optimal similarity and rotation angle retrieval. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(4):627–636, 2009.

[8] X. Tan and B. Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. In *Analysis and Modelling of Faces and Gestures*, volume 4778 of *LNCS*, pages 168–182, oct 2007.

[9] C.-H. Teh and R.T. Chin. On image analysis by the methods of moments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 10(4):496–513, Jul 1988.

[10] M. Turk and A. Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, 3(1):71–86, January 1991.

[11] L. Wolf, T. Hassner, and Y. Taigman. The one-shot similarity kernel. In *Proc. ICCV*, pages 897–902, 2009.

[12] D. Wu, X. Zhou, and X. Niu. A novel image hash algorithm resistant to print-scan. *Signal Processing*, 89(12):2415–2424, 2009.

[13] C. Zauner. Implementation and benchmarking of perceptual image hash functions, 2010. Diploma Thesis.