IST-2002-507932

ECRYPT

European Network of Excellence in Cryptology

Network of Excellence

Information Society Technologies

D.WVL.12

Applications, Application Requirements and Metrics

Due date of deliverable: 31. January 2006
Actual submission date: 24. February 2006

Start date of project: 1 February 2004
Duration: 4 years

Lead contractor: Katholieke Universiteit Leuven (KUL)
Document responsible: Fraunhofer Gesellschaft e.V. (FHG)

Revision 1.0

<table>
<thead>
<tr>
<th>Dissemination Level</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PU  Public</td>
<td>X</td>
</tr>
<tr>
<td>PP  Restricted to other programme participants (including the Commission services)</td>
<td></td>
</tr>
<tr>
<td>RE  Restricted to a group specified by the consortium (including the Commission services)</td>
<td></td>
</tr>
<tr>
<td>CO  Confidential, only for members of the consortium (including the Commission services)</td>
<td></td>
</tr>
</tbody>
</table>
Abstract

While deliverable D.WVL.7 was focused on forensic tracking\(^1\), this document gives an overview of perceptual hashing and the research activities with ECRYPT. The focus of this document is applications of perceptual hashing techniques, their requirements and related metrics.

The introduction in chapter 1 starts with a short summary of deliverable D.WVL.7. The relation of this deliverable and deliverable D.WVL.6 is explained as well. This summary revises applications and problems that are relevant for this deliverable D.WVL.12. Relevant standardisation activities are addressed in chapter 2. Here, a brief overview of MPEG-7 and MPEG-21 is given and the relevance for perceptual hashing technologies and vice versa is described. This chapter and the following chapters are split in two sections: content identification and content authentication and verification to reflect the different applications for perceptual hashing technologies.

After the introduction to the different application scenarios with an view from standardisation, corresponding metrics are given in 3. Here, the basic foundations are revised. Chapter 4 gives an overview of the partners’ work. The summary in chapter 5 concludes this document with an outlook on future and potential research.

\(^1\)Forensic tracking includes perceptual hashing techniques for tracking content and content usage as well as watermarking for the same application scenarios. Furthermore, watermarking techniques can be applied in applications where additional information has to be embedded into the content itself. Among these applications is the identification of leaks or traitors.
Applications, Application Requirements and Metrics

Editor
Martin Schmucker (FHG)

Contributors
Michael Arnold (FHG),
Andreas Uhl (GAUSS),
Xuebing Zhou (FHG)

24. February 2006
Revision 1.0

The work described in this report has in part been supported by the Commission of the European Communities through the IST program under contract IST-2002-507932. The information in this document is provided as is, and no warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.
## Contents

1 Introduction

2 Applications and Application Requirements
   2.1 Standardisation .......................................................... 4
       2.1.1 MPEG-7 or the linkage between content and meta data ........ 5
       2.1.2 MPEG-21 or the linkage between content and related (user) rights ... 5
   2.2 Content Identification (as described in MPEG-21) ............... 6
   2.3 Content Authentication and Verification (as described in MPEG-21) ... 8

3 Metrics
   3.1 Content Identification .................................................. 9
       3.1.1 Hypothesis test .................................................. 9
       3.1.2 FAR and FRR .................................................. 10
       3.1.3 ROC and DET .................................................. 12
   3.2 Content Authentication and Verification ............................ 14

4 Contributions of the Partners
   4.1 Fraunhofer Institutes - FHG .......................................... 17
       4.1.1 Perceptual Hashing of Video .................................. 17
       4.1.2 Perceptual Hashing of Graphical Documents (Sheet Music) ........ 20
       4.1.3 Perceptual Hashing for Mutual Observation of Peers in Filesharing Networks ................................................. 24
   4.2 GAUSS ................................................................. 29
       4.2.1 Hashing Security: Attacks and Key-Dependency Schemes .......... 29
       4.2.2 GAUSS/Robust Visual Hashing using JPEG2000 .................. 46

5 Summary ............................ 60
List of Tables

4.1 Cross Comparison Results for VKJM .......................... 32
4.2 Cross Comparison Results for Modified (vkjm-all) VKJM ........ 35
4.3 Parameterized Filters Key Space ................................. 44
4.4 Sensitivity against JPEG 2000 compression: wlev6 used for feature extraction, wlev5 used for JPEG2000 compression ................................. 49
4.5 Sensitivity against JPEG2000 compression: identical coding options used for compression and feature extraction (wlev5) ................................. 49
4.6 Sensitivity against JPEG compression: wlev5 used for feature extraction ................................. 49
4.7 JPEG compression (lena512): different wlev used for feature extraction ................................. 50
4.8 Sensitivity against the removed flag ................................. 51
4.9 local attacks: different wlev used for feature extraction ................................. 53
4.10 different attacks/lena512: different wlev used for feature extraction ................................. 54
4.11 standard stirmark testsetting, lena512: different wlev used for feature extraction ................................. 56
4.12 standard stirmark testsetting, lena512: different wlev used for feature extraction ................................. 57
4.13 signature bits (including packet header data) ................................. 58
## List of Figures

1.1 general fingerprinting scheme .................................................. 2
1.2 A fuzzy region separates authentic and inauthentic content. ............. 2
1.3 Examples for fuzzy boundary between authentic and inauthentic multimedia content ................................................................. 3

3.1 Graphical representation of FAR and FRR in dependence of $f(\rho|H_0)$, $f(\rho|H_1)$ and threshold ......................................................... 11
3.2 $FAR(t)$ and $FRR(t)$ ................................................................. 12
3.3 Examples of the ROC curves ........................................................ 13
3.4 Examples of the DET curves ......................................................... 14
3.5 Example for boosting ................................................................. 15

4.1 Structure of video perceptual hashing algorithm .............................. 17
4.2 Block diagram of the algorithm .................................................... 18
4.3 Boxplot for different types of noise ($a = 0.95$) .............................. 19
4.4 Boxplot for MPEG2 compression in different bit rate ($a = 0.95$) ......... 19
4.5 ROC using different algorithms for $a = 1$ ..................................... 19
4.6 ROC using different algorithms for $a = 0.95$ ................................ 19
4.7 cumulative percentages of the total variation [31, 30] in the different individual features and in the combined feature vector ..................... 22
4.8 distribution of the fingerprint hamming distance for the original scores between different scores ......................................................... 23
4.9 use cases ................................................................................. 26
4.10 The main entities within the general architecture of CONFUCO are trusted third parties (TTPs) for user registration and identification, TTPs for content registration and validation, and the peers that consist of several components like user interface, local storage interface, content identification and P2P-networking. ......................................................... 27
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.11 VKJM Testing Results</td>
<td>33</td>
</tr>
<tr>
<td>4.12 Results for stable VKJM</td>
<td>36</td>
</tr>
<tr>
<td>4.13 Results for modified VKJM</td>
<td>37</td>
</tr>
<tr>
<td>4.14 VKJM attack results</td>
<td>39</td>
</tr>
<tr>
<td>4.15 Visual examples for the effectiveness of the attack</td>
<td>41</td>
</tr>
<tr>
<td>4.16 Key dependency test: Hamming distances between hashes generated with different keys.</td>
<td>42</td>
</tr>
<tr>
<td>4.17 Key dependency test: Hamming distances between hashes generated with different keys (based on a parameterized filter).</td>
<td>43</td>
</tr>
<tr>
<td>4.18 Hamming distances</td>
<td>44</td>
</tr>
<tr>
<td>4.19 Attack resistance of the key dependency schemes</td>
<td>45</td>
</tr>
<tr>
<td>4.20 Different coding parameters used for feature extraction, lena512</td>
<td>48</td>
</tr>
<tr>
<td>4.21 Testimage plane512 original and under attack</td>
<td>50</td>
</tr>
<tr>
<td>4.22 local attacks (plane and goldhill)</td>
<td>52</td>
</tr>
<tr>
<td>4.23 local attacks (houses)</td>
<td>52</td>
</tr>
<tr>
<td>4.24 local attacks (graves)</td>
<td>53</td>
</tr>
<tr>
<td>4.25 local attacks (surfside)</td>
<td>53</td>
</tr>
<tr>
<td>4.26 reconstruction of lena</td>
<td>59</td>
</tr>
<tr>
<td>4.27 reconstruction of graves</td>
<td>59</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Due to the mass of information that is stored, managed and distributed, efficient technologies are required that allow the reliable identification of content. While cryptographic hashing techniques have become the tool of choice for the identification of binary data like executables, multimedia content has different representations e.g. due to applied format conversion, compression or quality enhancing technologies.

Thus, cryptographic hashing is neither applicable for multimedia content for identification or authentication. Nevertheless, the potential applications for methods that are able to identify multimedia content are numerous. As a consequence, research and development on this technology is increasing. Coevally, technologies for the identification of multimedia content are included in ongoing standardization activities and upcoming standards like MPEG-7 or MPEG-21, e.g. as the so-called ‘Persistent Association Technologies’ (PAT).

Different application scenarios as well as the basics of perceptual hashing technologies and their difference to forensic tracking by applying watermarking techniques were already addressed in D.WVL.7 [53].

A general scheme, as given in [10] and already described in D.WVL.7 [53], is shown in figure 1.1 and involves the following operations:

- fingerprint calculation, which consists of
  - feature extraction and processing
  - perceptual hash modeling
- fingerprint matching, which consists of
  - database lookup
  - hypothesis testing

Perceptual hashing techniques, however, cannot only be used for content identification. Also, content authentication and verification is possible with these technologies, as already introduced in D.WVL.6 [52]. D.WVL.6 is focused on content authentication.
Figure 1.1: The general identification based on fingerprints involves two functional blocks [10]: First, the perceptual hash value (fingerprint) is calculated. Second, a database look-up retrieves one or more stored values. A following hypothesis testing verifies if a content has been identified correctly.

Multi-media data is perceived. Thus, humans do not perceive or notice certain types of content modifications. Again, cryptographic hash functions for authenticating multimedia data are restricted to application where content is not processed at all. Even format conversions are critical as the content modifications result in different hash values.

In contrast to cryptographic hash functions, perceptual hash functions are designed to overcome this drawback: Only manipulations, which change the content noticeably or considerably should affect the calculated perceptual hash function. Unfortunately, there is no well defined boundary between authentic and inauthentic data, which is exemplified in figure 1.2 (cf. [75]). Not only for some processing operations it is difficult to decide (automatically) if the result of an applied processing operation is authentic.

Figure 1.2: There is no clear boundary between authentic and inauthentic content. Authentic and inauthentic content are separated through a fuzzy region [75].

For example, a compression removes the details that are less significant for humans. In a sense, these details are perceptually less important. An attacker, however, might also be interested in removing these details. For example, persons’ faces in surveillance videos or the cars’ license plates might be blurred to prevent their identification. Furthermore, the
results of a compression algorithm, which introduces strong artefacts, can be perceptually more annoying than a removal of details. Figure 1.3 gives an example of such effects.

![Figure 1.3: Examples for fuzzy boundary between authentic and inauthentic multimedia content: The image in the middle is a (strong) compressed version of the version shown on the left. The effects of compression (compression artefacts) influence have a stronger visual influence on the perceived quality in comparison with to the effects of reducing noise (details). If one of the modified images is or both of them are inauthentic depends on the application scenario.](image)

As a consequence, perceptual similarity for identification and perceptual similarity for authentication are different ‘qualities’ of similarity. Especially for authentication, similarity has to be defined carefully. Thus, the applied perceptual hashing technique has to be designed in order to fulfill the individual requirements of the given applications scenario. The difficulty here is similar to the problems for authentication watermarks.

In the next chapter, applications and their requirements are summarized considering content identification as well as content authentication applications. Metrics for the evaluation of perceptual hashing algorithms are described in 3. An overview of the work of the partners is given in chapter 4.
Chapter 2

Applications and Application Requirements

In this chapter, we shortly summarize different applications and their relation to standards.

2.1 Standardisation

During the past years, a tendency is becoming more and more obvious: accessibility and usability. The term accessibility describes ‘the degree to which a system is usable by as many people as possible without modification.’ [74]. In contrast to accessibility, the term usability describes ‘how easily a thing can be used by any type of user’ [74]. The past shows that missing standards in a wider sense influences accessibility as well as usability. Different standardisation bodies therefore address the need unique standards to ensure accessibility and usability.

The Moving Picture Expert Group (MPEG) ‘is a committee of ISO/IEC that is open to experts duly accredited by an appropriate National Standards Body. On average a meeting is attended by more than 300 experts representing more than 200 companies spanning all industry domains with a stake in digital audio, video and multimedia. On average more than 20 countries are represented at a meeting’ [35].

Different standards have been established, including:

- MPEG-1, e.g. the standardisation of Video CD and MP3
- MPEG-2, e.g. the standardisation of Digital Television set top boxes and DVD
- MPEG-4, e.g. the standardisation of multimedia content distribution
- MPEG-7, e.g. the standardisation of meta data representation for audio and visual content
- MPEG-21, e.g. the standardisation of a ‘Multimedia Framework’

New standard lines have been started, specifically:
2.1.1 MPEG-7 or the linkage between content and meta data

Linking content and meta data is very important not only for content management applications. Whenever data has to be accessed or retrieved, two issues are important:

- content identification
- content description

Thus, content identification should be accomplished with an open standardized mechanism. Several open standards have been created for this purpose in the digital world. Recently, the ‘extended Markup Language’ (XML) is used to describe the properties of meta data. XML is also used in MPEG-7 to define the content descriptors for audio visual content. ‘MPEG-7, formally named ”Multimedia Content Description Interface”, is a standard for describing the multimedia content data that supports some degree of interpretation of the information’s meaning, which can be passed onto, or accessed by, a device or a computer code.’ [37].

Obviously, the purpose of MPEG-7 is content identification by using meta data. In other words, MPEG-7 wants to enable access to content through (human- or computer-) understandable descriptors. To achieve this, different descriptors have already been defined MPEG-7. For example, there are low level descriptors like colour or texture descriptors as well as high-level descriptors. Perceptual hashing techniques can be based on these existing descriptors or MPEG-7 can be extended by suitable descriptors if necessary.

2.1.2 MPEG-21 or the linkage between content and related (user) rights

Content identification is not only relevant for content management but also for rights management. As DRM is the digital management of rights, rights have to be represented in a digital format to be digitally manageable. Therefore, rights (operation based permissions) are granted for a specific object or content to a specific user.\(^1\) Licenses can have a strongly varying complexity, reflecting everything from simple to complex rights situations. Therefore, the language used for the description of rights should be able to model even very complex situations, which can appear easily when dealing with digital content (e.g. audio-visual material).

MPEG-21 [36] comprises several parts:

\(^1\)A general problem of DRM systems is the fact that they do not (yet) qualitatively distinguish between the different kinds of usage. For example copying for personal purpose and copying for friends or even unknown persons is represented as the same action in current DRM systems.
• Part-1: Vision, Technologies and Strategy
• Part-2: Digital Item Declaration
• Part-3: Digital Item Identification
• Part-4: Intellectual Property Management and Protection (IPMP)
• Part-5: Rights Expression Language
• Part-6: Rights Data Dictionary
• Part-7: Digital Item Adaptation
• Part-8: Reference Software
• Part-9: File Format

Directly related to perceptual hashing is part 3: Digital Item Identification. Within this part, ‘Persistent Association Technologies’ (PAT) are considered\(^2\). PAT includes perceptual hashing\(^3\) as well as watermarking techniques.

Use cases taken from MPEG-21 are summarised in the following sections.

2.2 Content Identification (as described in MPEG-21)

In the deliverable D.WVL.7 \([53]\), several application scenarios have been introduced. These application scenarios are now reviewed. The review considers the the Use Cases given in MPEG-21 (see \([36]\) and section 2.1).

The application scenarios given in \([53]\) are

• content identification
• automated search and music distribution
• computer aided collecting
• broadcast monitoring
• broadcast coverage measurement
• copy protection
• securing of pre-mastering items
• securing online content distribution services
• protection of physical goods

\(^2\)see also http://www.chiariglione.org/MPEG/working_documents/MPEG-21/pat/tr.zip

\(^3\)Within MPEG-21 perceptual hashing is called fingerprinting
As deliverable D.WVL.7 [53] considered both watermarking and perceptual hashing, only a subset of the applications are relevant or applicable here. For example, in application scenarios, which require a personalization of content, perceptual hashing technologies are not applicable.

In the last years, due to the increase of counterfeiting of physical products, new technologies have been invented for to protect physical goods against counterfeiting. In this area, technologies are developed that are related to perceptual hash functions of digital content: physical features are extracted and digitally processed to result in a digital identifier. Nevertheless, within WVL4 these applications are not considered.

As we focus on digital content, the application scenarios that are considered and investigated in existing or evolving standards are important. As the ‘Moving Pictures Experts Group’ (MPEG) [35] has become an important standardisation body, we review the application scenarios, which are related to the ‘Persistent Association Technologies’ (PAT). PAT are considered in MPEG-21 [36].

Within MPEG-21, the given use cases focus on:

- Rights and Content Management: Perceptual hashing technologies have to support rights and content management. Especially, when content is separated from its metadata a robust identification of the work is required.\(^4\)

- Audio Content Tracking and Reporting: Rights holders, service providers and stakeholders whose business is based on content ‘consumption’ require a reliable technology for usage reporting and ‘hit lists’ generation.

- Internet Audio Content Services: This is the classical scenarios where fingerprinting technologies are used to limit content distribution via the Internet.

- Anti-Piracy Investigation and Enforcement: This application scenario addresses the commercial distribution of content. Professional investigators should be supported by perceptual hashing technologies to automatically identify ‘such work’.\(^5\) Furthermore, watermarking technologies allow identifying and tracing the illegal sources.

- Value Added Services: Customers can benefit from content identification as content dependent services can be built upon content identification. These services include on-line purchase services, promotional content or incentives.

It is obvious that MPEG strongly focuses on the commercial applications related to audio visual content. Thus, it somehow neglects so far the non-commercial content, which has a cultural or historical value.

Summarized we can see that these application scenarios do not differ from the ones listed in [53]. Thus, the same requirements apply here (details can be found in [53]):

\(^4\)One has to be aware that perceptual hashing technologies require a prior registration before identification. This is not ‘compatible’ the copyright where each new work is protected without registration.

\(^5\)From a practical point of view this is highly debatable. Professional anti-piracy investigators typically do not have to identify work as this is labelled e.g. on the CDs in case of audio recording. Instead they require methods that reliably identify forgeries.
• discrimination power
• size of the perceptual hash
• robustness against processing operations
• complexity and performance of the hash calculation process
• complexity and performance of the hash retrieval process
• security against attacks

While for some the above listed application scenarios robustness is the most important criteria (e.g. for value added services), security becomes important, whenever a monetary or moral gain can be achieved by an attacker. This is for example the case, when an attacker achieves the distribution of manipulated content through a network protected with perceptual hashing technologies. In some sense, this is inverse to the definition of security for authentication: Here an attacker wants a manipulated objects to be recognized as the original content (cf. section 3).

2.3 Content Authentication and Verification (as described in MPEG-21)

For content authentication and verification, the reliability of recognizing a manipulated content should be very high. As this is significant for some application scenarios, e.g. not only when content is considered as evidence but also when customers should not receive manipulated content, MPEG-21 also considers content authentication and content integrity.

• Authentication and Integrity: ‘Watermarks can verify that the content is genuine and from an authorized source. Watermarks can also be used to assure the integrity content (i.e. that it has not been altered) for example by using ”fragile watermarks” or by embedding digest information in the payload.’

Obviously, MPEG-21 has not considered so far the potential of perceptual hashing technologies in this area.
Chapter 3

Metrics

For watermarking numerous publications address the problems of robustness and security. For perceptual hashing techniques, however, there is little comparative material available for robustness and very few for the security of the perceptual hashing techniques. In this chapter, we describe metrics and approaches to measure the robustness and the security of perceptual hashing algorithms. In section

3.1 Content Identification

This section focuses on statistical evaluation of the matching process for multi-media content identification. The definitions of hypothesis, false acceptance rate and false rejection rate are given. It is shown, how a system threshold can be determined using the different criteria with respect to applications and how a system can be evaluated in terms of its identification abilities.

3.1.1 Hypothesis test

Generally during the matching process of an identification system, perceptual hashes of the queried multimedia data are compared with perceptual hashes that are stored in a database. The case that the compared perceptual hashes are matched is defined as the hypothesis \( H_0 \), that is to say, that they are extracted from perceptually identical contents. The case of mismatching is defined as the hypothesis \( H_1 \). The matching process is deciding one of the two hypotheses – the hypothesis test. The following four possible situations can occur:

1. The hypothesis \( H_0 \) is accepted when the hypothesis \( H_0 \) is true
2. The hypothesis \( H_1 \) is accepted when the hypothesis \( H_0 \) is true
3. The hypothesis \( H_1 \) is accepted when the hypothesis \( H_1 \) is true

\(^1\)Different searching strategies have been purposed to limit the search space for reducing the "of dimensionality".
4. The hypothesis $H_0$ is accepted when the hypothesis $H_1$ is true

Situation 1 and 3 are correct decisions and situation 2 and 4 are incorrect decisions.

The next section shows the evaluation of a system in terms of errors occurring in hypothesis test.

### 3.1.2 FAR and FRR

In consideration of the binary character of perceptual hash, the number of mismatched bits $i$ normalized by the number of bits per perceptual hashes $n$ describes the distance between two perceptual hashes. It is called bit error rate (BER) and denoted as $\rho$:

$$\rho = \frac{i}{n}$$

(3.1)

where $i \in [0, 1, 2, \ldots, n]$ and $0 \leq \rho \leq 1$. The smaller the BER is, the higher the probability is, that the corresponding multimedia data contains perceptually identical contents.

Since $\rho$ is a random variable, the following two distributions play an important roll in the matching process: the distribution of $\rho$ results from comparison of perceptually identical contents and the distribution of $\rho$ resulting from matching between different contents. The first one implies the robustness of perceptual hashing. Due to content modification and formats conversion, perceptual hashes of similar contents can differ from each other. But this deviation should be small. The second one shows the ability of discriminability of perceptual hashing. BER between perceptual hashes of different contents should be large enough\(^2\). We denote $f(\rho|H_0)$ as the probability density function of $\rho$ under the condition that $H_0$ is true, as well as $f(\rho|H_1)$, if $H_1$ is true. Both of them are dependent on perceptual hashing algorithms and test materials.

Suppose that $\Gamma$ is the set of all possible occurrences of $\rho$. The space $\Gamma$ is divided into two disjoint subsets $\Gamma_{H_0}$ and $\Gamma_{H_1}$ which cover $\Gamma$ such that

$$\Gamma_{H_0} \cup \Gamma_{H_1} = \Gamma \quad \text{and} \quad \Gamma_{H_0} \cap \Gamma_{H_1} = 0$$

The hypothesis $H_0$ is accepted, if $\rho \in \Gamma_{H_0}$, or else the alternative $H_1$ is accepted. The probability that situation 2 in section 3.1.1 occurs, which is called false rejection rate (FRR) or a false alarm, can be expressed as:

$$FRR = P\{\rho \in \Gamma_{H_1}|H_0\} = \int_{\Gamma_{H_1}} f(\rho|H_0) \, d\rho$$

(3.2)

The probability of situation 4 in section 3.1.1, which is called false acceptance rate (FAR) or a miss, can be written as:

$$FAR = P\{\rho \in \Gamma_{H_0}|H_1\} = \int_{\Gamma_{H_0}} f(\rho|H_1) \, d\rho$$

(3.3)

\(^2\)It can be shown that the expected value of $\rho$ under the condition that $H_1$ is true is 50%, if bits of perceptual hashes are uniformly distributed.
$FAR$ is the probability of deciding $H_0$, when $H_1$ is true. It can also be expressed in terms of $\Gamma_{H_1}$ as:

$$FAR = \int_{\Gamma} f(\rho|H_1) \, d\rho - \int_{\Gamma_{H_1}} f(\rho|H_1) \, d\rho$$

$$= 1 - \int_{\Gamma_{H_1}} f(\rho|H_1) \, d\rho$$

$$= 1 - P\{\rho \in \Gamma_{H_1}|H_1\}$$

(3.4)

So $P\{\rho \in \Gamma_{H_1}|H_1\} = 1 - FAR$ is the probability of detection and referred to as the power of the matching process.

Figure 3.1 represents graphically the relationship of FAR and FRR in dependence on $f(\rho|H_0)$, $f(\rho|H_1)$ and threshold. The red curve and blue curve depict the conditional probability density function $f(\rho|H_0)$ and $f(\rho|H_1)$. The threshold is marked by the black line, which divides the whole occurrence space into $\Gamma_{H_1}$ and $\Gamma_{H_0}$. FAR equals the dark red area between the threshold, $f(\rho|H_1)$ and $R-$axis. The blue area describes $FRR$. The critical region is where $f(\rho|H_0)$ and $f(\rho|H_1)$ overlap. Naturally if there is no overlap, a perfect threshold in non-overlapping region exists so that the matching performance is error free and both $FAR$ and $FRR$ equal 0.

![Figure 3.1: Graphical representation of FAR and FRR in dependence of $f(\rho|H_0)$, $f(\rho|H_1)$ and threshold](image)

The overlapping of $f(\rho|H_0)$ and $f(\rho|H_1)$ causes erroneous matching process. In this case a small threshold reduces the region $\Gamma_{H_0}$, so $FAR$ is suppressed, however, $FRR$ increases. $FAR$ and $FRR$ are functions of threshold $t$. Figure 3.2 depicts $FAR(t)$ curve and $FRR(t)$
For practical systems, it is difficult to model \( f(\rho|H_0) \) with a mathematical model. However, \( f(\rho|H_1) \) can be modeled with Gaussian distribution, if comparisons involve only statistically independent contents. But this condition may require a large amount of test materials. So in practice, empirical tests are needed to estimate both of these conditional distributions. Depending on these distribution and applications, the threshold of identification/verification system can be determined.

Practical systems have different requirement on \( FAR \) and \( FRR \). For example, in a monitoring system a high acceptance rate is needed so that fewer data are omitted, whereas, in a verification system lower \( FAR \) is required. Depending on different application scenarios, different criteria can be chosen to optimize the "decision" threshold \( t \). For example, the Neyman-Pearson Lemma criterion keeps the \( FRR \) less than or equal to some prechosen value and maximizes the probability of detection for this \( FRR \). The Bayes decision theory makes use of a systematic procedure of assigning costs to each hypothesis and then minimizing the total average cost. A special case of the Bayes decision theory is that \( FAR \) and \( FRR \) are even appraised. The \( t \) is chosen so that \( FAR(t) = FRR(t) \). The intersect of \( FAR(t) \) and \( FRR(t) \) is also called equal error rate (EER) (see the green point in figure 3.2).

![Figure 3.2: FAR(t) and FRR(t)](image)

### 3.1.3 ROC and DET

The *receiver operating characteristics* (ROC) curve and the *Detection Error Tradeoff* (DET) curve are widely utilized in evaluation of identification power. **ROC** is a plot of the probability of detection \( 1 - FAR \) for the false alarm probability \( FRR \).
It was developed in the 1950’s as a by-product of research that addresses radio signals contaminated by noise [26]. It efficiently describes the detection performance of a detector. Both $FRR$ and $1 - FAR$ are integrated over the area $\Gamma_{H_1}$, which is a function of the threshold $t$. Figure 3.3 represents some examples of the ROC curves. The ideal matcher has a ROC curve like the fat blue curve, because the probability of the detection is always 1. In this case there is no overlapping of $f(\rho|H_0)$ and $f(\rho|H_1)$. If a perceptual hashing algorithm is so inefficient that $f(\rho|H_0)$ and $f(\rho|H_1)$ completely overlap, the matching is random and the resulting ROC curve is like the red one. The lilac curve implies better performance of detection than the red one, since its probability of detection is higher at the same probability of false alarm. In addition, the nearer a ROC curve to the blue one is, the better its detection performance is. As we move from right to left along the ROC curves $\Gamma_{H_0}$, the area of region $H_0$ is increased.

**DET** curve is a plot of $FAR$ in dependence on $FRR$. It describes the relationship of $FAR$ and $FRR$. In contrast to ROC curve, DET curve is monotonic decreasing. A error free matching system has a DET curve locating on the $FAR$- and $FRR$- axes. The sum of $FAR$ and $FRR$ in the worst matching system is always 1 like the red curve in figure 3.4. The lilac curve is close to the $FAR$- and $FRR$- axes, it indicates a better identification performance than the red one.

ROC and DET curves are efficient methods to evaluate and compare perceptual hashing algorithms. An algorithm is better than another one, when its ROC curve is above another one or its DET curve is under another one. However, it is difficult to assess these performance if their ROC or DET curves intersect. Area under curve (AUC), which is defined as the area

![Figure 3.3: Examples of the ROC curves](image-url)
between the ROC or the DET curve to x-axis, is applied to summarize general ability of identification. The AUC gives a statement of overall performance of identification. An AUC of the ROC curve oughts to be close to 1 and that of the DET curve oughts to be close to 0.

3.2 Content Authentication and Verification

In contrast to watermarking (e.g. [39, 41, 40, 75]), only few research was published dealing with the security of perceptual hash functions in general. In [60, 61] the security of visual hash functions was considered. This publication analysed the so called visual hash function (VHF) as proposed by Fridrich [22]. This algorithm will not be described in detail here. Instead we will outline the security analysis, which was presented in [60, 61].

Second preimage resistance or weak collision property is a general problem for perceptual hashing technologies. Therefore, in [60, 61] the ‘modified weak collision’ property is defined as follows:

A robust hash function hash is said to satisfy the modified weak collision property if given $x$ and $hash(x)$ it is not feasible to find a $y$ such that $hash(x) = hash(y)$ and $x$ is significantly different from $y$.

---

$^3$Given an input $m_1$, it should be hard to find another input, $m_2$ (not equal to $m_1$) such that $hash(m_1) = hash(m_2)$ (see [53]).
The general problem however still remains: how to define ‘significantly’ different? Obviously, a formal definition is not straightforward as it is application specific.

From a descriptive point of view, perceptual hashing maps a subspace of the original domain, which embraces all possible contents, to a point in the target domain, which consists of the resulting fingerprints. Therefore, in [60, 61] this subspace is considered as a cluster around the original input. The trade-off between the robustness against processing operations and the security of the authentication is reflected in the size and the shape of the cluster. The interesting aspect shown in [60, 61] is the fact that this clustered can be learned.

To achieve this the authors propose a method based on boosting. Boosting itself is a relative new idea. First publications go back to the 90s [19]. The most commonly used implementation is the AdaBoost algorithm. Like the other boosting methods AdaBoost combines multiple weak classifiers (weak learner) to a stronger classifier (strong classifier). AdaBoost achieves relatively good results and is therefore is commonly used nowadays.

Detailed information about AdaBoost and other methods can be found e.g. in [19, 62, 20]. So we won’t go into details here. Nevertheless, we try to exemplify its principle shortly. As shown in 3.5 a simple classifier might not produce good results when applied to certain problems. A good combination of multiple classifiers, however, allow to separate the input space in a way that the results of this combined classifier strongly outperforms each individual classifier. To achieve this, a final hypothesis $H$ is a weighted majority vote of the given individual weak hypotheses. The weights are determined in a training phase.

![Figure 3.5](image)

Figure 3.5: In a training phase weak learners (left image) are combined to achieve better results (right image). The example is taken from [42].

The approach as described in [60, 61] ‘learns the statistical model of the hash function. As the system under test is block based [22], for each block the method can be applied independently. As a solution to this problem the authors suggest to treat the block not independently from each other.

In [27] a metric was introduced that is the differential entropy. This metric is used to quantify the amount of randomness in and to study the security of perceptual image hash functions in a mathematical framework. ‘The higher the differential entropy of the hash value, the higher the randomness and the larger the number of exhaustive searches required to forge the hash value $h$’ [27]. Different schemes are analysed and compared in terms of robustness
and security. As a robustness metric ROCs (see 3.1) are considered. The trade-off between robustness and security is discussed. This work is further elaborated in [70].
Chapter 4

Contributions of the Partners

4.1 Fraunhofer Institutes - FHG

The research of the Fraunhofer group, which consists of the Fraunhofer Institute for Computer Graphics Research (IGD), the Fraunhofer Institute for Integrated Circuits (IIS) and the Fraunhofer Institute for Integrated Publication and Information Systems Institute (IPSI), within WAVILA WVL4 (perceptual hashing), focused on the improvement of existing techniques, the development of new techniques and applications in which perceptual hashing can give a valuable contribution.

4.1.1 Perceptual Hashing of Video

Different approaches exist for extract perceptual hashes from video content. Generally, the incoming video content is firstly decoded into individual frames. Then, the individual frames are divided into small blocks. The statistical characteristics such as mean, variance, colour descriptor etc. are extracted. It is also called block processing. The resulting values should be processed and compressed in order to get robust and compact video perceptual hashes. Figure 4.1 shows a general structure for video perceptual hashing algorithms.

![Figure 4.1: Structure of video perceptual hashing algorithm](image)

The features of video signal can be extracted purely from spatial information like the algorithms in [13], [11] and [49], or from spatio- temporal information like [76] and [54]. The algorithms that extract features from individual frames offer good robustness to manipulations based on common image processing. However, they need a process to vanquish the
highly temporal correlation of the video signal. The video perceptual hashes extracted from the spatio-temporal changing have less redundancy and good robustness, in particular for rapidly varying video.

**Video perceptual hashes using differential block similarity** In [54] the spatio-temporal differentiation of mean luminance is calculated to get high frequency components of video clips. This method performs well both in discernibility and in robustness to most video manipulations. However, its effectiveness is significantly impacted by noise including compression noise or slowly varying videos. In [77] and [78], the structure of Philip’s algorithm is kept and the interframe similarity is extracted to enhance the robustness of the perceptual hashes. Figure 4.2 depicts the block diagramm of the algorithm.

Each frame is divided into $M + 1$ blocks. The luminance of pixel $j$ in the block $i$ of frame $n$ is denoted as $x(j, i, n)$ with $i \in [1, \cdots, M + 1]$. The similarity of temporal consecutive blocks is $S_{i,n}$ with:

$$S_{i,n} = \sum_j x(j, i, n) \cdot x(j, i, n - 1) \quad (4.1)$$

The resulting hashes $H_{i,n}$ are the sign of the spatio-temporal difference of $S_{i,n}$. A block diagram, describing the algorithm, is shown in figure 4.2.

$$H(i, n) = \begin{cases} 
1 & \text{if } (S_{i,n} - S_{i+1,n}) - a \cdot (S_{i,n-1} - S_{i+1,n-1}) \geq 0 \\
0 & \text{if } (S_{i,n} - S_{i+1,n}) - a \cdot (S_{i,n-1} - S_{i+1,n-1}) < 0
\end{cases} \quad (4.2)$$

**Analysis and evaluation** Philips perceptual hashing algorithm [54] in Matlab was implemented as a reference implementation and compared with the proposed algorithm. In figure 4.3 and 4.4 the boxplots of BER for different videos for noise contamination and MPEG2 compression are represented (BER is denoted as rate of mismatched bits between hashes of
the original videos and those of their manipulated videos). Comparing the two algorithms, the median as well as the range of BER are strongly suppressed in the case of noise. For MPEG2 compression the BER decreases notably. The algorithm using similarity has better performance in robustness to video processing (the detailed results are shown in [77]).

![Boxplot for different types of noise (a = 0.95)](image1)

![Boxplot for MPEG2 compression in different bit rate (a = 0.95)](image2)

The empirical tests show that the algorithm using similarity improves the identification performance. Figure 4.5 and 4.6 show ROC curves of Philips algorithm and the proposed algorithm in the case of Gaussian noise. Generally, the dashed lines are under the solid lines. Therefore the algorithm using similarity has better identification performance. For $a = 1$ the enhancement is very significant (figure 4.5).

![ROC using different algorithms for $a = 1$](image3)

![ROC using different algorithms for $a = 0.95$](image4)
Conclusion and Outlook  The video perceptual hashes using differential block similarity has improved robustness of perceptual hashing without any suppression of discriminability power. However, the perceptual hashing algorithms based on the spatio-temporal changing has trouble with video clips consisting of still images or slowly varying videos. In these cases, the spatial information will be necessary to get more information from video clips. Especially for slowly varying video clips or still video clips, the perceptual hashing algorithms can be combined with an algorithm based on the spatial information like [11].

4.1.2 Perceptual Hashing of Graphical Documents (Sheet Music)

The new possibilities of digital storage and digital distribution of sheet music provide opportunities as well as dangers[28, 29]. For sheet music however, so far only few passive protection methods have been developed. These developments are limited to watermarking [47, 48, 63, 64] and they suffer some practical limitations [8, 9] especially when the application scenario is IPR protection.

In [34] a first method for the identification of sheet music based on the graphical representation is described. To extract features from the staves of a music sheet these staves have to be identified and extracted. For this a simple method based on the horizontal projections (histograms) is used in some OMR publications. The known disadvantage of this methods are distortions like rotation or bending. These affect the quality of the staff line detection as a threshold is applied to identify individual staff lines.

After the detection of the staves, individual staff lines are removed so that only music symbols remain. From these remaining symbols, the features are extracted. Potential features are

• statistical (two dimensional) pixel distributions like moments,
• properties of connected components or other segmentations of the musical symbols, or
• properties of individual symbols like the center, the area, the width of the bounding box and the relation of the black to white pixels inside the bounding box, or
• simple graphical features like envelopes or projections.

The article considered the following features:

• Envelopes are known from signal processing. They are defined as the upper or lower bound of a signal, if this signal is symmetrical. They are retrieved by scanning the remaining symbols from above and below. The position of the first black pixel determines the position of the envelope.
• Projections are the sum of the black pixels along a scanline. Only the horizontal and vertical direction of the scanline are considered. So the horizontal projection has the same height as the original stave extraction and the vertical projection has the same width as the stave extraction.
To reduce the dimension of the input vectors Principal Component Analysis (PCA) was applied. PCA is a central tool in data analysis. The application areas range from neuroscience to computer graphics as the PCA is simple to use. Furthermore it is a non-parametric method that allows the extraction of relevant information from noisy or confusing (high-dimensional) data sets. 'With minimal additional effort PCA provides a road map for how to reduce a complex data set to a lower dimension to reveal the sometimes hidden, simplified dynamics that often underlie it' [68].

The PCA is based on the sample mean

$$M = \frac{1}{N} \sum_{k=1}^{N} X^k$$ (4.3)

and the sample covariance matrix

$$\Sigma = \frac{1}{N} \sum_{k=1}^{N} (X^k - M)(X^k - M)^T$$ (4.4)

where $X^k, k = 1, N$ are the samples.

The aim of PCA is to find a good representation (approximation) of the data. For this, redundancy is removed: a basic transformation is applied, which transforms the input data $X$ into a new coordinate system $Y$ where each variable co-varies as little as possible with other variables. The goal can be described as [68]:

'Find some orthonormal matrix $P$ where $Y = PX$ such that $S_Y \equiv \frac{1}{N-1} YY^T$ is diagonalized. The rows of $P$ are the principal components of $X$.'

The measure to identify the numbers of new features is the amount of variance ‘covered’ by the selected Principal Components (PCs). This is called the ‘cumulative percentage of the total variation’ [31, 30] and is defined as:

$$t_m = \frac{100}{p} \sum_{k=1}^{m} l_k$$ (4.5)

where $l_k$ is the variance of the $k$th PC and $p$ is the sum of the variances of all PCs.

The application of the PCA was split in two steps:

1. The principal components of the individual feature vector are calculated to reduce their size. The normalized size of the staff lines was 2144 pixels. As shown in figure 4.7, the upper and lower envelopes can be transformed into and reduced to a feature vector with a dimension less than 200 components, which still contains more than 90% of the total variation. Similarly, the projections can be transformed and reduced. However, the horizontal projection seems to be better suited for this.

2. A combined feature vector consisting of the lower and upper envelopes and the horizontal projection is created. On this combined feature vector the PCA is applied to result in a reduced feature vector containing most information available in the (already reduced)
input features. As shown in figure 4.7 after this second step the resulting features contain more than 80% of the total variance. Thus a feature vector with the dimension of 128, which contains more than 70% of the total variance of the initial input data, can be identified.

A simple thresholding method converts the resulting feature vector \( f_{\text{combined}} \) into a binary vector \( \text{hash}_{\text{perceptual}} \):

\[
\text{hash}_{\text{perceptual}}[i] = \begin{cases} 
0 & \text{for } f_{\text{combined}}[i] \geq 0, \\
1 & \text{for } f_{\text{combined}}[i] < 0 
\end{cases} \quad (4.6)
\]

Although this perceptual hashes for one or more staves of different music scores might be equal, by connecting the fingerprint of all staves of one music score, we are able to identify this music score as shown in the next section.

**First results** For evaluation 9000 music scores were selected. First the probability for each bit is investigated. The analysis showed that

\[
P(\text{hash}_{\text{perceptual}}[i] = 0) = P(\text{hash}_{\text{perceptual}}[i] = 1) = 0.5 \quad (4.7)
\]
Figure 4.8 shows the hamming distances for the perceptual hash values calculated for music scores containing twelve staves. Thus the overall perceptual hash value consists of 1536 bit. As shown, the hamming distance of this overall perceptual hash value between different music scores is almost Gaussian distributed. However, we found that there are scores which are very similar (individual staves are the same). Therefore there are some outliers which have a smaller hamming distance, which is indicated by the red ellipse in figure 4.8.

Figure 4.8: distribution of the fingerprint hamming distance for the original scores between different scores

**Conclusion and outlook** The present work shows our first steps in the development of a perceptual hashing methods for the identification of music scores based on graphical representation. The availability of such an algorithm provides new possibilities for applications in the areas of content, rights and meta data management.

Further work investigates the robustness and discriminability of the implemented algorithm. So far, the developed algorithm shows good capabilities for the identification of sheet music. A detailed analysis of its performance, including its robustness especially against warping, will be available soon and published elsewhere.

The next steps will especially address a more sophisticated approach for the feature reduction. As well known in other areas, e.g. in biometrics [4], the PCA is not the optimal method for the identification of suitable features. Therefore, the possibilities of the Fisher Discriminant Analysis (FDA) [18] or the Distortion Discriminant Analysis (DDA) [7] will be analysed.
4.1.3 Perceptual Hashing for Mutual Observation of Peers in Filesharing Networks

In the articles [66, 65] the requirements for a P2P-framework that overcomes the previously described drawbacks are investigated. The proposed architecture is a framework for the legal distribution of commercial and non-commercial content via P2P networks. It supports a wide-range of business models ranging from shareable (promotional) content to DRM protected commercial content, ensuring legal exchange without centralized content usage controls. The proposed framework exploits technological potentials while at the same time maximizing its usability and attractiveness to users. It ensures that consumers act in a legally acceptable manor and any illegal infractions will be flagged.

The system achieves this through a process where each peer observes the peers it is exchanging content with thus increasing the probability of identifying infractions. In addition to addressing the requirements of consumers, content owners and distributors alike, this framework can also incorporate technologies that increase its attractiveness to users by providing additional services like collaborative filtering. In this way users are assured that the content they are accessing is both legal and of commercial quality.

Background  The content industry spends a lot of effort in building awareness of file-sharing illegality to users. Also, it is in favor of protecting valuable commercial content with restrictive DRM technologies which has the side effect of deterring potential customers for purchasing content. Some artists do not concern themselves with illegal file-sharing of their content nor about the users’ rejecting reaction to DRM and are instead only interested in the promotion of their content. Thus these stakeholders need a legal platform for the promotion of their non-commercial content. Additionally this platform should also support a wide range of business models. One has to be aware that the focus of most artists is on the legal commercial and non-commercial distribution of their content not on usage control which is not in their best interests.

DRM-protected content has to compete with illegal distribution as both provide access to relatively the same content. There are always ways to access unprotected content - at least the analogue hole. This unprotected content can easily be distributed via the Internet as content distribution cannot be controlled (completely) on the Internet [5]. Thus legally offered material must always compete with illegally accessible content.

As a consequence, a legal platform that only allows legal exchange of content without limiting content usage itself is required. Ideally consumers should not experience any limitation to the content usage in such a system. In contrast to existing "grown" file-sharing systems the content exchange must be on a legal basis. This means that either the content is exchanged legally or any misuse is identified and traced back to the offending user.

The importance of these two requirements - a legal distribution platform neither limiting content usage nor its consumption while supporting a broad range of business models - has up to now not been considered adequately by content owners, providers, and distributors; nevertheless these are the main requirements upon which the presented framework is based.
**Architecture**  Considering the huge volume of data transferred when downloading audio-visual content, a centralized distribution solution will sooner or later be a bottleneck in the dissemination of content. Distributed systems and particularly P2P-systems allow the transfer of storage and network costs to customers. Other capabilities of P2P systems include reliability, scalability, and performance [6].

Each framework for content distribution has to address specific criteria for success: first, content distribution within a P2P-system must not infringe on IPR. Second, the usage of content distributed within the network must not interfere with traditional content utilization.

The first requirement for any architecture for content distribution is in itself challenging: the potential misuse of content exchange infringing on IPR. A perfect distribution solution will not allow the unauthorized distribution of content.

A truly perfect distribution solution however is only possible if it runs on a trusted device that is not under full control of the user (cf. [1]). Unfortunately for content providers this level of control cannot be achieved as users will neither accept the expensive of such solutions nor will they spend additional money for systems with reduced functional value. The optimal solution considers the protection of IPR, the functional loss and monetary costs for consumers. These requirements can be met by enforcing the customers’ liability in the case of misuse which can be implemented with the use of two strategies in tandem:

- Technology must be used within a distribution framework that is able to identify users and the content that is distributed. However under this strategy the identification of content must not be limited to cryptographic hashes. Content based identification, also known as fingerprinting technology, is also mandatory [2, 3, 25, 55]. Combined with “black lists” and “white lists” exchanged content can be limited to authorized content.

- Social issues like community affiliation strongly affect users’ behaviors within the group, therefore building a user community with adequate rules is significant for the success of the system.

The typical use cases have to be analyzed as shown in figure 4.9. A User can have two roles:

1. As a Content Owner the user inserts content in the P2P-distribution framework. Additionally he can revoke the right to distribute content within the distribution framework.

2. As a Content Consumer the user downloads or exchanges content and “unbags” it from the distribution framework. The last use case is especially important for the usability of content distributed within the network. Content migration from the P2P-system to the outside should be as easy as possible.

For ensuring the authorized distribution of content two strategies have so far been implemented:

- The benefits of a DRMS are limited, and restrictive DRMS has the disadvantage that content usage is impeded which drastically lowers the interest of consumers. Thus users will not accept DRMS solutions and content owners will not reach enough consumers.
Fingerprinting and watermarking technologies are considered to be passive protection technologies. Up until now these technologies required an external control entity which analyzes the data exchanged within (similar to a police man observing traffic speed) [3, 12, 1].

The required control unit imposes major obstacles and thus P2P-network which integrates fingerprinting and cryptographic hash technologies in each peer is proposed. In this type of P2P-network, each peer acts as the previously described control instance. Illegal content exchange is only possible within a group of "traitors". As these groups also exchange content with other peers, their risk of being identified is very high and thus these users will likely use other solutions to exchange content illegally.

The simplified architecture of CONFUOCO is shown in figure 4.10 and consists of several sub-systems. Each peer has a user interface which controls the login to the P2P-network and the content exchange. User registration and identification is managed by trusted third parties who also manage and validate content exchange.

During the registration process the User is authenticated by the UserRegistration TTP. This could be done for example through Internet Service Providers (ISP) where the User is already registered, or he could be requested to enter some personal information (e.g. address and phone number). In the second case, this information should be checked for credibility and validated with a confirmation letter sent to the stated address.

Afterwards, a unique identification number is generated as UserID and the User can choose a pseudonym or nickname for identification within the P2P network. The User is requested to set an initial password and the certificates to identify the TTPs are stored on her local

---

1 Traitors refers in this context to people intending to misuse this system for illegal content exchange.
2 It is important that the User can choose a personal name for his “virtual personality” in the community. This name can reflect some personal attitudes or help other users to visualize something with the User’s ID (as it is easier for a user to remember than a number).
The main entities within the general architecture of CONFUOCO are trusted third parties (TTPs) for user registration and identification, TTPs for content registration and validation, and the peers that consist of several components like user interface, local storage interface, content identification and P2P-networking.

The detailed user data is stored at the UserRegistration TTP and only the pseudonyms (UserID and nickname) are submitted to the UserIdentification TTP and added to the list of valid users. During transactions in the P2P network the UserIdentification TTP simply has to check if the pseudonym of a User is in the list of valid users and if the password provided by the User is correct.

Only registered content can be exchanged within the P2P network. If a User wants to register new content he first submits the fingerprint (or hash value) of the new content to the ContentValidator TTP to verify that the content is not currently registered (on either the black or white lists).

If the validation process is successful the User submits the new content to the ContentRegistration TTP. This TTP calculates the fingerprint and hash value of the content and verifies that the content is valid.

The calculation of each content identifier depends on the content type. For unencrypted content the fingerprint and the hash value can be calculated and used for content identification, whereas for encrypted content (e.g. DRM protected audio files) only the cryptographic hash value is available.\(^3\)

Following these steps the content can be registered and detailed content data (UserID, timestamp, fingerprint and hash value of the content, optionally: meta-data and validity period) is stored at the ContentRegistration TTP.

Only the fingerprint and the hash value of the content (optionally meta-data and validity period) are submitted to the ContentValidator TTP and added to the white list of sharable content. The User receives a license containing the sharing permission, the identification of the content and the ContentRegistration TTP registering the content.

\(^3\)Encrypted content is considered as a binary large object (‘blob’).
During transactions the ExchangeValidator simply checks that all content exchanged is registered as sharable using the ContentValidator.

The P2P-Client is the interface between the users and the P2P-system. It allows users to upload new content, exchange it and transfer content out of the P2P system (e.g. to other devices).

- The **UserInterface** manages communication with Users such as login to the P2P-network or browsing for new content. It also provides a file manager for the insertion of new content and the transfer of existing content out of the P2P system.

The P2P-client’s repository is represented by an ordinary directory of the file system. Users can therefore copy selected content in and out of the P2P-system within an easy to use interface which results in a simple transfer of the content file within the file system.

No distinction between encrypted objects and unencrypted content is necessary here. The main advantage is that unprotected content can easily be copied to and from other directories, hard discs, or other devices.

- The **LocalStorageInterface** observes changes in the repository directory not caused by a file download. When new content is added a fingerprint value is calculated and the TTP responsible for validation is contacted. Depending on the result of the validation different actions are initiated.

  If the content is

  - already registered and sharable: no limitations apply.
  - already registered and not sharable: it is transferred to the QuarantineWard.
  - not registered: the User is asked if he wants to publish it. If so, the content is uploaded to the ContentRegistration TTP and registered to this User.

Encrypted (DRM protected) content can only be registered by particular users known as Content Owners.

- The **ContentID** component calculates content identifiers depending on the content type. While for unencrypted content the fingerprint and the cryptographic hash value can be calculated, for encrypted content (e.g. DRM protected audio files) only the cryptographic hash value is available.

- The **QuarantineWard** temporarily stores content which must not be shared. This allows the User to delete the files or move them to another folder or device.

- The **MagicTrunk** implements the P2P-functionality like content search and exchange. It initializes the calculation of fingerprints or hash values for the exchanged files.

  Each communicating peer (both the sender and receiving peer) transmits its calculated content identifiers to the ExchangeValidator TTP. This prevents peers from receiving illegal content while it allows the identification of peers illegally transmitting content.
Thus each peer observes the communication which increases the identification of manipulated peers drastically.\footnote{As this information does not directly identify the user it can be used for example as for alternative fee distribution models.}

4.2 GAUSS

The following subsections summarize the work done by the GAUSS-Salzburg group, on the one hand focussing on attacks against robust visual hashing schemes proposed for authentication purposes (see [44] for the corresponding publication) and corresponding generic key-dependency schemes to improve attack resistance (see [45] for the corresponding publication), on the other hand focussing on the employment of JPEG2000 as a robust visual hash function (see [50] and [51] for results on robustness and sensitivity of a corresponding scheme, respectively).

4.2.1 Hashing Security: Attacks and Key-Dependency Schemes

Introduction

The widespread availability of multimedia data in digital form has opened a wide range of possibilities to manipulate visual media. In particular, digital image processing and image manipulation tools offer facilities to intentionally alter image content without leaving perceptual traces. Therefore, it is necessary to provide ways of ensuring integrity other than human vision.

Similar problems have occurred for all kinds of digitally stored data and cryptography offers various solutions to hinder undetected manipulation. These are mostly combinations of hash functions and encryption algorithms. Whereas such techniques may be applied in principle to visual data as well, these data types have specific features that make specific procedures desirable.

Classical cryptographic tools to check for data integrity like the cryptographic hash functions MD-5 or SHA are designed to be strongly dependent on every single bit of the input data. While this is desirable for a big class of digital data (e.g. executables, compressed data, text), manipulations to visual data that do not affect the visual content are very common and often necessary. This includes lossy compression, image enhancement like filtering, and many more. All these operations do of course change the bits of the data while leaving the image perception unaltered.

To account for this property of visual data new techniques are required which do not assure the integrity of the digital representation of visual data but its visual appearance. In the area of multimedia security two types of approaches have been proposed to satisfy those requirements in recent years: semi-fragile watermarking and robust multimedia hashes [21, 22, 46, 32, 60, 69, 72, 73]. Main advantages of semi-fragile watermarking schemes are that watermarks are inserted into the image and become integral part of it thereby degrading it to a certain extent (depending on the type of scheme this may be reversible or not) and that
image manipulations may be localized in most schemes. On the other hand, robust visual hashes are not integral part of the image data (which may be a disadvantage) but do not degrade the image at all (which is an important advantage). In most visual hashing schemes, image manipulations may not be localized (which is a disadvantage). Watermarking and robust visual hashing may be combined as well: hash values may be embedded into visual data using watermarking technologies, but in this case robust watermarking as employed for copyright protection is required.

Among other proposals, several wavelet based robust hash algorithms have been introduced in recent years. While there are some stirmark [57] results available for most of them, fewer have been tested extensively, including both robustness and security tests. In this paper we evaluate and improve one of these algorithms proposed in 2000 [72] by Venkatesan, Koon, Jakubowski, and Moulin. All tests are performed with our own implementation of the algorithm, following the description in the paper, so that results might differ from results of the original implementation. Based on the results of the experiments and the proposed improvements, we derive certain general properties desirable for a robust hashing algorithm.

The methodology section reviews the algorithm subject to evaluation and describes the tests used to determine robustness to JPEG compression, noise addition as well as brightness and contrast adjustments, which are considered tolerable image operations. Another experiment is used to test the sensitivity to the insertion of objects into the image.

The results section is a discussion of the experimental results and possible shortcomings in the hashing scheme.

From the results of the experiments conducted, we examine potential weaknesses of the original algorithm and suggest improvements to overcome each of the discovered shortcomings in the improvements section. The experiments are then repeated for every modified version of the algorithm to evaluate the modifications’ success.

Finally, the most successful improvements are combined to create a more effective version of the algorithm, which is compared to the original. This version shows significant improvements in robustness to acceptable modification and an increased sensitivity to malicious modifications.

In the attack section we describe an attack that exploits the use of publicly known features in the scheme and allows an attacker to modify a forgery, such that it results in a hash value similar to the original image. Attack results show, that the attack can successfully reduce the hamming distance of hashes for the original and forgery to a value below the difference caused by high quality JPEG compression.

Regarding the attack resistance, we argue that it cannot be improved without either using different image features, which would result in an almost completely different algorithm, or by adding key dependence to early stages of the algorithm (e.g. during the wavelet decomposition step, see e.g. [15, 16]).

Methodology

The algorithm proposed by Venkatesan, Koon, Jakubowski, and Moulin [72] intends to provide a robust and secure hashing algorithm for security purposes. It combines wavelet decomposi-
tion with various error correcting codes to achieve stability. Like other secure schemes it uses a secret key to initialize a pseudo-random number generator, which is used at multiple steps of the algorithm.

The algorithm consists of 4 basic steps.

- The image is transformed, using a 3-level pyramidal wavelet transformation
- For each of the resulting subbands a feature vector $F_i$ is calculated. This is done by randomly partitioning the subband and calculating a statistical measure for each region. For the approximation the statistical measure used is the arithmetic average, for all other subbands the variance computed.
- The real number elements of each $F_i$ are projected to $\{0 \ldots 7\}$ using randomized rounding. The resulting values are concatenated to form a preliminary hash string $H_p$.
- The hash string $H_p$ is shortened by feeding it into the decode stage of a Reed-Muller error correcting code. This does not only shorten the hash string, but also improves robustness.
- In the final step a Linear Code algorithm is applied to the hash, again both shortening it and increasing robustness.

Note that every robust Hash algorithm should satisfy the following property: it should be robust to common (non-hostile) image processing operations but sensitive to malicious modifications of the image.

**Robustness** Early tests with stirmark showed, that all wavelet based schemes are very sensitive to geometric transformations, while being stable to pure amplitude based modifications. This is both a result of the design, as well as the inherent nature of wavelet transformation.

We further examined non-geometric transformations in greater depth. Each of them represents a class of image processing algorithms commonly used. All test cases were created using the `convert` command line tool, which is part of ImageMagick.

**JPEG** This DCT-based, lossy compression scheme is the most common file format for full color pictures, such as photographs. Besides the actual JPEG compression, this test also indicates resistance to other DCT based modifications.

**Noise** In this test an increasing amount of uniform noise was added to the image. Though noise is generally associated with analog transmission technologies, many image processing algorithms leave artifacts comparable to random noise. This holds for many watermarking schemes, dithering, color reduction and some lossy compression schemes.

**Contrast/Gamma** Changes to image contrast and gamma correction are very common, when trying to increase visual image quality or adjusting the appearance of multiple images to fit together. Other image improvements like histogram equalization leave similar traces.
**Object Insertion** This test was added to evaluate the ability of authentication hashes to detect malicious changes. Within the test, an increasing number of small icons of different size (8, 16, 32 or 64 pixels) are randomly inserted into the image.

**Experimental Results**

<table>
<thead>
<tr>
<th></th>
<th>baboon</th>
<th>barb</th>
<th>boat</th>
<th>jet</th>
<th>lena</th>
<th>peppers</th>
<th>truck</th>
<th>zelda</th>
</tr>
</thead>
<tbody>
<tr>
<td>baboon</td>
<td>0</td>
<td>0.367</td>
<td>0.375</td>
<td>0.392</td>
<td>0.375</td>
<td>0.417</td>
<td>0.375</td>
<td>0.383</td>
</tr>
<tr>
<td>barb</td>
<td>0.367</td>
<td>0</td>
<td>0.358</td>
<td>0.475</td>
<td>0.375</td>
<td>0.483</td>
<td>0.442</td>
<td>0.417</td>
</tr>
<tr>
<td>boat</td>
<td>0.375</td>
<td>0.358</td>
<td>0</td>
<td>0.45</td>
<td>0.45</td>
<td>0.425</td>
<td>0.367</td>
<td>0.458</td>
</tr>
<tr>
<td>jet</td>
<td>0.392</td>
<td>0.475</td>
<td>0.45</td>
<td>0</td>
<td>0.367</td>
<td>0.392</td>
<td>0.483</td>
<td>0.358</td>
</tr>
<tr>
<td>lena</td>
<td>0.375</td>
<td>0.375</td>
<td>0.45</td>
<td>0.367</td>
<td>0</td>
<td>0.375</td>
<td>0.45</td>
<td>0.375</td>
</tr>
<tr>
<td>peppers</td>
<td>0.417</td>
<td>0.483</td>
<td>0.425</td>
<td>0.392</td>
<td>0.375</td>
<td>0.425</td>
<td>0</td>
<td>0.358</td>
</tr>
<tr>
<td>truck</td>
<td>0.375</td>
<td>0.442</td>
<td>0.367</td>
<td>0.483</td>
<td>0.45</td>
<td>0.425</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>zelda</td>
<td>0.383</td>
<td>0.417</td>
<td>0.458</td>
<td>0.358</td>
<td>0.375</td>
<td>0.45</td>
<td>0.358</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.1: Cross Comparison Results for VKJM

To obtain an initial estimate and upper bound of the hamming distance threshold for considering an image untampered, a set of different images is compared (see Table 4.1). The hamming distance between two independent images is consistently below the optimal distance of $\frac{1}{2}$. This is mainly a result of the fixed values used in the randomized rounding procedure, which favor the lower and upper bounds, and a non uniform distribution of feature values.

For the tests we assumed a threshold of $\frac{1}{2}$ of the average distance between unrelated images (i.e. two images above that threshold are assumed to be tampered with (or even different)). This threshold is shown in all the graphs and varies for different versions of the algorithm.

Figure 4.11 shows the hamming distance of images after various manipulations to the respective original. The normalized hamming distance is shown on the ordinate, while the operation parameter is on the abscissa.

All the curves show a very unsteady behavior not seen in other algorithms. This does not only complicate the assessment of results and aggravate the determination of a meaningful threshold, it also denotes the presence of some amount of randomness in the hash construction.

The scheme does not distinguish between the different subbands except for using another statistical measure for the LL subband. This results in a high dependency of the resulting hash on high frequency portions of the image. This behavior is very uncommon, as it defies one of the very basic properties of the HVS.

**JPEG** This curve is the most erratic among all tests. The average distance reaches the threshold at a quality level of about 60, which is still fairly high quality. In the worst case, even a very high quality setting of 90 prevents authentication of the image.

The bad performance can be explained by VKJM’s sensitivity to changes in the high frequency subbands. These subbands are strongly affected by lossy compression. Over a wide range of quality settings (20-90) the results for these subbands are almost completely random, causing the curve’s bumpiness. For higher quality levels only a subset of the coefficients seem to be affected. At lower quality levels, lower frequencies are modified as well, causing even worse results.
Figure 4.11: VKJM Testing Results
Noise Though the noise resistance behavior is much smoother, it is even worse in absolute terms, than JPEG results. The reasons are similar to those of the JPEG test. The sensitivity to noise can be explained by the use of variance as feature for high frequency subband. The variance in these subbands directly increases with the addition of noise, such that all values of the feature vector are modified.

Contrast/Gamma Both of these suffer from the fact, that absolute values are used during feature vector creation. Thus, even very small changes cause completely different hash values.

Note, that the exact process of quantization is not described in the paper, so that this problem might be limited to this implementation.

Object Insertion In the average case 6 objects of size 16x16px can be inserted into the image, without reaching the threshold. In the worst case even the insertion of 20 objects is not sufficient to leverage the hamming distance above the threshold. The distance between the \( \min \) and \( \max \) curve is vary large, which makes automated integrity evaluation even harder.

Robustness Improvements

Several problems of the scheme, which were outlined in the previous section, can be lessened by fairly small changes.

Gray Coding Both the error correcting coders and the hash comparison operate on the bit level, while the feature vectors are 3 bit numeric values. To make values more suitable for bit operation, they are encoded using gray codes. This encoding guarantees, that very small changes in the numerical value will only lead to one modified bit.

This modification only works for small changes, which are rather uncommon. Therefore results are only very slightly better, than the original ones.

Weighting In bits of the hash value created by VKJM, the contributions of the individual subbands can be identified. A closer analysis shows, that most of the incorrect bits are contributed by the high frequency subbands. This seems reasonable as these subbands are the first to be changed by the tested manipulations.

To increase stability the impact of the more volatile subbands has to be diminished. Two ways of achieving this have been tested. First the segmentation depth of higher subbands was decreased, such that the feature vector contains less elements, which cover larger areas. Second, the number of bits used during quantization was changed from 3 to a subband dependent number (bits = dwt level).

Of these two, the variable bit allocation proved to be more efficient. The reduction of the segmentation depth lead to stronger fluctuations of individual values within the feature vector.

Weighting is especially successful in JPEG compression for images with small high frequency components, such as lena.
Normalization The bad performance in the *gamma* and *brightness* tests strongly suggests the use of an image based scale for quantization rather than fixed values used previously. For the variances the second moment of the subband is used as maximum. In the low pass subband, the maximum is set to be twice the overall average of the low frequency coefficients.

Results in the contrast test improve significantly after this modification, especially when images are made darker. For lighter images, pixel values are likely to be clipped off, leading to modifications in variance, which cannot be compensated. For very dark images, the amount of information left is simply not sufficient to calculate a reasonable hash value.

The gamma curve does not exhibit similar improvement. Other than contrast modification, gamma correction is not a linear transformation of pixel values, so that it cannot be compensated by scaling.

Correlation Besides robustness to some common modifications, authentication hashes are expected to react strongly to massive local changes. This behavior is not visible in the original test results. One problem is, that changes within a small region have a limited set of hash bits, they can potentially affect.

To increase the number of affect-able bits while retaining stability the elements of the extracted feature vectors are correlated by xor’ing them after quantization. In theory, global changes will affect approximately the same bits within all coefficients, such that flipped bits cancel each other out during the correlation step, while changes limited to a single coefficient are propagated to multiple elements.

As for other improvements, the efficiency of this one varies from image to image.

<table>
<thead>
<tr>
<th></th>
<th>baboon</th>
<th>barb</th>
<th>boat</th>
<th>jet</th>
<th>lena</th>
<th>peppers</th>
<th>truck</th>
<th>zelda</th>
</tr>
</thead>
<tbody>
<tr>
<td>baboon</td>
<td>0</td>
<td>0.512</td>
<td>0.438</td>
<td>0.487</td>
<td>0.388</td>
<td>0.512</td>
<td>0.388</td>
<td>0.537</td>
</tr>
<tr>
<td>barb</td>
<td>0.512</td>
<td>0</td>
<td>0.45</td>
<td>0.475</td>
<td>0.375</td>
<td>0.5</td>
<td>0.375</td>
<td>0.475</td>
</tr>
<tr>
<td>boat</td>
<td>0.438</td>
<td>0.45</td>
<td>0</td>
<td>0.45</td>
<td>0.55</td>
<td>0.575</td>
<td>0.475</td>
<td>0.5</td>
</tr>
<tr>
<td>jet</td>
<td>0.487</td>
<td>0.475</td>
<td>0.45</td>
<td>0</td>
<td>0.575</td>
<td>0.425</td>
<td>0.425</td>
<td>0.45</td>
</tr>
<tr>
<td>lena</td>
<td>0.388</td>
<td>0.375</td>
<td>0.55</td>
<td>0.575</td>
<td>0</td>
<td>0.5</td>
<td>0.375</td>
<td>0.475</td>
</tr>
<tr>
<td>peppers</td>
<td>0.512</td>
<td>0.5</td>
<td>0.575</td>
<td>0.425</td>
<td>0.5</td>
<td>0</td>
<td>0.525</td>
<td>0.45</td>
</tr>
<tr>
<td>truck</td>
<td>0.388</td>
<td>0.375</td>
<td>0.475</td>
<td>0.425</td>
<td>0.375</td>
<td>0.525</td>
<td>0</td>
<td>0.45</td>
</tr>
<tr>
<td>zelda</td>
<td>0.537</td>
<td>0.475</td>
<td>0.5</td>
<td>0.45</td>
<td>0.475</td>
<td>0.45</td>
<td>0.45</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.2: Cross Comparison Results for Modified (vkjm-all) VKJM

After testing the improvements separately, two combinations of improvements were combined - *vkjm-stable* contains all described stability improvements, but does not use correlation, *vkjm-all* contains all proposed improvements. The variable bit quantization is used for weighting in both cases. The *stable* algorithm is superior to the original in all test cases (compare Figure 4.12). Note that the average distance between different images in the cross comparing results is slightly higher (see Table 4.2), leading to an increased threshold. Surprisingly, the sensitivity tests (i.e. object insertion) also benefit from the stability improvements.

For *vkjm-all* some of the stability gains are lost (especially with respect to robustness against JPEG, see Fig. 4.13.a) to an increased sensitivity to forbidden modifications (see Figure 4.13). Especially the worst case scenarios are significantly improved.
Figure 4.12: Results for stable VKJM
Figure 4.13: Results for modified VKJM
Given a big number of testing results, as well as the attempt to straighten out the most important problems of the algorithm a few inherent shortcomings stand out, which can not easily be overcome.

**Erratic results** Independent of modification, images and test case, the scheme produces very unsteady result curves. This behavior makes very high thresholds for authenticity decisions necessary, which increases the probability of false positives.

**Unpredictability** The behavior of the scheme varies dramatically from image to image, making the evaluation of modifications extremely hard and pushing the requirement for high thresholds further.

**Sensitivity** Even choosing the best performing modifications, the results are still very sensitive to certain allowed operations, especially noise and gamma.

Only sensitivity to object insertion consistently reaches acceptable levels, when using feature vector correlation.

All of these problems are directly dependent of the choice of image features. The variances within high frequency subbands are naturally unstable. The bit oriented stabilization step is unable to compensate for these fluctuations. The disadvantages of this choice become clearer, when comparing it to the intra-image relationships used by SDS [38].

**Attack**

From a security point of view, the major problem of VKJM is the use of variance and average as basis of the hash value [21]. Both of these are publicly available and very easy to modify. The whole security lies in the random partitioning, which makes it impossible to know, which parts are actually used to calculate the statistics.

However, both average and variance mostly change gradually within an image, so that if the measures of two images match within a certain partition, they will at least be similar within any other partition covering approximately the same area as well.

This can be easily exploited for an attack as follows.

- For a given image $I$ create a manipulated image $F$.
- Wavelet transform both $I$ and $F$
- Use the same partition to separate $I$ and $F$ into regions $I_1 \ldots I_n$ respectively $F_1 \ldots F_n$
- For each $i = 1 \ldots n$ region adjust the variance or average, depending on the subband, of $F_i$ to match the value for $I_i$. Noise($x$) produces random noise with mean 0 and variance $x$.

$$
\forall x, y \in I_i, I_i \in LL : F'_i(x, y) = F_i(x, y) \cdot \left( \frac{\text{Avg(lena)}}{\text{Avg(lena)}} \right)^{\alpha_{\text{multiplicative}}} + (\text{Avg(lena)} - \text{Avg(lena)}) \cdot \alpha_{\text{additive}}
$$
∀x, y ∈ I_i, I_i \notin LL : F_i'(x, y) = F_i(x, y) \cdot \sqrt{\frac{\text{Var}(\text{lena})}{\text{Var}(\text{lena})}}^{\text{multiplicative}} + \text{Noise}((\text{Var}(\text{lena}) - \text{Var}(\text{lena})) \cdot \alpha_{\text{additive}})

F_i$ does not need to match $I_i$ exactly. The attack strength $\alpha \in [0..1]$ is used to control by how much the forgery $F$ is modified. It is used to control both additive and multiplicative adjustments. The balance between these two is an algorithm parameter, that has been determined experimentally. As the additive adjustment of variance can only be used to increase variance, multiplicative adjustment is used as dominant component in our experiments.

We use $\alpha_{\text{multiplicative}} = \alpha, \alpha_{\text{additive}} = \frac{\alpha}{10}$ in all results displayed. Different parameters for variance and average adjustment could be used to achieve improvements in image quality.

A lower strength will improve the visual quality of the resulting forgery, but also increase the difference in the final hash value.

- Use inverse wavelet transformation on the modified $F$ to receive $F'$

$F'$ will now have a hash very similar or equal to that of the original image $I$. The actual difference and the visual quality of the resulting image depends on the original difference and the parameters used during the adjustment step. Even for $\alpha = 1$ the $F'$ is not guaranteed to have the same hash value as $I$, because the hash calculation calculates variances and averages for other parts of the image than the attack.

In the experiments a sliding window was used instead of a fixed partition to avoid blocking artifacts. The basic mechanism is not affected by this change.

Figure 4.14 shows the hamming distance between original image and attacked forgery depending on the attack strength parameter $\alpha$ (note that $\alpha$ is multiplied by 100 in the plot). The case $\alpha = 0$ is equivalent to the unmodified forgery, thus, for an effective attack, the distance must drop below this value for some $\alpha > 0$. This is achieved for both test cases. For truck the optimal result is reached at $\alpha = 0.9$, while a much weaker attack is required for lena. This explains, why the visual impact of the attack is less significant for lena [Figure 4.15(e)], than for truck [Figure 4.15(f)]. [Figure 4.15] provides visual examples for forged images onto which the attack has been mounted. The amount of difference the algorithm can be made to tolerate is astonishing.

It should be noted, that the authors propose to calculate hashes in different transformation domains (DCT, etc.) to improve security. This would make the attack somewhat more complicated, but it does not solve the underlying security problem.

The proposed attack takes place in a very early stage of the algorithm, thus additional security can’t be added during the coding stages, but before or during the statistics calculation. This problem is not limited to this particular algorithm. Many algorithms suffer from
the fact, that the collected image features are publicly known and only latter stages introduce key dependency. This allows an attacker to directly modify the underlying image features, without the need of knowing any secret key.

Changing the feature extraction itself would merely result in a completely new scheme. We propose a general approach, that can applied to all wavelet based algorithms. In order to improve security and key dependency for the resulting hash value, we propose a general approach, that can applied to all wavelet based algorithms. The wavelet decomposition itself could be made key dependent (this approach has been used for securing wavelet-based watermarking schemes [14, 15, 43] and efficient encryption schemes [58, 59].

Key-Dependency Schemes

Pseudo Random Partitioning A common approach to generate secret image features is to first create a pseudo-random partitioning of the image and compute features independently for every partition. The exact values of the features can not be computed without knowledge of the key used to seed the PRNG, because the regions on which the features are computed are not known.

Random partitioning is used as original key-dependency scheme in the hash algorithm of Venkatesan et al. [72]. Its use is orthogonal to the following two schemes and can be easily combined with either of them to further increase security (which will be done in our experiments).

Random Wavelet Packet Decomposition In the classical wavelet transformation only the low-low-sub-band can be further decomposed, resulting in the typical pyramidal structure. Wavelet packet decomposition [17] removes this constraint and allows to further decompose any sub-band. The decision which sub-bands are decomposed is either determined by a given structure or based on some measure of optimality.

By using a pseudo random number generator to decide, if a sub-band should be further decomposed, we can make the decomposition structure key dependent. This approach has been shown to be effective in selective image encryption [59] and in securing watermarking schemes [17].

Parameterized Filters Wavelet decomposition typically uses fixed, well known filters, such as the Daubechies filters. There are also methods to generate families of wavelet filters from a number of parameters, that can be freely chosen (we employ a family of parameterized orthogonal Daubechies wavelet filters [67]). If these parameters are kept secret, they can be used as a key for the decomposition. Similar to the wavelet packet case, this type of key-dependency has been used before in selective image encryption [33] and watermarking [15].

Experiments and Results

We have tested both proposed schemes by including them into a authentication hash algorithm introduced by Venkatesan et al. [72]. The original algorithm achieves key dependency through
Figure 4.15: Visual examples for the effectiveness of the attack

(a) Original lena

(b) Original truck

(c) Forged lena

(d) Forged truck

(e) Attacked lena $\alpha = 55\%$, hamming distance < 0.02

(f) Attacked truck $\alpha = 90\%$, hamming distance < 0.02
random partitioning. We use this algorithm as described in the previous section.

**Key Dependency** A key dependency scheme can only improve security if the choice of the key has a significant impact on the resulting hash value. All following figures show the normalized Hamming distance of a hash created with some fixed key value to other hashes, produced with varying other key values. Key values are displayed along the ordinate, resulting Hamming distances along the abscissa.

The random partitioning approach, though vulnerable by a simple attack (see [44] and the previous subsection), is very effective in adding key dependency, with average Hamming distance 0.336 and very few keys reaching values below 0.2 (see Fig. 4.16(a)). The figure shows the results 10000 different partitions, compared to a fixed key at position 5000. A similar phenomenon (i.e. security weaknesses in spite of a key-dependent hash) was pointed out by Radhakrishnan et al. [60] for the block-based VHF. This contradictory behaviour was improved by adding block inter-dependencies to VHF.

Random wavelet packet decompositions with a constant decomposition probability for all subbands makes shallow trees far more likely than deep trees. This increases the chance of collisions, especially for shallow trees. Following a previous suggestion [59], we use a higher initial decomposition probability for the first decomposition level and decrease it subsequently for every subsequent decomposition recursion (we use a base value of 0.9 ($p = 0.55$) and a change factor of $-0.1$ [59]). The obtained average Hamming distance (Fig. 4.16(b)) is 0.3570 and about 0.73% of all distances are below 0.1. However, we result in 20 “almost” correct keys (distance < 0.05) which makes the approach less reliable.

Even with random decomposition in place, the key of the standard algorithm required to create partitions for extracting localized feature vectors may be varied as well, thus increasing the key space and possibly overall security. Fig. 4.16(c) shows key dependency results for varying both keys. The average distance for this setup increases to 0.3884 with no incorrect keys reaching distances below 0.1. Combining both strategies obviously significantly increases the keyspace while maintaining the high sensitivity to key variations of the original standalone random partitioning scheme.

Experiments concerning filter parametrization are based on a parameterized filter with 4.
parameters $(1.0, 1.5, -2.0, -1.0)$, all parameters were modified in a range of $\pm 1.0$ in steps of 0.2, resulting in $11^4 = 14641$ combinations. The correct key for this test is 7320. The results for parameterized filters are almost as good as the random partition scheme, with an average of 0.265 and only 0.53% of the keys below 0.1 (see Fig. 4.17(a)).

![Figure 4.17: Key dependency test: Hamming distances between hashes generated with different keys (based on a parameterized filter).](image)

Similar to the random decomposition, using parameterized filters adds key dependency to the decomposition stage. Thus, the parameterization key can also be combined with the standard partitioning key used during a later stage of the scheme. When both keys are used, the average hamming distance increases slightly to 0.2795, additionally there are no more incorrect keys reaching values below 0.1 (see Fig. 4.17(b)). Again, combining the two schemes maintains sensitivity towards key alterations while increasing the keyspace.

**Key Space** A major concern of any key dependent algorithm is the number of distinct keys that can be used. If the number of keys is too small, the scheme is vulnerable to brute force attacks. The discrete key space of both random partitioning and random decomposition grows exponentially with a free algorithm parameter (e.g., following the formula given in [59], a decomposition depth of 5 leads to $\approx 2^{1043}$ different keys in random decomposition). Thus the size of the key space can be easily adjusted and it seems that a suitable number of keys is available for any level of security desired. However, a bigger number of keys may have some undesired side effect on the overall algorithm.

In random partitioning, the areas get smaller with an increasing number of keys. This makes the hash more sensitive to minor image modifications and many keys will produce fairly similar results. Random decompositions suffers from the fact, that high decomposition depth leads to a big number of very similar tree structures, which lead to identical hash values. Therefore, the key space needs to be set to some sensible compromise in this two cases (e.g. decomposition depth 5 is a good choice for random decomposition).

Contrasting to the previous cases, the key values are continuous rather than discrete for filter parametrization. Therefore, a quantization must be defined to determine the number of possible keys. This can be done by defining a range of valid parameters $(d_{\text{min}} \ldots d_{\text{max}})$
and quantization function \( Q(d) = \left\lfloor \frac{d}{q} \right\rfloor \). Now the number of keys \( f(n) \) for a filter with \( n \) parameters can be calculated: 

\[
\frac{d_{\text{max}} - d_{\text{min}}}{q} \approx 2^n
\]

The filter parametrization used is based on trigonometric functions (\( \sin, \cos \)). Thus, the parameters have a range of \( (-\pi \ldots \pi) \).

In the following, we determine the quantization function by a simple experiment. Fig. 4.18(a) shows the results, if only one parameter of a 6 dimensional parameterization is modified in the range of \( \pm 1.0 \) with a step size of 0.01. There is a curve for every one of the six single parameters. The graph’s values change in multiple steps, suggesting that key values within about 0.05 produce the same hash. Thus, when generating parameters from the key the granularity should be 0.05–0.10 (the parameters used to create the graph were \((1.0, 1.5, -2.0, -1.0, 0.0, 0.5)\)). To be on the safe side, we limit the distance in a single parameter between two keys to be no smaller than 0.1. Using these values, the number of available keys can be calculated as: 

\[
\left\lfloor \frac{\pi - (-\pi)}{0.1} \right\rfloor = 200 \cdot \pi \approx 628^n
\]

The number of resulting keys dependent on \( n \) is shown in Table 4.3.

![Figure 4.18: Hamming distances.](image)

The granularity \( q \) is very important for the security of the scheme and might be dependent also on the number of parameters \( n \). It seems intuitive, that the influence of a single parameter on the overall result will decrease for a higher number of parameters. This, however, is not the case as shown in Fig. 4.18(b). For every filter dimension shown on the x-axis, the average Hamming distance between the hash for a fixed parameter vector and all hashes resulting from the first parameter of this vector being changed in the range of \( \pm 1.0 \) is shown on the y-axis. This average distance indicates how much influence a single parameter has on the resulting hash value – it varies significantly from 0.12 to almost 0.2 without any clear trend up or downwards for an increasing number of dimensions. Thus, \( d \) does not have to be selected dependent on \( n \).
**Attack Resistance**  The reason for the idea of enhancing the original partitioning scheme with a key dependent wavelet transformation is its vulnerability to the simple attack shown in the previous section (and [44]). The major problem of the use of variance and average as basis of the hash value is that both are publicly available and very easy to modify [22]. Both average and variance mostly change gradually within an image, so that if the measures of two images match within a certain partition, they will at least be similar within any other partition covering approximately the same area as well. This is exploited by the referenced attack.

The goal of the proposed new schemes is to eliminate feature correlations between transformations computed with different key values. Though some parameters apparently result in the exact same hash value, overall hash values strongly depend on the selected parameters as we have seen in the previous subsections. Attempting an attack gets very hard without knowledge of the transform domain used for creating the original hash. The underlying assumption of the attack is, that it is operating on a transformed image identical to the one used to calculate the hash value. Only if this is the case, adjusting the forgery’s features to match those of the original has the desired effect on the hash value. By using a transform domain with an incorrect set of parameters, this assumption is weakened. The adjusted forgery’s features will only match those of the original for the filter chosen for the attack. This does not necessarily make them more alike for any other filter. Fig. 4.19 shows the results of the attack using both techniques and various decomposition keys.

The Hamming distance for the correct key in the random decomposition case after the attack has been mounted is 0.0166. The average distance after the attack for all random decompositions considered is increased to 0.0728, however, the large number of “correct” keys (i.e. leading to the same result as the key used to compute the original hash) makes the scheme unreliable (Fig. 4.19(a)). This corresponds well to the results with respect to key dependency displayed in Fig. 4.16(b).

Given the key dependency tests (Fig. 4.17(a)), filter parameterization seems more promising than random decomposition. Though only a small number of filters renders the attack completely useless, its effects are attenuated considerably, thus improving the scheme's
overall security. The average distance of 0.0666 after the attack, compared to 0.0083 for the correct key, is a definite improvement (see Fig. 4.19(b)). The number of successful attacks (i.e. equally successful as without filter parametrization) is negligible. However, considering the high number of key values with still rather low Hamming distances, the effects of the attack can only said to be weakened to some extent.

Conclusion

We have discussed the use of key dependent wavelet transforms as a means to enhance the security of wavelet based hashing schemes. Whereas key dependency and keyspace of the hashing scheme considered in experiments have been significantly improved, the attack resistance has been improved by using parametrized wavelet filters to a small extent only.

4.2.2 GAUSS/Robust Visual Hashing using JPEG2000

Introduction

In order to ensure the integrity and authenticity of digital visual data, algorithms have to be designed which consider the special properties of such data types. On the one hand, such an algorithm should be robust against compression and format conversion, since such operations are a very integral part of handling digital data. On the other hand, such an algorithm should be able to recognize a large amount of different intentional manipulations to such data.

A robust visual hashing scheme usually relies on a technique for feature extraction as the initial processing stage, often transformations like DCT or wavelet transform are used for this purpose. Subsequently, the features (a set of carefully selected transform coefficients) are further processed to increase robustness and/or reduce dimensionality. Two different approaches have been followed with respect to the final stage of the algorithms which has to produce the final hash value:

- The features are either directly converted into binary representation or fed into the decoder stage of error correcting codes or linear codes. This approach has the advantage that different hash values can be compared by evaluating the Hamming distance which serves as a measure of similarity in this case. Whereas it is desirable from the applications point of view to estimate the amount of difference between images by using those hash functions, this property severely threatens security and facilitates “gradient attacks” by iteratively adjusting hostile attacks to minimize a change in the hash value.

- A classical cryptographic hash function is applied to the extracted robust feature values. This approach guarantees security but the result is simply binary: image modification detected or not.

Authentication of the JPEG 2000 bitstream has been described in previous work. In [24] it is proposed to apply SHA-1 onto all packet data and to append the resulting hash value after the final termination marker to the JPEG 2000 bitstream. Contrasting to this approach, we focus onto robust authentication. This means that only certain parts of the bitstream are
subject to authentication. The technical solution of how authentication can be applied to the entire codestream while it remains valid also for parts of it has been derived using Merkle hash trees [56] (and tested with MD-5 and RSA).

**Authenticator JPEG 2000**

**JPEG 2000 Basics** The JPEG 2000 [71] image coding standard uses the wavelet transform as energy compaction method. The major difference between previously proposed wavelet-based image compression algorithms such as EZW or SPIHT [71] is that JPEG 2000 operates on independent, non-overlapping blocks whose bit-planes are coded in several passes to create an embedded, scalable bitstream. JPEG 2000 may be operated in lossy and lossless mode (thereby using a reversible integer transform) and outperforms JPEG with respect to rate/distortion performance especially at lower bitrates.

The final JPEG2000 bitstream is organized as follows: The main header is followed by packets of data which are all preceded by a packet header. In each packet appear the code-words of the code-blocks that belong to the same image resolution (wavelet decomposition level) and layer (which roughly stand for successive quality levels). Depending on the arrangement of the packets, different progression orders may be specified. Among others, resolution and layer progression order are most important for grayscale images.

**Robust JPEG 2000 Authentication** In the context of robust authentication it turns out to be difficult to insert the hash value directly into the codestream itself (e.g. after termination markers), since in any operation with involves decoding and recompression the original hash value would be lost (which should not automatically imply that the image content was changed significantly!). The only applications which do not destroy the hash value are purely bitstream oriented, like e.g. rate adaptation transcoding by simply dropping parts of the packet data. As a consequence, a possible solution to this dilemma would be to use a robust watermarking scheme to embed the hash value into the codestream, provided that the embedding does not change the features involved in computing the hash value. A different solution would be to signal the hash value in the context of a MPEG-7 or MPEG-21 description, separated but attached to the codestream. These questions are not further covered in this work, they are subject to further investigation.

In the following we restrict the attention to the assessment of different parts of the codestream with respect to their usefulness as robust feature values. Due to the embeddedness property of the bitstream, the perceptually more relevant bitstream parts are positioned at the very beginning of the file. Consequently, the bitstream is scanned from the very beginning to the end, and the data of each data packet - as they appear in the bitstream, excluding any header structures - are collected sequentially to be then used as visual feature values.

**Experiments: Robustness Results**

In this section we investigate if our proposed method is robust to JPEG 2000 recompression and JPEG compression on the one side. On the other side, sensitivity to hostile local image alterations is discussed in the next subsection.
In our experiments we use classical 8bpp image data, including the well known Lena image at varying image dimensions (512 × 512, 1024 × 1024, and 2048 × 2048 pixels), the plane image (see 4.21.a), and frame no. 17 from the surfside video sequence.

The experiments are conducted as follows: first, the feature values (i.e. packet data) are extracted from the JPEG 2000 codestream. Subsequently, the codestream is decoded and the image alteration is performed. Finally, the image is again JPEG 2000 encoded using the coding settings of the original codestream and the feature values are extracted and compared to the original ones.

The results which are presented in this section show the number of feature values (in bytes) required to detect an image modification (recall that packet data is used according to its appearance in the codestream). A value of - for instance - 42 means that the first 41 bytes of feature values (i.e. first 41 bytes of the codestream) are equal when comparing the modified image to the corresponding original codestream. The value itself can be easily interpreted: The higher the value, the more robust is the proposed method against the tested attack. In general, we want to see high values against JPEG 2000 and JPEG compression (robustness), but low values against local manipulations.

As a first step we test the JPEG 2000 recompression robustness by varying the coding options of the initial generation of the code stream (different parameters are used for feature extraction). These options include the JPEG 2000 standard parameter setting as well as coding in lossless mode, in resolution progression order, together with a varying wavelet-transform decomposition level. The JPEG 2000 compression (interpreted as image modification) is used in default mode, i.e. layer progressive order, 5 decomposition levels (wlev5), and lossy coding.

![Figure 4.20: Different coding parameters used for feature extraction, lena512](image)

Fig. 4.20 shows that if the parameters of JPEG 2000 compression and feature extraction match each other, the robustness against compression is very high. If they do not match, the robustness can be low and this is especially true if lossless coding is used for feature extraction (Fig. 4.20.b). The reason is that no quantization is used in lossless mode, therefore no robustness can be expected against compression. Since JPEG 2000 compression will often be used with default settings, we subsequently apply the feature extraction in lossy mode with layer progression order since we get maximal robustness in this case.
Tables 4.4 and 4.5 show the robustness of our proposed feature extraction mechanism against JPEG 2000 compression for different images. If the same coding options are used for feature extraction and for compressing the image, our method proves to be extremely robust against JPEG2000 compression (see table 4.5). This also means that JPEG 2000 encoding-decoding-encoding does not change the bitstream very much which was one of the design goals and is important for image editing applications.

<table>
<thead>
<tr>
<th>bpp</th>
<th>4.5</th>
<th>2.66</th>
<th>0.8</th>
<th>0.4</th>
<th>0.2</th>
<th>0.133</th>
<th>0.05</th>
</tr>
</thead>
<tbody>
<tr>
<td>lena512</td>
<td>850</td>
<td>309</td>
<td>77</td>
<td>28</td>
<td>28</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>lena1024</td>
<td>220</td>
<td>9</td>
<td>9</td>
<td>24</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lena2048</td>
<td>224</td>
<td>83</td>
<td>109</td>
<td>45</td>
<td>45</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>plane512</td>
<td>293</td>
<td>64</td>
<td>41</td>
<td>33</td>
<td>33</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>surf2048x1024</td>
<td>239</td>
<td>248</td>
<td>262</td>
<td>41</td>
<td>41</td>
<td>41</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.4: Sensitivity against JPEG 2000 compression: wlev6 used for feature extraction, wlev5 used for JPEG2000 compression

Nevertheless, if different options are used, we can see a good robustness against moderate compression up to 1 bpp as well for all tested images (see Table 4.4).

<table>
<thead>
<tr>
<th>bpp</th>
<th>4.5</th>
<th>2.66</th>
<th>0.8</th>
<th>0.4</th>
<th>0.2</th>
<th>0.133</th>
<th>0.05</th>
</tr>
</thead>
<tbody>
<tr>
<td>lena512</td>
<td>4995</td>
<td>309</td>
<td>77</td>
<td>28</td>
<td>28</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>lena1024</td>
<td>4205</td>
<td>60</td>
<td>40</td>
<td>24</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lena2048</td>
<td>1517</td>
<td>8119</td>
<td>4755</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>plane512</td>
<td>1357</td>
<td>466</td>
<td>460</td>
<td>233</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>surf2048x1024</td>
<td>1795</td>
<td>4964</td>
<td>4755</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.5: Sensitivity against JPEG2000 compression: identical coding options used for compression and feature extraction (wlev5)

Sensitivity against JPEG compression (see Table 4.6) is comparable to the sensitivity against JPEG 2000 compression in case the parameters do not match (Table 4.4) for better quality, at lower bitrates JPEG robustness is lower (which matches the poorer JPEG compression performance at low bitrates).

<table>
<thead>
<tr>
<th>quality</th>
<th>90</th>
<th>80</th>
<th>70</th>
<th>60</th>
<th>50</th>
<th>40</th>
<th>30</th>
<th>20</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>lena512</td>
<td>42</td>
<td>24</td>
<td>24</td>
<td>4</td>
<td>28</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lena1024</td>
<td>24</td>
<td>4</td>
<td>4</td>
<td>24</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lena2048</td>
<td>232</td>
<td>109</td>
<td>125</td>
<td>31</td>
<td>45</td>
<td>22</td>
<td>23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>plane512</td>
<td>177</td>
<td>64</td>
<td>43</td>
<td>43</td>
<td>42</td>
<td>64</td>
<td>38</td>
<td></td>
<td></td>
</tr>
<tr>
<td>surf2048x1024</td>
<td>306</td>
<td>41</td>
<td>41</td>
<td>27</td>
<td>7</td>
<td>56</td>
<td>7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.6: Sensitivity against JPEG compression: wlev5 used for feature extraction

The decomposition level used for the feature extraction can be used to influence the sensitivity against image alterations. This effect is shown in Table 4.7. We observe that a higher number of decomposition levels generally shows a higher sensitivity against image modifications including JPEG compression (see the left columns in table 4.7), and a smaller number
decreases sensitivity against compression of this type - even against higher compression ratios (lower rows in the table, up to quality 50).

<table>
<thead>
<tr>
<th>wlev</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>quality 90</td>
<td>8</td>
<td>9</td>
<td>254</td>
<td>277</td>
<td>42</td>
<td>516</td>
<td>30</td>
</tr>
<tr>
<td>quality 70</td>
<td>8</td>
<td>9</td>
<td>26</td>
<td>113</td>
<td>42</td>
<td>23</td>
<td>45</td>
</tr>
<tr>
<td>quality 50</td>
<td>35</td>
<td>38</td>
<td>31</td>
<td>50</td>
<td>67</td>
<td>12</td>
<td>27</td>
</tr>
<tr>
<td>quality 30</td>
<td>25</td>
<td>11</td>
<td>26</td>
<td>72</td>
<td>1</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>quality 10</td>
<td>9</td>
<td>10</td>
<td>7</td>
<td>27</td>
<td>15</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 4.7: JPEG compression (lena512): different wlev used for feature extraction

A wavelet decomposition level of 6 or 5 applied for feature generation seems to be well suited to result in satisfactory robustness against JPEG compression even at higher compression ratios.

Note that the presented extraction algorithm does not only have to be robust against compression, but also sensitive towards intentional image alterations. Here, a higher robustness against compression may mean that the algorithm is no longer able to be sensitive enough against other malicious image alterations. In order to investigate the sensitivity of our proposed scheme against intentional or malicious image alterations we have removed the US Air Force flag from the plane512 image (see Fig. 4.21.b).

![Figure 4.21: Testimage plane512 original and under attack.](image)

In Table 4.8 we list the sensitivity results with respect to a chosen wavelet decomposition level. The wavelet decomposition level influences the ability of our algorithm to detect local image modifications significantly. Using a high value for wlev the local image modification is detected with a low number of feature values. At wlev 9, only 6 feature values are needed to detect the local attack.

As a consequence, there is the need for a compromise between the sensitivity against intentional image modifications on the one side, but robustness against JPEG2000 and JPEG
Table 4.8: Sensitivity against the removed flag.

<table>
<thead>
<tr>
<th>wlev</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>plane</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>13</td>
<td>29</td>
<td>28</td>
<td>101</td>
</tr>
</tbody>
</table>

compression on the other side. Regarding our results, we can say that a value for wlev of 5 or 6 seems to be best suited to be used for JPEG 2000 bitstream feature extraction. In this case, our method shows to be robust enough against compression up to a medium quality level, and the tested local attack can be detected with a rather low number of feature values. To give a concrete value based on these first results, we suggest to apply the hash function to the first 30 packet data bytes of the JPEG 2000 codestream to result in a robust authentication scheme.

Experiments: Sensitivity Results

We use classical 8bpp image data in our experiments, including the well known lena image at varying image dimensions (512 × 512, 1024 × 1024, and 2048 × 2048 pixels), the houses (see 4.23.a), the plane (see 4.22.a), the graves image (see 4.24.a), the goldhill image (see 4.22.c), and frame no. 17 from the surfside video sequence (see 4.25.a). In the following we present detailed results regarding the sensitivity towards different local image alterations and global Stirmark modifications:

- local: different intentional image modifications:
  - plane: plane without call sign (see figure 4.22.b)
  - graves: one grave removed (see figure 4.24.b)
  - houses: text removed (see figure 4.23.b)
  - goldhill: walking man removed (see figure 4.22.d)
  - surfside frame: twisted head (see figure 4.25.b)

- global: different Stirmark attacks (see www.cl.cam.ac.uk/~mgk25/stirmark/)

The experiments are conducted as follows: first, the feature values (i.e. packet data) are extracted from the JPEG 2000 codestream. Subsequently, the codestream is decoded and
the image alteration is performed. Finally, the image is again JPEG 2000 encoded using the coding settings of the original codestream and the feature values are extracted and compared to the original ones.

The results which are presented in the following show the number of feature values (in bytes) required to detect a global or local image modification. A value of - for instance - 42 means that the first 41 bytes of feature values are equal when comparing the computed features from the modified image to the feature values of the corresponding original image. The value itself can be easily interpreted: The higher the value, the more robust is the proposed method against the tested attack. In general, we want to see high values against JPEG2000 and JPEG compression, but low values against all other tested attacks. [50] showed that the feature extraction method is robust against moderate JPEG and JPEG2000 compression. In most cases, feature values of 50 or more were required for detecting JPEG and JPEG2000 compression ratios up to 1 or 0.8 bits per pixel. Here we want to detect all the described image alterations reliably. Therefore, we want to see significant lower feature values in all tests.
Table 4.9 lists the obtained results for the different local attacks with respect to a chosen wavelet decomposition level. The wavelet decomposition level obviously influences the ability of our algorithm to detect local image modifications. At a higher wlev parameter all local image modifications are detected with a low number of feature values. At wlev 9 for instance, only 7 feature values are needed to detect any of the tested local attacks. The modification of the graves image is detected with 2 feature values, in the plane image case only about 3 values are needed. At lower decomposition levels, more feature values are needed in general to detect the tested local image manipulations. At a wlev of 3, 412 feature values are needed to recognize the twisted head in the surfside frame, at wlev 4, only 68 are needed, and at the highest tested wlev, only about 6 are needed. Since the local changes are kept relatively small, the sensitivity regarding local image manipulations can be considered as high (depending on

<table>
<thead>
<tr>
<th></th>
<th>wlev9</th>
<th>wlev8</th>
<th>wlev7</th>
<th>wlev6</th>
<th>wlev5</th>
<th>wlev4</th>
<th>wlev3</th>
</tr>
</thead>
<tbody>
<tr>
<td>goldhill without man</td>
<td>7</td>
<td>7</td>
<td>28</td>
<td>44</td>
<td>29</td>
<td>48</td>
<td>155</td>
</tr>
<tr>
<td>houses without text</td>
<td>6</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>17</td>
<td>60</td>
<td>187</td>
</tr>
<tr>
<td>graves attacked</td>
<td>2</td>
<td>4</td>
<td>11</td>
<td>10</td>
<td>28</td>
<td>23</td>
<td>84</td>
</tr>
<tr>
<td>plane, no callsign</td>
<td>3</td>
<td>5</td>
<td>34</td>
<td>37</td>
<td>73</td>
<td>27</td>
<td>74</td>
</tr>
<tr>
<td>surfside, twisted head</td>
<td>6</td>
<td>17</td>
<td>7</td>
<td>20</td>
<td>2</td>
<td>68</td>
<td>412</td>
</tr>
</tbody>
</table>

Table 4.9: local attacks: different wlev used for feature extraction
the wavelet decomposition level) - which of course is desired.

The Stirmark benchmark is used to rate the robustness and efficiency of various watermarking methods. Therefore, numerous image attacks are defined including rotation, scaling, median filtering, luminance modifications, gaussian filtering, sharpening, symmetric and asymmetric shearing, linear geometric transformations, random geometric distortions, and others. More details about the different attacks can be downloaded from the web page [www.cl.cam.ac.uk/~mgk25/stirmark/](http://www.cl.cam.ac.uk/~mgk25/stirmark/), where the Stirmark testsetting is discussed at length. Our robust feature extraction method is tested against the standard Stirmark attacks, and due to the field of application our proposed method should be sensitive regarding all Stirmark attacks. In table 4.10 a selection of the obtained results against global modifications is listed. Here we see the sensitivity against Stirmark attacks with parameter i, b, as well as global luminance modifications.

<table>
<thead>
<tr>
<th>wlev</th>
<th>stirmark i=1</th>
<th>stirmark i=2</th>
<th>stirmark b=1</th>
<th>stirmark b=2</th>
<th>luminance+1</th>
<th>luminance+2</th>
<th>luminance+3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>6</td>
<td>7</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5</td>
<td>6</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>36</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>36</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>12</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 4.10: different attacks/lena512: different wlev used for feature extraction

Again the results are delivered with respect to a chosen wlev for feature extraction, and only the results for the lena image at a resolution of 512 × 512 pixels are given. We can observe a high sensitivity against the presented global image alterations. Except for a minimum change of the global luminance by a factor of 1, which shows a worse result. Nevertheless, the sensitivity is high enough - as desired. Interestingly, a lower wlev parameter also shows a higher sensitivity against the Stirmark attacks with parameter i and parameter b. This effect can also be seen in other Stirmark attacked images. For this reason, a lower wlev could be preferred to be used for the feature extraction algorithm, since a lower wlev is also more robust against JPEG2000 and JPEG compression. However, all the local attacks presented in table 4.9 could not be detected any longer when using such a low wlev parameter.

In table 4.11 and table 4.12 the results for the standard Stirmark testsetting is listed. Again, only results for the lena image at a resolution of 512 × 512 pixels are given with respect to a specific wlev. The first column of both tables clearly identifies the applied Stirmark attack and should be self-contained. Overall we can see that the sensitivity against all tested attacks is very high for a low and a high wlev value. For a wlev of 5 and 6, only the Gaussian filtering shows slightly higher feature values of about 36 and 23. Also a minor rotation and scale is slightly harder detectable. Here we need about 31 and 18 (wlev 5,6) feature values (see table 4.12 first data row). The results for the other testimages are similar and therefore not listed here. In general, the sensitivity regarding Gaussian filtering as well as slight rotations and scalings is slightly inferior as compared to the other Stirmark tests. Regarding the graves image, these two test attacks are detected at a lower number of feature values, since the graves image is more sensitive to any image modification than the other.
tested images.

There is the need for a compromise between the sensitivity against intentional image modifications on the one side, but robustness against JPEG2000 and JPEG compression on the other side. Regarding the robustness results in [50], a level of about 6 or 5 seems to be best suited to be used for JPEG2000 bitstream feature extraction. In this case, we see a good sensitivity against local and global image attacks, and robustness against JPEG2000 and JPEG compression up to moderate compression ratios.

**Application Scenarios**

Using parts of the JPEG2000 bitstream as robust visual features has important advantages, especially in the context of real world usability:

- Soft- and hardware to perform JPEG2000 compression will be readily available in large quantities in the near future which makes our proposed scheme a very attractive one (and also potentially cheap one).

- JPEG2000 Part 2 allows to use different types of wavelet transforms in addition to the Part 1 pyramidal scheme, in particular anisotropic decompositions and wavelet subband structures may be employed in addition to freedom in filter choice. This facilitates to add key-dependency to the hashing scheme by concealing the exact type of wavelet decomposition in use, which would create a robust message authentication code (MAC) for visual data. This could significantly improve the security against attacks (compare [44]).

- Most robust feature extraction algorithms require a final conversion stage to transform the computed features into binary representation. This is not necessary since JPEG2000 is of course given in binary representation.

We get two scenarios where our method can be applied in a straightforward manner: First, our method can be applied to any raw digital image data, via computing the JPEG2000 bitstream and then the JPEG2000 feature values. Second, any JPEG2000 bitstream can be used itself as starting point. In this case, the considered bitstream is the original data which should be protected, and the features are extracted directly from the investigated JPEG2000 bitstream. This scenario is useful, where some image capturing device directly produces JPEG2000 coded data instead of raw uncompressed data (i.e. JPEG200 compression implemented in hardware, no raw data saved).

After having extracted the feature values out of the JPEG2000 bitstream, three strategies may be followed:

- The extracted features are fed into the decoder stage of error correcting codes or linear codes to reduce the number of hash bits and to increase robustness. This approach has the advantage that different hash strings can be compared by evaluating the Hamming distance which serves as a measure of similarity in this case. Whereas it is desirable from the applications point of view to estimate the amount of difference between images
<table>
<thead>
<tr>
<th>Feature</th>
<th>wlev9</th>
<th>wlev8</th>
<th>wlev7</th>
<th>wlev6</th>
<th>wlev5</th>
<th>wlev4</th>
<th>wlev3</th>
</tr>
</thead>
<tbody>
<tr>
<td>17_row_5_col_removed</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1_row_1_col_removed</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>12</td>
<td>5</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>1_row_5_col_removed</td>
<td>6</td>
<td>4</td>
<td>15</td>
<td>2</td>
<td>12</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3x3_median_filter</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>13</td>
</tr>
<tr>
<td>5_row_17_col_removed</td>
<td>4</td>
<td>4</td>
<td>9</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>5_row_1_col_removed</td>
<td>4</td>
<td>4</td>
<td>7</td>
<td>7</td>
<td>5</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>5x5_median_filter</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>7x7_median_filter</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>9x9_median_filter</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>13</td>
</tr>
<tr>
<td>Gaussian_filtering_3_3</td>
<td>1</td>
<td>5</td>
<td>7</td>
<td>23</td>
<td>36</td>
<td>5</td>
<td>23</td>
</tr>
<tr>
<td>Sharpening_3_3</td>
<td>1</td>
<td>4</td>
<td>7</td>
<td>2</td>
<td>15</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>cropping_1</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>cropping_10</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>cropping_15</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>cropping_2</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>cropping_20</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>cropping_25</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>cropping_5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>cropping_50</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>cropping_75</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>flip</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>linear_1.007_0.010_0.010_1.012</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>linear_1.010_0.013_0.009_1.011</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>linear_1.013_0.008_0.011_1.008</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>ratio_x_0.80_y_1.00</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>ratio_x_0.90_y_1.00</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>ratio_x_1.00_y_0.80</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>ratio_x_1.00_y_0.90</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>ratio_x_1.00_y_1.10</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>ratio_x_1.00_y_1.20</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>ratio_x_1.10_y_1.00</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>ratio_x_1.20_y_1.00</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_-0.25</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_-0.50</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_-0.75</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_-1.00</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_-2.00</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_0.25</td>
<td>4</td>
<td>4</td>
<td>12</td>
<td>2</td>
<td>12</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_0.50</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>12</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_0.75</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>12</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_1.00</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_10.00</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_15.00</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_20.00</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_30.00</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_45.00</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_50.00</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rotation_90.00</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.11: standard stirmark testsetting, lena512: different wlev used for feature extraction
<table>
<thead>
<tr>
<th>rotation</th>
<th>scale</th>
<th>wlev9</th>
<th>wlev8</th>
<th>wlev7</th>
<th>wlev6</th>
<th>wlev5</th>
<th>wlev4</th>
<th>wlev3</th>
</tr>
</thead>
<tbody>
<tr>
<td>rotation</td>
<td>scale</td>
<td>0.25</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>18</td>
<td>31</td>
<td>4</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>0.50</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>16</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>0.75</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>1</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>1.00</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>2.00</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>0.25</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>0.50</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>0.75</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>1.00</td>
<td>3</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>10.00</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>15.00</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>2.00</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>30.00</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>45.00</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>5.00</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>rotation</td>
<td>scale</td>
<td>90.00</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>scale</td>
<td>0.50</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>scale</td>
<td>0.75</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>scale</td>
<td>0.90</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>scale</td>
<td>1.10</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>scale</td>
<td>1.50</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>scale</td>
<td>2.00</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>shearing</td>
<td>x</td>
<td>0.00</td>
<td>y</td>
<td>1.00</td>
<td>5</td>
<td>4</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>shearing</td>
<td>x</td>
<td>0.00</td>
<td>y</td>
<td>5.00</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>shearing</td>
<td>x</td>
<td>1.00</td>
<td>y</td>
<td>0.00</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>shearing</td>
<td>x</td>
<td>1.00</td>
<td>y</td>
<td>1.00</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>shearing</td>
<td>x</td>
<td>5.00</td>
<td>y</td>
<td>0.00</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>shearing</td>
<td>x</td>
<td>5.00</td>
<td>y</td>
<td>5.00</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4.12: standard stirmark testsetting, lena512: different wlev used for feature extraction
by using those hash functions, this property severely threatens security and facilitates “gradient attacks” by iteratively adjusting hostile attacks to minimize a change in the hash value.

- A classical cryptographic hash function (like MD-5 or SHA-1) is applied to the feature data to result in an overall robust but cryptographically secure robust visual hash procedure. The possibility to measure the amount of difference between two hash strings is lost in this case, however, gradient attacks and other security flaws are avoided.

- The extracted feature values are used as hash strings as they are without any further processing. The obvious disadvantages in terms of the higher amount of hash bits and lower security against attacks is compensated by the possibility to localize and approximately reconstruct detected image alterations since the hash string contains data extracted from a low bitrate compressed version of the original image.

In the latter case, with the available feature value data (consisting of JPEG2000 packet body data), and the corresponding packet headers which need to be generated and inserted into the codestream, the original image can be reconstructed up to the point the codeblock data is available in the packet bodies. A packet header indicates, among other information, which codeblocks are included in the following packet body, whereas the body contains the codeblocks of compressed data itself. Without the packet header, a reconstruction of the corresponding packet body is not possible in general. Therefore, these packet headers need to be inserted.

In figures 4.26 and 4.27 we visualize the approximations of the original images using feature value data of the lena and the graves image only. In each case, the first 512, 1024, and 2048 bits of feature values are used.

Since the given number of feature value bits which are used for the visual reconstruction include packet body data only, the overall number of bits used for reconstruction - including the needed packet header data - must be somewhat bigger. Table 4.13 shows the number of bits which are required for the corresponding images. The first column gives the number of feature bits used, and the entries in the table show the overall number of bits which are needed for the visual reconstruction. We see that a considerable number of “extra” bits are needed. These "extra bits” stem from the corresponding packet headers and are needed to reconstruct the image data up to the point where codeblock packet body data is given in the features.

<table>
<thead>
<tr>
<th>Feature Bits</th>
<th>lena512</th>
<th>graves512</th>
<th>plane512</th>
</tr>
</thead>
<tbody>
<tr>
<td>512 bits</td>
<td>552</td>
<td>552</td>
<td>552</td>
</tr>
<tr>
<td>1024 bits</td>
<td>1144</td>
<td>1136</td>
<td>1136</td>
</tr>
<tr>
<td>2048 bits</td>
<td>2224</td>
<td>2208</td>
<td>2224</td>
</tr>
</tbody>
</table>

Table 4.13: signature bits (including packet header data)

The number of feature bits used have been chosen in a way to demonstrate a possible application where the hash string could be signed using a digital signature algorithm like El Gamal or RSA. In this context, using a 512 feature bits signature already could help to
localize and approximately reconstruct severely manipulated regions in the image, whereas a 2048 feature bits signature allows to gain information about some details as well.

**Conclusion**

The JPEG2000 algorithm can be employed to extract robust features from an image. The presented method has shown to be robust against moderate JPEG2000 and JPEG compression. In this work we showed that the method is also very sensitive regarding global and local image alterations including Stirmark attacks and different intentional local image modifications. Application scenarios for our approach are discussed and show this method to be of interest for practical employment.
Chapter 5

Summary

Although the first applications of perceptual hashing technologies are becoming regulars in today’s services, not all aspects of perceptual hashing are considered appropriately. The focus (of the publicly available material) so far is on robustness. Security, however, seems to be a poor cousin.

WVL4 addresses a broad range of areas that requires further research. This includes the analysis and improvement of existing techniques. Furthermore, new approaches are developed, investigated and compared with existing technology. This ensures the continuous evolution of perceptual hashing technologies. In addition to that, security is identified as a topic that has not been considered adequately.

The general focus on robustness is caused by applications of perceptual hashing technologies: So far, these technologies are mainly applied in value added services. For these applications, robustness might be the most important applications. On the one hand attacks are less likely. For example, in broadcast monitoring or for the identification of unknown songs attacks are less likely. Additionally, a related monetary damage is limited. For emerging applications this will change. Among these applications are those which apply perceptual hashing techniques for content filtering in P2P-networks.

Thus, security of perceptual hashing technologies has to be considered more thoroughly, e.g. as described in this deliverable. Further steps toward a general security evaluation framework are necessary. Among the open questions that have to be investigated is the relation of security in identification applications and in authentication applications.
Bibliography


[27] IEEE. Ashwin Swaminathan, Yinian Mao, Min Wu, Mar Security of Feature Extraction in Image Hashing.


