IST-2002-507932

ECRYPT

European Network of Excellence in Cryptology

Network of Excellence

Information Society Technologies

D.WVL.16
Report on Watermarking Benchmarking
And Steganalysis

Due date of deliverable: 31. July 2007
Actual submission date: 30. September 2007

Start date of project: 1 February 2004 Duration: 4.5 years

Lead contractor: Katholieke Universiteit Leuven (KUL)

Revision 1.0

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Report on Watermarking Benchmarking And Steganalysis

Editor
Jana Dittmann (GAUSS)
Christian Kraetzer (GAUSS)

Contributors
Christian Kraetzer (GAUSS),
Elke Franz (GAUSS)
Jean-Luc Dugelay (CNRS)
Andreas Lang (GAUSS)

16 November 2007
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.

Contents

1 Abstract – Executive Summary 3
2 Introduction 2
3 Watermarking Benchmarking using SMBELL 3
3.1 Introduction of SMBELL v.1.0 3
3.2 Parameters 3
3.3 Usage of SMBELL 4
3.3.1 Using the help command 4
3.3.2 Creating and using an attack profile 4
3.3.3 Loading of an attack profile 5
3.4 Document type definition (DTD) 5
3.5 SMBELL-XML-Documents 6
3.6 Included libraries 7
4 Steganalysis and Detectability Benchmarking for Digital Watermarks 8
4.1 Analysing Correlations between Pixels 8
4.1.1 Introduction into Steganography, Steganalysis and pixel correlation computation 8
4.1.2 Strategy for Analysing Correlations 9
4.1.2.1 Correlations between Pixels 9
4.1.2.2 Basic Approach for the Analysis: Correlation Coefficient 10
4.1.2.3 Parameters for the Calculation of the Correlation Coefficient 11
4.1.3 Calculated Correlation Coefficients 12
4.1.3.1 Analysing Steganographically Unused Images 12
4.1.3.2 Analysing Stego Images 14
4.1.4 Analysing Correlation Coefficients for a Mixed Sequence 15
4.1.4.1 General Considerations and Theoretical Expectations 15
4.1.4.2 Possible Results in an Idealised Scenario 17
4.1.4.3 Estimation Based on Further Scans of the Image 19
4.1.5 Summary and Outlook 20
4.2 The AMSL Audio Steganalysis Toolset Version 1.03 22
4.2.1 Structure of the AAST 22
4.2.1.1 Pre-processing of the audio/speech data 22
4.2.1.2 Feature extraction from the signal 23
4.2.1.3 Post-processing of the resulting feature vectors 23
4.2.1.4 Analysis 24
4.2.2 Computed Features 24
4.2.2.1 Empirical variance 24
4.2.2.2 Covariance 24
4.2.2.3 Entropy 25
4.2.2.4 LSB ratio 25
4.2.2.5 LSB flipping rate 25
4.2.2.6 Mean of samples in time domain 25
4.2.2.7 Median of samples in time domain 25
4.2.2.8 Mel-Frequency Cepstral Coefficients (MFCCs) 25
4.2.2.9 Filtered Mel-Frequency Cepstral Coefficients (FMFCCs) 29

4.2.2.10 Pattern search 30
4.2.3 Supported Methods for inter-window analysis 30
4.2.4 Supported Output formats 30
4.2.4.1 The CSV format 30
4.2.4.2 The SVM format 30
4.2.4.3 The ARFF format 30
4.2.5 Usage of the AAST (version 1.03) 31
4.2.6 Practical Results 31
4.2.7 Summary 31
5 Further Benchmarking Activities in WVL3 32
5.1 3D Watermarking Benchmarking 32
5.2 Digital Watermarking and Perceptual Hashing of Audio Signals with Focus on their Evaluation 32
6 Summary 33
7 Bibliography 34
8 Acknowledgements 36
9 Appendix A – AAST (v.1.03) parameters 37
9.1 Mandatory Parameters 37
9.2 Optional Parameters 37

1 Abstract – Executive Summary

The focus of WAVILA WVL3s research activities is on the development of benchmarking tools and schemes for digital watermarking and steganography as well as the evaluation of selected algorithms.

Digital watermarking and steganography are two of the most important aspects of information hiding in digital media. While the first is most commonly used for authentification, proof of ownership, proof of integrity and non-repudiation mechanisms it is part of many Digital Rights Management schemes and has therefore a huge commercial interest. Steganography, as the second information hiding technique considered by WVL3, provides hidden communication channels in seemingly “harmless” media like images, audio material or VoIP telephony sessions and is therefore of huge interest for security considerations and for the development of steganalysis techniques to detect such hidden communication channels in their cover mediums.

Benchmarking itself has not only the possibilities to identify possible weaknesses of tested algorithms. It can also provide a fair comparison of different algorithms under different evaluation aspects, making it possible to identify from a list of given solutions the algorithm most fitting for a concrete application scenario.

In this report we introduce the results of WVL3s activities in audio watermarking benchmarking which led to the implementation of SMBELL, a management tool for the renown SMBA benchmarking suite developed ECRYPT partner GAUSS and provided as a commonly available tool to the ECRYPT consortium.

The second large part of WVL3s research activities described here is concerned with the results in audio and image steganalysis. In this field of research within WVL3 an approach to source modelling in image steganalysis and the AAST (AMSL Audio Steganalysis Toolset) are covered by this report.

Concluding this report new research areas, which arose during the considered reporting period, like 3D watermarking benchmarking and the combination of watermarking and perceptual hashing algorithms in evaluation, are presented briefly.

2 Introduction

In this report we summarise the WVL3 activities in the fields of a) benchmarking methods and tools for digital watermarks and b) steganography and steganalysis for the period M25-M42 of the ECRYPT project.
Based on the previous deliverable D.WVL.10, where the benchmarking suite Stirmark Benchmark for Audio was introduced in detail, this report will summarise consecutive work enhancing the SMBA by introduction of SMBELL. With the SMBELL module the only existing audio watermarking benchmarking suite becomes more suitable for the application by end users and for large scale automated testing.
Special attention is paid in this report on source modelling in image steganalysis. Pixels of natural images are not stochastically independent. Rather, there exist correlations between them. Embedding usually introduces some randomness into the cover media; existing correlations are not considered. Thus, it can be assumed that existing correlations between pixels are influenced by embedding. This report summarises results of analysing correlations between pixels in steganographically unused images as well as in stego images produced by a selected steganographic algorithm. More exactly, we focussed on correlations in noise extracted from images. The results can be used for further investigations on steganalytical methods as well as for assessing the effects of embedding techniques on existing correlations
As a further highlight the report introduces in detail the AMSL Audio Steganalysis Toolset (AAST) as a versatile tool for statistical analyses on audio signals. The AAST is used a measure of audio watermarking and steganography detectability.
Furthermore new research areas in WVL3, like 3D watermarking benchmarking and the combination of watermarking and perceptual hashing algorithms in evaluation are presented.

This report is structured as follows: Section 3 addresses the usage and application in audio watermarking benchmarking of the SMBELL software. Section 4 focuses on steganalysis and detectability benchmarking for digital watermarks. Here in subsection 4.1 first an approach on source modelling in image steganalysis. Subsection 4.2 describes in detail the AAST and its application.
Section 5 introduces additional benchmarking activities in WVL3 during the reporting period. Section 6 summarises the activities of WVL3 in the fields of steganalysis and watermarking benchmarking and thereby concludes this report.

3 Watermarking Benchmarking using SMBELL

Authors: Christian Kraetzer, Andreas Lang, Otto-von-Guericke University Magdeburg, Faculty of Computer Science, Department of Technical and Operational Information systems, 39106 Magdeburg, Germany, kraetzer@iti.cs.uni-magdeburg.de

This section describes the evaluation tool SMBELL developed by the research group Multimedia and Security at the Otto-von-Guericke University of Magdeburg, Germany. This tool is providing the established StirMark Benchmark for Audio (SMBA) [LANG2005ACM] with an own batch processing language and aims at increasing the performance of SMBA in large scale tests as well as improving the usability of the benchmarking suite.

3.1 Introduction of SMBELL v.1.0

SMBELL (v.1.0) is a wrapper tool fort he already established audio watermarking evaluation suite StirMark Benchmark for Audio (SMBA). This wrapper improves the usability of SMBA in large tests by allowing the tester to define attack profiles which can be saved in XML files. These XML files could be shared between different computers (e.g. for complexity evaluations).

3.2 Parameters

In the following table the parameters for SMBELL are listed and described.

Parameter		Description	Optional
-h	--help	Displays the help	Yes
-i*)	--input*)	Selection of the input files(s This parameter can be used repeatedly; either –i or –j has to be set	No
-j*)	--input_dir*)	Specifies an input directory for processing. All audio files in the directory will be used as input for SMBELL. This parameter can be used repeatedly; either –i or –j has to be set.	No
-o	--output_dir	Specifies the output directory for SMBELL. The output is the result of processing the input files with SMBA.	No
-a	--attack	Selection of the SMBA attack to be used in the evaluation. This parameter can be used repeatedly, the order of the given in the listing of the attacks will be preserved during execution. For a complete list of possible attacks and their descriptions please consult the SMBA documentation.	No
-d	--dtd	Output path for the DTD used. If no path is given the SMBELL directory is used.	Yes
-s	--save	Specifies the file in which the batch processing operations for SMBA are saved.	Yes
-f	--xmlfile	Specifies the file in which the attack profile in XML format is saved.	Yes
-g	--generate	This parameter determines whether the batch	Yes

		processing job using SMBA is executed or not.	
-x	--xml	Loads an attack profile from the specified XML file.	No

*) With the current version of SMBELL only audio files in WAV format can be evaluated.

In the execution of SMBELL two different modes can be used: the first mode (enabled with –x or --xml) expects the specification of a predefined attack profile in XML format. If this mode is used every other parameter used in the launching command is ignored. In the second mode, is used to generate XML attack profiles or to perform instant tests, the specification of input files/directories, output directory and attacks is expected. Attacks are specified in the form:

```
… -a AddBrumm:1234:1234 …
… --attack AddBrumm:1234:1234 …
```

3.3 Usage of SMBELL

In the following sections the usage of SMBELL is elucidated.

3.3.1 Using the help command

```
Smbell
smbell    -h
smbell    --help
```

With the commands introduced above the help text for SMBELL is shown.

3.3.2 Creating and using an attack profile

```
smbell -i /home/smbell/audio1/file1.wav
       -i /home/smbell/audio1/file2.wav
       -j /home/smbell/audio2/
       -j /home/smbell/audio3/
       -o /home/smbell/result/
       -a AddBrumm:1234:1234
       -a AddDynNoise:4321
       -f /home/smbell/xml/attack.xml
```

In this example attack profile two audio files (`file1.wav` and `file2.wav`) are specified as input for SMBELL. Additionally the directories `audio2` and `audio3` are parsed for further input files. The audio files modified by SMBA are written into the directory `result`. Two attacks (`AddBrumm` and `AddDynNoise`) are specified to be performed on each input file with the parameters given.
In the introduced example the attack profile is saved for later re-use in the file `attack.xml`.

In a second, less complex, example the SMBA call for one file is compared with the SMBELL attack profile describing the same operation.

SMBA:

```
read_write_stream -f /home/smbell/audio1/file1.wav | smfa
-AddBrumm 1234 1234 -p +s 44100 -c 2 -b 16 |
read_write_stream -f /smba_result/file1.wav -p -s 44100 -
c 2 -b 16
```

SMBELL:

```
smbell -i /home/smbell/audio1/file1.wav
       -o /home/smbell/
       -a AddBrumm:1234:1234
```

It can be seen that the SMBELL attack profile not only enables a more transparent signal handling but is also better readable for human users.

3.3.3 Loading of an attack profile

```
smbell -x /home/smbell/xml/attack.xml
       -d /home/smbell/dtd/smba.dtd
       -s /home/smbell/savedcommands.txt
       -g
```

This command will load the XML file `attack.xml` using the DTD `smba.dtd` and generate from the profile the SMBA run script `savedcommands.txt`. The parameter forces SMBELL to generate the SMBA run script without launching the evaluation process.

3.4 Document type definition (DTD)

The Document type definition (DTD) fort he XML file is generated by each start of SMBELL using SMBAs help function. Therefore changes in the development of SMBA, like the addition of new attacks ort he implementation of different parameters for existing attacks, do not influence SMBELL. The following example shows the DTD head and the description of two attacks (AddBrumm and AddDynNoise).

```
<?xml version='1.0' encoding='UTF-8'?>
  <!ELEMENT smba    (media , attack)>
  <!ELEMENT media   (audio+)>
  <!ELEMENT audio   EMPTY>
  <!ATTLIST audio   src CDATA #REQUIRED
                    dest CDATA #REQUIRED
                    samplerate CDATA #REQUIRED
                    channels CDATA #REQUIRED
                    bits CDATA #REQUIRED>
  <!ELEMENT attack  (
    AddBrumm?, AddDynNoise?, AddFFTNoise?, AddNoise?,
    AddSinus?, Amplify?, BassBoost?, BitChanger?,
  Compressor?, CopySample?, CutSamples?,
    DynamicPitchScale?, DynamicTimeStretch?, Echo?,
    Exchange?, ExtraStereo?, FFT_HLPassQuick?,
FT_Invert?,         FFT_RealReverse?, FFT_Stat1?,
FlippSample?, Invert?,        LSBZero?, Noise_Max?,
Normalizer1?, Normalizer2?,             Nothing?,
Pitchscale?, RC_HighPass?, RC_LowPass?,
  ReplaceSamples?, Resampling?, Smooth?, Smooth2?,
    Stat1?, Stat2?, TimeStretch?, VoiceRemove?,
    ZeroCross?, ZeroLength1?, ZeroLength2?,
ZeroRemove?)+>

  <!ELEMENT AddBrumm EMPTY>
```

```
  <!ATTLIST AddBrumm
    strength  CDATA #REQUIRED
    frequency CDATA #REQUIRED
    AP_p      CDATA #IMPLIED>
  <!ELEMENT AddDynNoise EMPTY>
  <!ATTLIST AddDynNoise
    strength  CDATA #REQUIRED>
...
```

The root element of the XML file is defined as the element “smba” with the children “media” and “attack”. The element “media” has a child named “audio” with further attributes characterising the audio data (source, destination, sample rate, number of channels, quantisation).
The child elements of node “attack” are generated during the construction of the DTD from the list of available attacks in SMBA. After defining all attacks, for each of the attacks the list of attributes/parameters is defined.
SMBELL is designed to handle batch processing jobs for watermark evaluation. Therefore at least one audio file has to be specified and at least one attack has to be selected. No constrains for the number of files and attacks are defined.

3.5 SMBELL-XML-Documents

The SMBELL document follow the DTD described above. An example XML file might look as follows::

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE smba SYSTEM "smba.dtd">
<smba>
  <media>
    <audio src="/music/b1.wav" bits="16" channels="2"
           samplerate="44100" dest="/result/out_b1.wav"/>
    <audio src="/music/b2.wav" bits="16" channels="2"
           samplerate="44100" dest="/result/out_b2.wav"/>
  </media>
  <attack>
    <AddBrumm strength="320" frequency="455"/>
    <AddNoise strength="120"/>
    <Nothing/>
    <Resampling samplerate="44320"/>
  </attack>
</smba>
```

As defined in the DTD the XML structure starts with a “smba” tag, followed by the “media” definitions (in the example above two audio files `b1.wav` and `b2.wav`). Then the list of attacks to be performed on each signal in the “media” list is defined (in the example the `AddBrumm`, `AddNoise`, `Nothing` and `Resampling` attacks with their parameters).

3.6 Included libraries

The SMBELL is implemented C++ for Linux/Unix systems. It includes for the handling of XML-files the *libxml2* [LIBXML2] (version 2.6.26). For the handling of audio material (accessing audio files and extracting signal information like sample rate, quantisation, number of channels etc) the C++ library *libsndfile* [LIBSNDFILE] (version 1.0.17) is used. The parsing of command line parameters is done with the *getopt* [GETOPT] library and directory access with *dirent* library [DIRENT].

4 Steganalysis and Detectability Benchmarking for Digital Watermarks

Steganalysis, although its primary goal is focussed more on the detection of hidden communication, rather than the detection of a watermark embedding, can be a very useful tool for detectability benchmarking as shown in section 4.2.6.

Basically a steganographic system is considered to be insecure in practice if there is an algorithm that can decide with probability better than random guessing whether intercepted data contains a hidden message or not. Many application scenarios for digital watermarking have the same requirements in terms of undetectability (perceptual as well as statistical) as steganography algorithms, namely: having a high perceptual transparency and being statistical undetectable. Therefore a good universal steganalysis technique (relying on an appropriate modelling of source distributions) has to be considered an appropriate detectability measure in applied watermarking benchmarking.

In the following subsections we will focus first on analysing correlations between pixels of steganographically unused images and stego images. Within this analysis, difference vectors are considered, calculated by computing difference images between each of the images of the set to be analysed and an average image. We focussed on difference vectors since we want to use them as an estimation of the noise present in images. The presented analysis of source characteristics is an important fundamental step to any statistical evaluation of watermarked material.

The second large part in this section focussed on steganalysis and detectability benchmarking for digital watermarks gives a description of the composition, implementation and usage of the AAST (AMSL Audio Steganalysis Toolset) in its current version v.1.03. This steganalysis toolset has already been successfully applied in watermarking detectability benchmarking. Details on this are given in subsection 4.2.6.

4.1 Analysing Correlations between Pixels

Author: Elke Franz, TU Dresden, Faculty of Computer Science, Institute of Systems Architecture, D-01062 Dresden, Germany, Elke.Franz@tu-dresden.de

4.1.1 Introduction into Steganography, Steganalysis and pixel correlation computation

Steganography is a method for confidential communication that has been used since ancient times. While cryptography allows protecting the confidentiality of a message, steganography additionally hides the mere existence of the message.

Generally, a secret message (*emb*) is embedded into inconspicuously looking data (*cover*). The result of this operation, called *stego* data, is transmitted to the recipient who can extract the secret message. According to Kerkhoffs' principle, the security of such system should not depend on the secrecy of the algorithm but on keys used to parameterize the process.

There is a variety of steganographic algorithms using different media types as cover objects, e.g., digital images, audio files, or video. Within our analysis, we considered digital grey scale images as cover objects.

The goal of steganalysis is to detect whether an intercepted object contains embedded messages, i.e., whether it was produced by a steganographic system. A steganographic system is considered to be insecure in practice if there is an algorithm that can decide with probability better than random guessing whether intercepted data are stego data or not. Steganalysis often exploits the fact that steganographic algorithms characteristically change features of cover data. There are mainly two different steganalytical strategies: Targeted attacks are tailored to a specific steganographic algorithm. From analysing the algorithm, expectations about characteristical traces are derived. In contrast, universal attacks are based on the use of blind classifiers. The classifiers are trained by analysing a number of steganographically unmodified images and stego images. During the last years, many features and their modifications due to embedding have been investigated, including also analysis of features describing dependencies between pixels or coefficients (e.g., [FrGS_03, Frid_04, HeHQ_05, HoFV_05, LyFa_05]).

This report summarises results of analysing correlations between pixels by means of calculating correlation coefficients (based on [Penn_05]). More precisely, we analysed correlations in noise introduced by each digitalisation process such as scanning. The analysis was motivated by investigations to describe noise introduced by scanning and to mimic this noise while embedding (stego algorithm MimicNoise, [FrSc_05]): Differences between subsequently scanned images were calculated in order to assess the random part of noise introduced by scanning. The mean and variance of these differences were used as means to describe the noise. While embedding, a noise signal generated according to these parameters was added to the cover after a pre-processing step which aims to reduce existing noise. Since this basic approach did not consider possibly existing correlations, we aimed at investigating correlations in noise extracted from scanned images as well as from stego images. Our expectation is that the results differ, what could be probably exploited for steganalysis.

Within our analysis, we worked with sequences of images delivered by repeated scanning to compute correlations between differences of these steganographically unused images. Furthermore, we performed similar analysis for sequences of stego images. The analysis can be used for further investigations in analysing correlations in images. After some basic ideas about correlations, we introduce our general approach and the settings of the parameters used for the analysis. Practical results are presented afterwards. Finally, we give a summary and outlook on future analysis.

4.1.2 Strategy for Analysing Correlations

4.1.2.1 Correlations between Pixels

It is a known fact that pixels of natural images are not stochastically independent; this fact is, e.g., the basis for image compression [GoWo_02]. An image without correlation between pixels would be an array of stochastically independent random values. The correlations decrease with growing distance between the pixels.

Correlations depend on the image content, but they can be additionally introduced by the image acquisition process. Usually, digital cameras or scanners deliver digital images (we do not consider computer generated images). They use CCD (charge coupled device) elements to measure the incoming light as electrical charges which are first converted into voltages and digitised afterwards. Optical elements are also required in this process. Further image processing steps can be performed, e.g., in order to enhance image quality. During this process, correlations can be introduced if processing the results of the single elements is not completely independent from processing results of adjacent elements.

Considering digital cameras, correlations are also introduced by colour interpolation: In a typical digital camera, each CCD element measures only one of the three basic colours red, green, or blue. Colour interpolation is necessary to determine the remaining colours from neighboured pixels. However, this aspect is not relevant for our investigations since we examined images delivered by a flatbed scanner. Scanners do not have to apply colour interpolation because for each pixel all colours are measured: A scanner digitises analogue images using a CCD line sensor. Usually, the line sensor consists of three lines, one for each basic colour.

Within our analysis, we calculated correlation coefficients as a measure for correlations. Since correlation coefficients can only be computed between vectors containing several realisations of the random variables, we analysed sequences of images. A sequence consists either of a number of scans of one and the same analogue images or of a number of stego images generated by repeated embedding into one and the same cover image. For each pixel, the sequence delivers a number of realisations, i.e., a *pixel vector*.

Regarding such sequences, mechanical irregularities of the scanner might introduce further correlations: There can be minor differences between the exact scan positions of repeated scans even if the position of the analogue image on the scanner's platen is not changed. Such shifts will not be relevant in homogenous areas of an image. However, there will be dependencies between the values of pixel vectors on colour edges: Given the case that the raster of the first scan is placed on a colour edge between the two colours $colour_1$ and $colour_2$. There will be no absolutely sharp colour edge between $colour_1$ and $colour_2$; thus, the pixels contain parts of both colours. If the raster is shifted a bit, one of the pixels becomes brighter while the other one becomes darker. Thus, we expect that shifts mainly influence correlations between pixels at colour edges.

Each digitalisation process introduces noise which is inherently present in digital images. This noise mainly consists of two different parts: temporal and spatial noise. Spatial noise is a relatively stable part which is present in all images delivered by one device, what makes it useful for forensic investigations [FrGL_05]. On the other hand, temporal noise is a stochastically independent, random noise. If one tries to reduce noise, it will not be possible to clearly separate the two parts and, thus, the extracted noise will not be stochastically independent.

To estimate the noise present in scanned images, we first calculated the average of n scans of one and the same analogue image. The resulting image was used as an estimation of the "original" image. Difference images between the estimated original and the scans delivered estimations of noise introduced by scanning.

4.1.2.2 Basic Approach for the Analysis: Correlation Coefficient

A known statistical measure for correlations between two random variables x, y is the correlation coefficient $\rho_{x,y}$, which is computed according to (e.g., [Ochi_90]):

$$\rho_{x,y} = \frac{Cov[x,y]}{\sigma_x \sigma_y} \qquad \text{with} \qquad Cov[x,y] = E[xy] - E[x]E[y]$$

(1)

The correlation coefficient is a measure for describing a linear dependency between two random variables. It ranges from -1 to 1; values close to 0 indicate that there is no linear dependency between the random variables. Values close to -1 indicate a strong negative correlation while values close to 1 indicate a strong positive correlation.

Correlation coefficients can only be calculated between vectors containing realisations of the random variables. As mentioned before, we used sequences of *n* images delivered by scanning one and the same image *n* times without changing any scan parameter or by repeated embedding in one and the same cover image. In the following, $p_i(x,y)$ represents the pixel of image *i* at position (x,y) while $\boldsymbol{p(x,y)}$ represents the vector of pixels of all images at this position:

$$\boldsymbol{p(x,y)} = \{p_i(x,y) \mid i = 1, 2, \dots n\}$$

(2)

For each pixel, we can now compute a difference vector $\boldsymbol{d(x,y)}$ as difference of the scan to the estimated "original image":

$$\boldsymbol{d(x,y)} = \boldsymbol{p(x,y)} - \overline{p(x,y)} \qquad \text{with} \qquad \overline{p(x,y)} = \frac{1}{n}\sum_{i=1}^{n} p_i(x,y)$$

(3)

Generally, the differences are an estimation of the noise. A more detailed discussion is given in the following sections. We can compute the correlation coefficient $\rho_{d(x_1,y_1),d(x_2,y_2)}$ between two difference vectors $\boldsymbol{d(x_1, y_1)}$, $\boldsymbol{d(x_2,y_2)}$ according to:

$$\rho_{d(x_1,y_1),d(x_2,y_2)} = \frac{E[(\boldsymbol{d(x_1,y_1)} - \mu(\boldsymbol{d(x_1,y_1)}))(\boldsymbol{d(x_2,y_2)} - \mu(\boldsymbol{d(x_2,y_2)}))]}{\sigma(\boldsymbol{d(x_1,y_1)})\sigma(\boldsymbol{d(x_2,y_2)})}$$

(4)

where μ represents the mean and σ the standard deviation of the vectors. Thus, we investigated correlations existing in noise estimated by the difference vectors.

4.1.2.3 Parameters for the Calculation of the Correlation Coefficient

By calculating correlation coefficients according to (4), we aimed at investigating our assumption regarding existing correlations in the estimated noise. However, computing correlation coefficients between all difference vectors of an image would deliver a huge amount of data: There are $num_{pv} = m \cdot n$ difference vectors for an image with *m* rows and *n* columns; all in all, $\frac{1}{2}(num_{pv}(num_{pv} - 1))$ correlation coefficients could be calculated between these difference vectors. We used scans of photographs of size 9x13 cm; scanning such an image with commonly used resolution of 200 dpi yields an image of about 1000 by 700 pixels, i.e., there would be 700 000 difference vectors and 244 999 650 000 correlation coefficients.

However, it is not necessary to compute all these coefficients. Two parameters can be used to reasonably decrease the set of difference vectors to be considered:

- *distance* between the difference vectors and

- *colour difference* between the pixel vectors.

Since dependencies between pixels decrease with increasing distance, we focussed on pixels up to a certain distance at maximum. This maximum was set to 3 according to empirical tests.

According to our consideration regarding the influence of small shifts between repeated scans, we expected stronger correlations between pixels of different grey scales than between pixels of nearly the same grey scale. Thus, we focussed on computing correlations between pixels of different grey scales. As threshold for the grey scale difference, we used 40. The grey scale difference between two difference vectors was computed as the difference between the averages of the pixel vectors.

To summarize, we calculated correlation coefficients between two difference vectors $\boldsymbol{d(x_1, y_1)}$, $\boldsymbol{d(x_2,y_2)}$ only when the corresponding pixel vectors $\boldsymbol{p(x_1, y_1)}$, $\boldsymbol{p(x_2,y_2)}$ fulfilled the following two conditions:

(C1) The difference between the average grey scale of the pixels is at least 40:

$$\overline{\boldsymbol{p(x_1, y_1)}} - \overline{\boldsymbol{p(x_2, y_2)}} \geq 40$$

(C2) The distance between the vectors is at most 3:

$$\left(|x_1 - x_2| \leq 3\right) \wedge \left(|y_1 - y_2| \leq 3\right)$$

Even if the number of pixel vectors was limited by this way, there are still enough correlation coefficients per image (in the analysis reported in [Penn_05], about 1 000 000 correlation coefficients were calculated per image). Within our analysis, we calculated the mean μ_{cc} of all computed *c*orrelation *c*oefficients to get an impression of correlation coefficients between the difference vectors.

4.1.3 Calculated Correlation Coefficients

4.1.3.1 Analysing Steganographically Unused Images

First, we analysed correlation coefficients in sequences of steganographically unused images according to the basic approach described above. The procedure is summarised in Figure 1. More precisely, we can describe the analysed values as follows:

$$scan_i = orig + n_i$$

where *orig* means "original image without noise" and n_i stands for scanner noise in $scan_i$. Averaging delivers an estimation of the original image since it reduces noise:

$$av = \frac{1}{n}\sum_{i=1}^{n} scan_i \;; \qquad av \sim orig$$

Finally, the calculated difference images $dscan_i$ are an estimation of the scanner noise n_i:

$$dscan_i = scan_i - av \sim n_i.$$

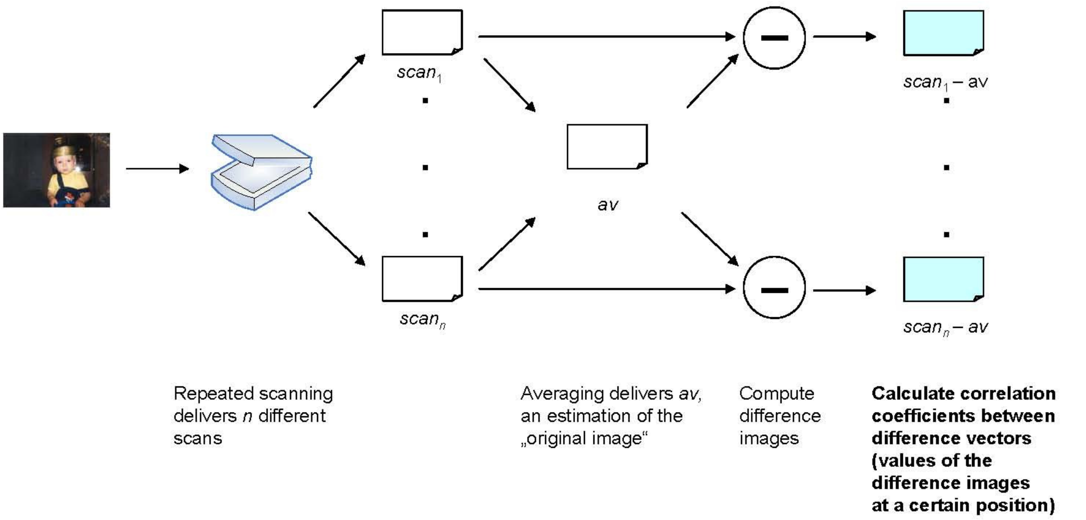


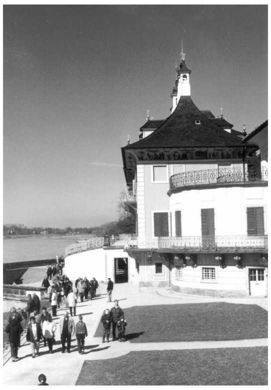
Figure 1: Calculating correlation coefficients for a sequence of scanned images.

For our tests, we used grey scale images of different characteristic. Figure 2 shows the test images which were used for calculating the results summarised in this report. Each of these test images was scanned 10 times.

test image 1

test image 2

test image 3



test image 4

test image 5

Figure 2: Test images used within the analysis.

To reduce the computing effort, we partitioned the images in blocks of 20x20 pixel before analysing them. As a result, some of the correlations were not considered. However, this does not affect the general result as some further tests have shown: Repeating the analysis for a shifted partitioning did yield very similar results (differences between the average correlation coefficients are less than 10^{-8}).

Furthermore, there are different possibilities regarding the distances. It is either possible to calculate all correlation coefficients with distance up to 3 at once, or to calculate all correlation coefficients for the considered distances separately and evaluating these results.

The results of this analysis are summarised in Table 1. Even if there are no strong linear dependencies, the absolute values of the calculated correlation coefficients are significantly greater than zero.

Table 1: Average correlation coefficients calculated for steganographically unused images.

	Distance between difference vectors			
	1	**2**	**3**	**1 – 3**
test image 1	0.404757	0.337123	0.338672	0.346819
test image 2	0.157886	0.184034	0.165346	0.170893
test image 3	0.366752	0.353175	0.347718	0.352135
test image 4	0.369429	0.368272	0.366900	0.367702
test image 5	0.309731	0.274670	0.247304	0.265092

4.1.3.2 Analysing Stego Images

Analysing stego images was done in a similar manner. As mentioned in Section 1, we focussed on analysing stego images produced by the algorithm MimicNoise [FrSc_05]. This algorithm first reduces noise before embedding by averaging a number of scans. The result of this averaging, *est* (*est*imation of original image), is than used for embedding. The message is embedded by adding a noise signal which should adhere to noise characteristics measured before. Embedding itself can be done, e.g., according to stochastic modulation [FrGo_03].

For our analysis, we produced a sequence of stego images for each of the test images using this approach. We worked both with the original approach and with adding just a Gaussian noise signal to *est* due to simplicity. There were no significant differences in the results.

The whole procedure is shown in Figure 3. A stego image is considered as

$$stego_i = est + sn_i \qquad \text{with } est = \frac{1}{n}\sum_{i=1}^{n} scan_i \sim orig\,.$$

The variable sn_i stands for the stego noise introduced by embedding. Averaging the stego images delivers an estimation of the cover image *est*:

$$av = \frac{1}{n}\sum_{i=1}^{n} stego_i \sim est\,.$$

Thus, the calculated difference images $dstego_i$ are an estimation of the stego noise sn_i:

$$dstego_i = stego_i - av \sim sn_i.$$

The resulting correlation coefficients can be expected to be close to 0, since we analyse correlations in stego noise which is stochastically independent.

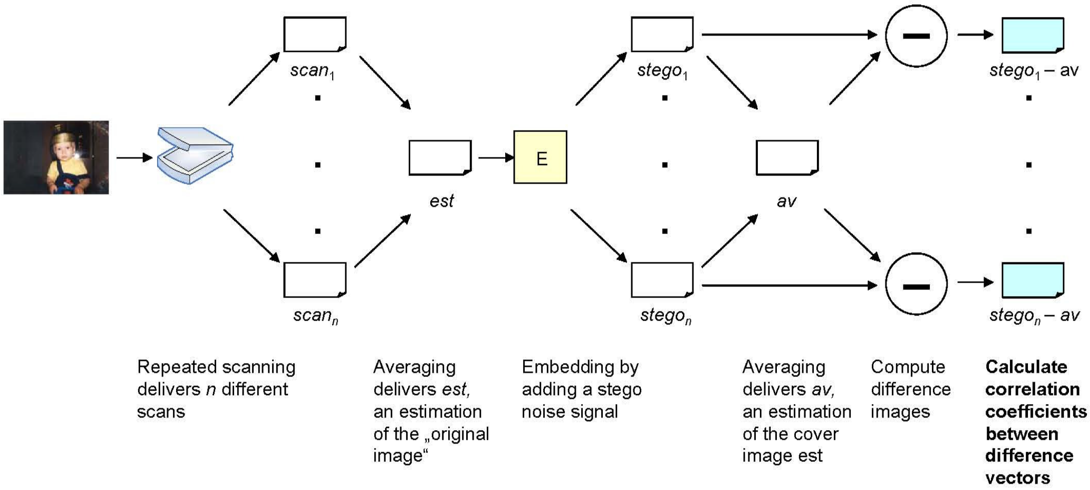


Figure 3: Calculating correlation coefficients for a sequence of stego images.

This expectation was confirmed by our tests (Table 2). Of course, analysing a sequence of stego images produced by embedding into a scan instead into a noise reduced version delivers similar results (this case was also tested but is not reported here).

Table 2: Average correlation coefficients calculated for stego images.

	Distance between difference vectors			
	1	**2**	**3**	**1 – 3**
test image 1	$4.0876 \cdot 10^{-4}$	$7.0388 \cdot 10^{-5}$	$-6.1315 \cdot 10^{-5}$	$4.6312 \cdot 10^{-5}$
test image 2	$-3.9078 \cdot 10^{-4}$	$4.5933 \cdot 10^{-4}$	$-3.6518 \cdot 10^{-4}$	$-8.2656 \cdot 10^{-5}$
test image 3	$-4.0351 \cdot 10^{-4}$	$2.8892 \cdot 10^{-4}$	$-2.1756 \cdot 10^{-6}$	$4.7377 \cdot 10^{-5}$
test image 4	$5.1061 \cdot 10^{-4}$	$-5.2145 \cdot 10^{-4}$	$4.0930 \cdot 10^{-5}$	$-9.4596 \cdot 10^{-5}$
test image 5	$-2.5038 \cdot 10^{-4}$	$-1.3868 \cdot 10^{-4}$	$-1.6872 \cdot 10^{-4}$	$-1.6879 \cdot 10^{-4}$

As expected, there are significant differences between correlation coefficients computed for a sequence of scanned images and correlation coefficients computed for a sequence of stego images. The results achieved so far could be used to assess to which degree a steganographic system can mimic realistic scanner noise.

However, they cannot be used for steganalysis since an attacker does not possess a sequence of images for his analysis; he rather can only analyse a single intercepted image. It remains open to analyse to which degree features that can be computed from single images reflect these results, especially features considering relations between pixels. Despite these investigations which are subject of future work, the next section discusses another general possibility for a steganalytical approach.

4.1.4 Analysing Correlation Coefficients for a Mixed Sequence

4.1.4.1 General Considerations and Theoretical Expectations

As mentioned above, since an attacker does not possess a sequence of images, he cannot apply an analysis like described here. In the best case, he could be able to generate a sequence of stego images. However, just calculating correlation coefficients for this sequence would not help for analysing correlations of the image he would like to analyse.

But what would happen if one of the images of the generated sequence is replaced by the image to be analysed? In case the image under investigation is also a stego image, the resulting correlation coefficients will again be close to zero. In case the image is a scan, there would be correlations which might introduce the results.

We checked the assumption that replacing one value of a vector for all pairs of vectors influences the average correlation coefficient with generating a number of random, stochastically independent vectors. First, we calculated the average correlation coefficient μ_{cc} between pairs of these vectors; second, we exchanged one of the values of the second vector of each pair to introduce correlations. To assess what can be expected in the best possible case, we introduced really strong correlations by just adding a constant to the appropriate value of the first vector. After that, we again calculated the average correlation coefficient μ_{cc}^* for these mixed vectors. The ratio $r_{cc} = \left|\frac{\mu_{cc}^*}{\mu_{cc}}\right|$ gives an impression how much the average correlation coefficient is influenced by exchanging one of the stochastically independent values. Results of this test for a vector size of 10 values are presented in Table 3.

Table 3: Average correlation coefficients for mixed and random vectors.

		Number of analysed pairs of vectors					
		10	**100**	**1 000**	**10 000**	**100 000**	**1 000 000**
test 1	μ_{cc}	0.0085	0.0020	0.0041	0.0033	-0.00002	0.0003
	μ_{cc}^*	0.0940	0.0417	0.0707	0.0689	0.0656	0.0647
	r_{cc}	11.0456	20.4609	17.1778	21.0061	2977.9600	239.1980
test 2	μ_{cc}	0.1133	0.0108	0.0003	0.0020	-0.0003	-0.0002
	μ_{cc}^*	0.2534	0.0530	0.0700	0.0705	0.0643	0.0641
	r_{cc}	2.2370	4.8981	254.7210	35.3877	205.6890	352.8340
test 3	μ_{cc}	0.0666	0.0582	-0.0007	0.0011	0.0024	-0.00004
	μ_{cc}^*	0.1172	0.1462	0.0635	0.0669	0.0662	0.0644
	r_{cc}	1.7589	2.5106	92.2378	59.3952	26.9105	1468.7800

Of course, the size of the vectors as well as the number of analysed vectors strongly influences the results. In all cases, the ratio is greater than 1, i.e., the average correlation coefficient increases after replacing even one stochastically independent value. There can be greater differences in the average correlation coefficients calculated for the random vectors due to their randomness. However, the average correlation coefficients for the mixed vectors become more similar with a growing number of analyzed pairs of vectors.

When we analyse images, the number of analysed pairs tends to 1 000 000 (Sec. 4.1.2.3). However, in contrast to the tests performed here, the correlations in scanner noise are not as strong as the correlations introduced in this little test. Moreover, the values surely depend on the image content. Thus, we cannot expect that μ_{cc}^{*} will become similar for different images. Furthermore, it still needs to be clarified how an attacker could generate the necessary sequence of images.

A first idea could be to generate a sequence of stego images by repeated embedding into the image under investigation *img*. However, this will not provide a possibility for a reasonable analysis: Correlation coefficients calculated for the generated images would be close to zero as expected since the difference images are estimations of the embedded stego noise; but even if *img* is a scan, replacing one of the generated images with it will not increase correlation coefficients. A closer look at the analysed data reveals the reason. We consider the image as

$$img = orig + n_i$$

and the generated stego images as

$$stego_i = img + sn_j = orig + n_i + sn_j.$$

If we average a mixed sequence, the difference images will still be just an estimation of the stego noise since all images contain the same representation of scanner noise. Consequently, the average correlation coefficient will not increase. Hence, what we need is a sequence of stego images that do not contain the representation of scanner noise inherit in the image under investigation.

4.1.4.2 Possible Results in an Idealised Scenario

Before considering how an attacker should generate a suitable sequence of stego images, we analysed what results can be expected at all. According to the assumptions regarding the analysed steganographic algorithm, this would require that the attacker is able to apply a noise reduction method on *img* so that he receives an estimation *est* of the original image without noise. Repeated embedding into this image yields the required sequence of stego images (Figure 4).

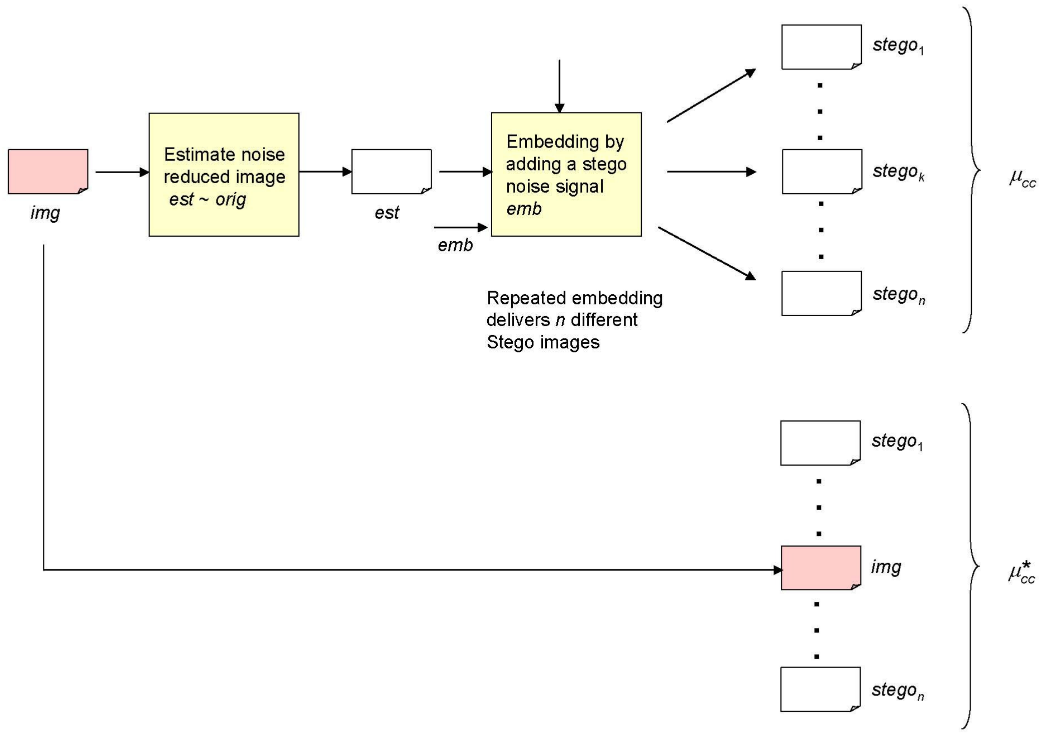


Figure 4: Adapted analysis for comparing correlation coefficients in cover and stego images.

The difference images calculated from the mixed sequence would contain an estimation of the scanner noise containing correlations if *img* is a steganographically unused image. According to the expectation described above, the average correlation coefficient of the mixed sequence μ_{cc}^{*} would be greater than the average correlation coefficient calculated from the sequence of stego images μ_{cc}: All vectors which are used for calculating the correlation coefficients are modified in the mixed sequence; one of their values is replaced by values which are correlated. Thus, the correlation coefficients are expected to increase in this case even if it will be less than an average correlation coefficient computed for a set of scanned images only.

On the other hand, if *img* was a stego image, replacing one of the images will not introduce correlated values; consequently, the correlation coefficients are not expected to increase. Thus, in case of $\mu_{cc}^{*} > \mu_{cc}$, the attacker assumes that *img* was not steganographically used, in case of $\mu_{cc}^{*} \sim \mu_{cc}$, he assumes *img* to be a stego image.

In this idealised scenario, we just assumed that the attacker would have been able to calculate *est*. Actually, we used a sequence of stego images delivered by repeated embedding into *est*. For the mixed sequence, we replaced on of the stego images by one of the scans used to calculate *est*. Results of this analysis are summarised in Table 4. The table shows the ratio r_{cc} for the average correlation coefficients calculated for all difference vectors up to the maximum distance.

Table 4: Ratio r_{cc} of average correlation coefficients for images – idealised scenario.

	Correlation coefficients for distance 1-3		
	μ_{cc}	μ^*_{cc}	r_{cc}
test image 1	$4.0876 \cdot 10^{-4}$	-0.010862	234.5400
test image 2	$-3.9078 \cdot 10^{-4}$	0.008879	107.4210
test image 3	$-4.0351 \cdot 10^{-4}$	0.003194	67.4167
test image 4	$5.1061 \cdot 10^{-4}$	$9.3577 \cdot 10^{-4}$	9.8923
test image 5	$-2.5038 \cdot 10^{-4}$	0.028649	169.7420

In practice, calculating *est* is rather difficult. Since the attacker does only possess a single image, he cannot calculate *est* by averaging. Another well-known approach to reduce noise is to apply a suitable filter. However, this might be also complicated: A filter usually estimates the resulting value from adjacent pixels. Consequently, especially grey edges are problematic to handle; but they are mainly considered in our analysis. Some first tests using the wavelet based filter introduced in [MKRM_99] did not yield satisfactory results: the average correlation coefficient did not increase for a mixed sequence containing a steganographically unused image.

4.1.4.3 Estimation Based on Further Scans of the Image

As one possibility to calculate *est*, one could consider the case that an attacker first generates a reproduction of the analogue image from the intercepted image. Due to the increasing popularity of digital cameras, there are meanwhile a lot of such services available. Subsequently, he scans this image *n* times and calculates *est* by averaging the *n* scans. We worked again under idealised assumptions in order to assess what would be possible in the best case.

More precisely, we scanned the same photo again 10 times using the same scanner. The resulting sequence of scans was used to calculate *est'*, in which we repeatedly embedded in order to generate the sequence of stego images needed for the analysis. Now we could test two cases (Figure 5):

(1) *img* is a stego image and

(2) *img* is a steganographically unused image.

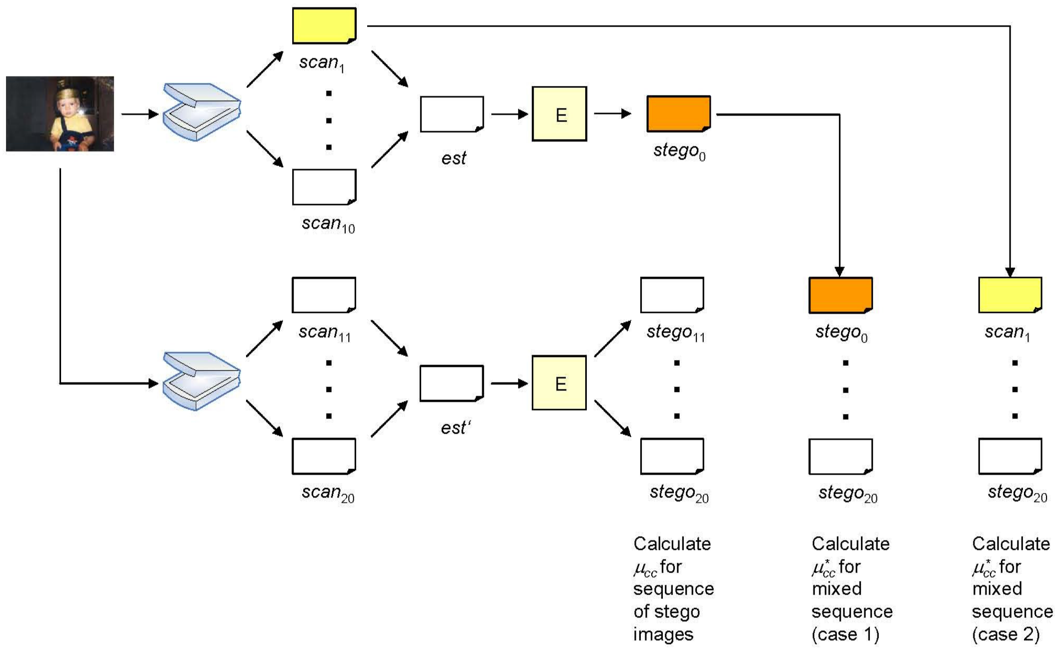


Figure 5: Adapted analysis based on further scans.

As the results in Table 5 show, the correlation coefficients for the mixed sequence increase in both cases. However, if *img* is a scan, it increases much more than if *img* is a stego image.

Table 5: Ratio r_{cc} of average correlation coefficients for images – using further scans of the image.

		Case 1		**Case 2**	
	μ_{cc}	μ^*_{cc}	r_{cc}	μ^*_{cc}	r_{cc}
test image 1	$1.3963 \cdot 10^{-6}$	-0.004284	3068.1100	-0.013889	9947.0000
test image 2	$5.1604 \cdot 10^{-5}$	0.002125	41.1782	0.011058	214.2820
test image 3	$-8.4979 \cdot 10^{-7}$	0.015177	17859.7000	0.049375	58102.6000
test image 4	$-7.4884 \cdot 10^{-5}$	0.003292	43.9613	0.412316	5506.0600
test image 5	$-2.4720 \cdot 10^{-4}$	0.001264	5.1133	0.031187	126.1610

Using such an approach as a steganalytical method is only possible if a decision can be made how strong the average correlation coefficient is expected to increase, perhaps depending on the image content or on the average correlation coefficient calculated for the sequence of stego images. Additionally, there are two further important issues which must be investigated: First, it is necessary to repeat the analysis with a real reproduction of the analogue image. Second, the analysis needs to be done for a comprehensive set of test images.

4.1.5 Summary and Outlook

This document summarises first results on analysing correlations between pixels of steganographically unused images and stego images. Within this analysis, we have considered difference vectors, calculated by computing difference images between each of the images of the set to be analysed and an average image. We focussed on difference vectors since we want to use them as an estimation of the noise present in images.

Altogether, the expectations regarding correlations calculated for a set of scan images and for a set of stego images were confirmed in the tests. Correlation coefficients for a mixed sequence also increase according to the expectations if the image under investigation is a steganographically unused image.

Currently, the results can aim to assess possible effects of embedding. In the ideal case, the correlation coefficients calculated for a set of stego images should be similar to correlation coefficients calculated from a set of scans.

As pointed out, we worked with idealized conditions in the tests reported here. The goal was to get an impression about possible results at all. Further investigations are necessary in order to check whether similar results can be achieved in a realistic scenario.

It is also a topic of ongoing work to analyse correlation coefficients between pixel vectors and studying their implications on features that can be calculated from single images. In contrast to the investigations done so far, several relations are considered separately.

4.2 The AMSL Audio Steganalysis Toolset Version 1.03

Author: Christian Kraetzer, Otto-von-Guericke University Magdeburg, Faculty of Computer Science, Department of Technical and Operational Information systems, 39106 Magdeburg, Germany, kraetzer@iti.cs.uni-magdeburg.de

The AAST (AMSL Audio Steganalysis Toolset) in its current version v.1.03 is a versatile pre-processing and statistical analysis suite for audio material. It has been developed since 2005 (under different names) by the Advanced Multimedia Security Lab (AMSL) at the faculty of computer science of the Otto-von-Guericke University of Magdeburg, Germany. As can be seen in numerous publications, this steganalysis toolset has already been successfully applied so far in universal and application specific steganalysis, watermarking detectability benchmarking and media forensics for microphone and environment classification.

4.2.1 Structure of the AAST

The AAST consists of the following four modules:
1. pre-processing of the audio/speech data
2. feature extraction from the signal
3. post-processing of the resulting feature vectors (for intra- or inter-window analysis)
4. analysis (classification for steganalysis)

A visualisation of these four modules with a list of possible operations in each module can be found in figure 6. In the following sections these modules are described in more detail.

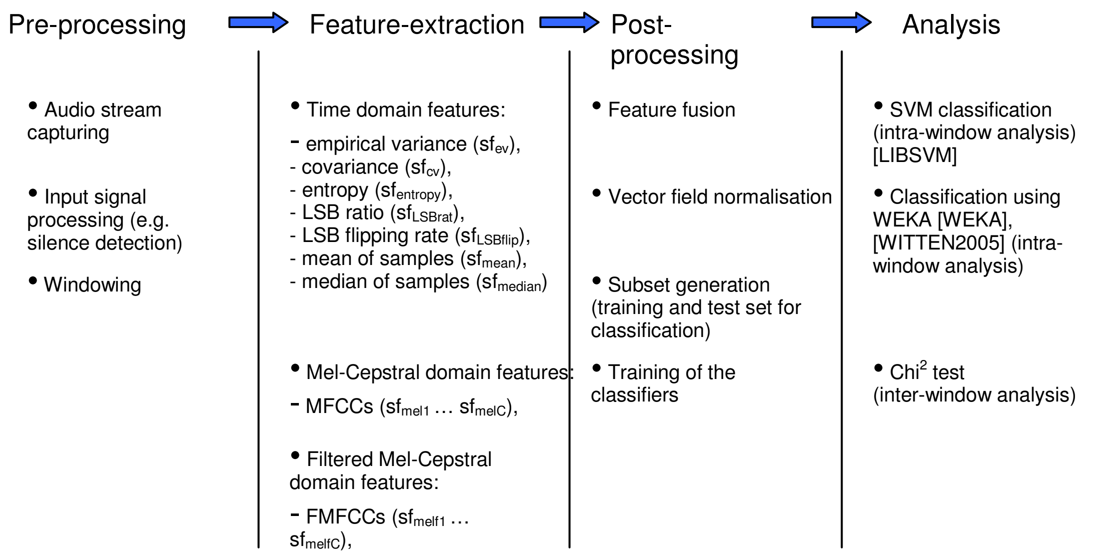


Figure 6: four modules of the AAST v.1.03

4.2.1.1 Pre-processing of the audio/speech data

The core of AAST, the feature extraction process, assumes audio files as input media. Therefore audio signals in other representations (e.g. the audio stream of a VoIP application) have to be captured into files. This is doneby the application of specific hardware or software based capturing modules on the host or in the network. In the

case of the VoIP application considered, a modified version of the IDS/IPS (Intrusion Detection/ Intrusion Prevention System) described by Dittmann et. al in [DH2004] is used as capturing device.
Additional pre-processing of the audio data (in our application scenario the speech data) handles the input and provides basic functions for data filtering (bit-plane filtering, silence detection), windowing and media specific operations like channel-interleaving/demerging.

4.2.1.2 Feature extraction from the signal

The core part of the steganalysis tool set is the feature extractor computing first order statistical features (sf_i; $sf_i \in SF$; SF = set of features in the steganalysis framework) for a window (of definable size) of the audio signal. Based on the initial idea of a universal blind steganalysis tool for multimedia steganalysis a set of statistical features used in image steganalysis was transferred to the audio domain. Originally the set of statistical features (SF) computed for windows of the signal (intra-window) consisted of:

- sf_{ev} empirical variance,
- sf_{cv} covariance,
- $sf_{entropy}$ entropy,
- sf_{LSBrat} LSB ratio,
- $sf_{LSBflip}$ LSB flipping rate,
- sf_{mean} mean of samples in time domain and
- sf_{median} median of samples in time domain.

This set of features is enhanced by the in AAST version 1.03 by the frequency domain based features:

- sf_{MFCC1} , ..., sf_{MFCC_C} with C = number of MFCCs which is depending on the sampling rate of the audio signal; for a signal with a sampling rate of 44.1 kHz C = 29) computed Mel-frequency cepstral coefficients (MFCCs) describing the rate of change in the different spectrum bands
- sf_{FMFCC1} , ..., sf_{FMFCC_C} with C = number of FMFCCs with the same dependency on the sampling rate like the MFCCs) computed filtered Mel-frequency cepstral coefficients (FMFCCs) describing the rate of change in the different spectrum bands after applying a filtering function to remove the frequency bands carrying speech relevant components in the frequency domain

4.2.1.3 Post-processing of the resulting feature vectors

In the steganalysis tool set the post-processing of the resulting feature vectors is responsible for preparing the following analysis by providing normalisation and weighting functions as well as format conversions on the feature vectors. This module was introduced to make the approach more flexible and allow for different analysis or classification approaches. Besides the operations (subset generation, normalisation, SVM training, etc) on the vector of intra-window features computed in the second module, a second feature vector can be provided by applying statistical operations like X^2 testing to the intra-window features, thereby deriving inter-window characteristics describing the evolution of the signal over time.

4.2.1.4 Analysis

The subsequent analysis as the final step in the steganalysis process is either done using a SVM (Support Vector Machine), a multi-class classifier (e.g. a Bayesian classifier) or a clustering algorithm for the classification of the signals (in the case of intra-window analysis) or by X^2 testing (for inter-window analysis).
The SVM technique is based on Vapnik's statistical learning theory [VAPNIK1995] and was used as a classification device in different steganalysis related publications (e.g. by [JOHNSON2005], [RU2005] or [MICHE2006]). For more details on SVM classification see for example Chih-Chung Chang and Chih-Jen Lin [LIBSVM] or the section concerned with SVM classification in steganography by Johnson et. al in [JOHNSON2005]. For a description of Bayesian classification see Borgelt et. al [BORGELT2001], for general classification and clustering see [HAND2001].

4.2.2 Computed Features

This section gives a detailed description of the statistical features computed by AAST v.1.03 in the feature extraction step.

4.2.2.1 Empirical variance

The empirical variance (sf_{ev}) of a set of variables (in our case the time domain features in a window of audio material) is a measure of their statistical dispersion, indicating how the values are spread around the arithmethic mean. While the mean indicates the center of the distribution of the samples, the variance indicates the variability of the values. It is computed as:

$$sf_{ev} = \frac{1}{n}\sum_{i=1}^{n}(x_i - sf_{mean})^2 \qquad (5)$$

Where n is the number of samples in the window, x_i is the *i-th* sample and sf_{mean} is the arithmetic mean of the samples in a window.

4.2.2.2 Covariance

In probability theory and statistics, covariance is the measure of how much two random variables vary together (as distinct from variance, which measures how much a single variable varies). If two variables tend to vary together (that is, when one of them is above its expected value, then the other variable tends to be above its expected value too), then the covariance between the two variables will be positive.
On the other hand, if when one of them is above its expected value, the other variable tends to be below its expected value, then the covariance between the two variables will be negative.
Here the covariance sf_{cv} is computed for $n/2$ pairs of samples from one window, the expected value used is the arithmetic mean of all values in the window (sf_{mean}). The feature sf_{cv} is computed as:

$$sf_{cv} = \frac{1}{n}\sum_{i=1}^{n}(x_i - sf_{mean})\cdot(x_{n-1} - sf_{mean}) \qquad (6)$$

4.2.2.3 Entropy

In information theory, the Shannon entropy or information entropy is a measure of the uncertainty associated with the development of a random variable. The feature sf_{ev} is computed as follows: For each window a histogram of the occurring values is generated and the value for each column in the histogram is divided by the window size resulting in a new histogram entry y_i. Equation (7) shows the computation of the feature from the h in the histogram.

$$sf_{ev} = -\sum_{i=1}^{h} y_i \log_2(y_i) \quad (7)$$

4.2.2.4 LSB ratio

The feature sf_{LSBrat} describes the ratio between the "0" and "1" LSBs within a window of the audio material. It is computed as:

$$sf_{LSBrat} = \frac{number \quad of \quad LSBs \quad set \quad to \quad 0}{number \quad of \quad LSBs \quad set \quad to \quad 1} \quad (8)$$

4.2.2.5 LSB flipping rate

The feature $sf_{LSBflip}$ counts the number of flips of the LSB within the window. It is computed as:

$$sf_{LSBflip} = |flips \; from \; LSB=0 \; to \; LSB=1| + |flips \; from \; LSB=1 \; to \; LSB=0| \quad (9)$$

4.2.2.6 Mean of samples in time domain

The feature sf_{mean} computes the arithmetic average for the samples in the window. It computed as:

$$sf_{mean} = \frac{1}{n}\sum_{i=1}^{n} x_i \quad (10)$$

Where n is the number of samples in the window and x_i is the *i-th* sample.

4.2.2.7 Median of samples in time domain

The feature sf_{median} returns the $n/2$-th value of an ordered array of size *n,* containing all samples of the window.

4.2.2.8 Mel-Frequency Cepstral Coefficients (MFCCs)

The cepstrum (an anagram of the word spectrum) was defined by B. P. Bogert, M. J. R. Healy and J. W. Tukey [BHT1963] in 1963. Basically a cepstrum is the result of taking the Fourier transform (FT) or short-time Fourier analysis [ALLEN1977] of the decibel spectrum as if it were a signal. The cepstrum can be interpreted as information about the rate of power change in different spectrum bands. It was originally invented for characterising seismic echoes resulting from earthquakes and bomb explosions. It has also been used to analyse radar signal returns. Generally a cepstrum $\tilde{S}$ can be computed from the input signal S (usually a time domain signal) as:

$$\tilde{S} = FT(\log(FT(S))) \quad (11)$$

It has to be differentiated between real cepstrum and a complex cepstrum. The difference lies in the logarithm function used and the handling of the phase information of the initial spectrum. While the real cepstrum uses a logarithm function defined only for real values the complex cepstrum uses a complex logarithm function and thereby allowing the reconstruction of the frequency domain representation of the signal (and S itself) from the complex cepstrum.
For the work described in this report only a real cepstrum is considered since the feature of invertability is not required in the analysis performed, instead the lower computation power required for the real cepstrum is appreciated.

Besides its usage in the analysis of reflected signals mentioned above, the cepstrum has found its application in another field of research. As was shown by Douglas A. Reynolds [REYNOLDS1992] and Robert H. McEachern [MCEACHERN1994] a modified cepstrum called Mel-cepstrum can be used in speaker identification and the general description of the HAS (Human Auditory System). McEachern models the human hearing based on banks of band-pass filters (the ear is known to use sensitive hairs placed along a resonant structure, providing multiple-tuned band-pass characteristics; see Hugo Fastl and Eberhard Zwicker [ZWICKER1999] or David J. M. Robinson and Malcolm O. J. Hawksford [ROBINSON2000]) by comparing the ratios of the log-magnitude of energy detected in two such adjacent band-pass structures. The Mel-cepstrum is considered by him an excellent feature vector for representing the human voice and musical signals. This insight led to the idea pursued in this work to use the Mel-cepstrum in speech steganalysis.
For all applications which are computing the cepstrum of acoustical signals, the spectrum is usually first transformed using the Mel frequency bands. The result of this transformation is called the Mel-spectrum and is used as the input of the second FT computing the Mel-cepstrum represented by the Mel frequency cepstral coefficients (MFCCs) which are used as $sf_{MFCC1},...sf_{MFCC\_C}$ in AAST. The complete transformation for the input signal S is described in equation (12).

$$MelCepstrum = FT(MelScaletransformation(FT(S))) = \begin{pmatrix} sf_{MFCC1} \\ sf_{MFCC2} \\ \cdots \\ sf_{MFCC\_C} \end{pmatrix} \quad (12)$$

Figure 7 shows the complete transformation procedure for a FFT based Mel-cepstrum computation as introduced by T. Thrasyvoulou and S. Benton [TB2003] in 2003. Other approaches found in literature use LPC based Mel-cepstrum computation. A detailed discussion about which transformation should be used in which case is given by Thrasyvoulou et. al [TB2003]. From these discussions it is obvious that the FT based approach suffices the means of steganalysis (since no inversion of the transformation is required in any of the analyses). Therefore the further descriptions are focusing on this approach.

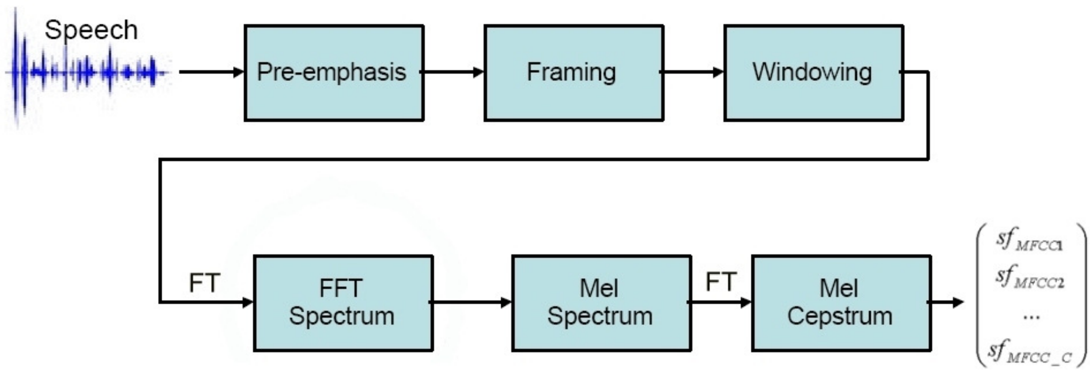


Figure 7: FFT based Mel-cepstrum computation as introduced by Thrasyvoulou et. al [TB2003]

The pre-emphasing step is taken to boost the digitalised input signal by approximately 20dB/decade using a first-order FIR (Finite Impulse Response) filter. The pre-emphasis function given by Thrasyvoulou et. al [TB2003] is:

$$H(z) = 1 + \alpha z^{-1} \quad (13)$$

The filter coefficients α used is determined empirically to be α = -0.95. The following framing and windowing are necessary preparations for the FT. Usually the frames have a size between 10 and 20ms. Thrasyvoulou et. al recommend using a standard Hamming window (see equation (14), where *N* is the number of samples in the frame and $-N/2 \leq n < N/2$) in the windowing step to reduce the edge effect when computing the FT and thereby improve the spectral estimate accuracy. Klakow [KLAKOW2006] discusses using other established window functions like Gauss or Parabola windows, but for the analysis performed in this work a standard hamming window suffices. In the implementation the Hamming window function given in equation (14) is used.

$$f_h(n) = 0.56 - 0.46\cos(\frac{2\pi n}{N-1}) \quad (14)$$

After performing the FT the magnitude squared spectrum is computed. In the next step the Mel-spectrum is computed by applying a Mel filter bank. This filter bank consists of a set of overlapping triangular band-pass filters modelled after the properties of the Mel scale. This scale was obtained in listening tests by Stevens, Volkmann and Newman [STEVENS1937] in 1937 and attempts to mimic the behaviour of the HAS in terms of the manner with which frequencies are sensed and resolved. Therefore a Mel is a unit of measurement of perceived frequency of tones. The transfer function for computing the Mel-scale value of a given frequency *f* is given in equation (15). For the computation of the Fourier transforms the AAST uses functions from the *libgsl* [LIBGSL] package.

$$f_{Mel} = 2595 * \log_{10}(1 + \frac{f_{Hz}}{700}) \tag{15}$$

For the design of the corresponding filter-bank with a number of *L* filters, the following rules are applied in Thrasyvoulou et. al:

1. In the range 0 to 1000 Hz the center frequency f_c and the bandwidths *bw* of the filters are uniformly distributed ($f_{c1} = 100$ Hz, $f_{ci+1} = f_{ci} + 100$ Hz for $i = 2...10, i \in \aleph$ and $bw_j = 100$ Hz for $j = 1...10, j \in \aleph$).
2. Above 1000 Hz a logarithmical scaling is used following the Mel-scale. To keep the overlap ratio between the filters at 50%, the bandwidths are approximated with $bw_j = 1.2 bw_{j-1}$ for $10 < j < L, j \in \aleph$. The center frequency is computed iteratively using the bandwidth by $f_{ci} = f_{ci-1} + bw_i$ for $10 < i < L, i \in \aleph$.
3. The weight *w(f)* derived from the filter bank for a frequency *f* is computed by:

$$w(f) = \begin{cases} \dfrac{\left| f_{ci} + \dfrac{bw_i}{2} - f \right|}{\dfrac{bw_i}{2}} & if \quad \left(f_{ci} - \dfrac{bw_i}{2} \right) \le f < \left(f_{ci} + \dfrac{bw_i}{2} \right) \\ & 0 \quad else \end{cases} \tag{16}$$

After applying the filter bank the resulting Mel-spectrum $\hat{S}$ is non-linearly scaled using an *log* function to generate the input for the second FT (which might be replaced for performance reasons by the modified DCT introduced by Thrasyvoulou et. al [TB2003] in equation (17).

$$c(m) = \sqrt{\frac{2}{L}} \quad \sum_{i=1}^{L} \log_{10}\left(\left|\hat{S}(i)\right|\right) \cos\left(\frac{\pi m}{L}(i - 0.5)\right) \qquad i, m \in \aleph; \quad m < (C - 1); \quad 1 \le i \le L \tag{17}$$

With *c* being the resulting cepstral coefficients, *C* being the number of cepstral coefficients desired, *L* is the number of triangular Mel weighting filters applied in the transformation of the spectrum, $\hat{S}(i)$ the Mel-spectrum energy for f_{ci} .

In a final step the result (the cepstrum $\tilde{S}$ represented by a set of cepstral coefficients $sf_{MFCC1}, ..., sf_{MFCC\_C}$) is scaled into the range [-1,1] for compatibly reasons. According to T. Pohle et al. [POHLE2005] cost commonly only the first five cepstral coefficients are used in audio signal analysis to give an description of the envelope of the frame's spectrum. In the literature various different ways to calculate the MFCCs are given (e.g. by Ernst G. Schukat-Talamazzini [Schu95] or D. Klakow [KLAKOW2006]), but in all those cases the procedure adheres the same basic concept introduced in equation

(11). They differ mainly in the way the two transformations are done (FT, FFT, STFT, DCL or LPC), the boosting and windowing operations used in the pre-emphasis as well as the number of filters in the filter-bank (some authors like
Ernst G. Schukat-Talamazzini [Schu95] consider only the range 0 - 4000Hz, others like Douglas A. Reynolds [REYNOLDS1992] advise to choose the number of the filters according to the sampling frequency of S and the Nyquist-Shannon [NYQUIST1928] sampling theorem, which would give a set of 28 filters for a signal with 44100Hz sampling rate). For the analysis performed here a modified version of the procedure introduced by Thrasyvoulou et. al. This approach is a real (therefore not invertable) algorithm which features a low complexity when compared with other approaches.

4.2.2.9 Filtered Mel-Frequency Cepstral Coefficients (FMFCCs)

In [KD2007SPIE] a Modification of the Mel-cepstral based signal analysis was introduced by Kraetzer and Dittmann introduced. It is based on the application scenario of VoIP telephony and the basic assumption that a VoIP communication consists mostly of speech communication between human speakers. This, in conjunction with the knowledge about the frequency limitations of human speech (see e.g. Fastl et. al [ZWICKER1999]), led to the idea of removing the speech relevant frequency bands (the spectrum components between 200 and 6819.59 Hz) in the spectral representation of a signal before computing the cepstrum. This procedure returning the FMFCCs (filtered Mel frequency cepstral coefficients; $sf_{FMFCC1},...,sf_{FMFCC\_C}$) is shown in figure 8.

This procedure, which enhances the computation described by equation (12) by a filter step, returns the FMFCCs (filtered Mel frequency cepstral coefficients; $sf_{FMFCC1},...,sf_{FMFCC\_C}$ in AAST) and is expressed in equation (18).

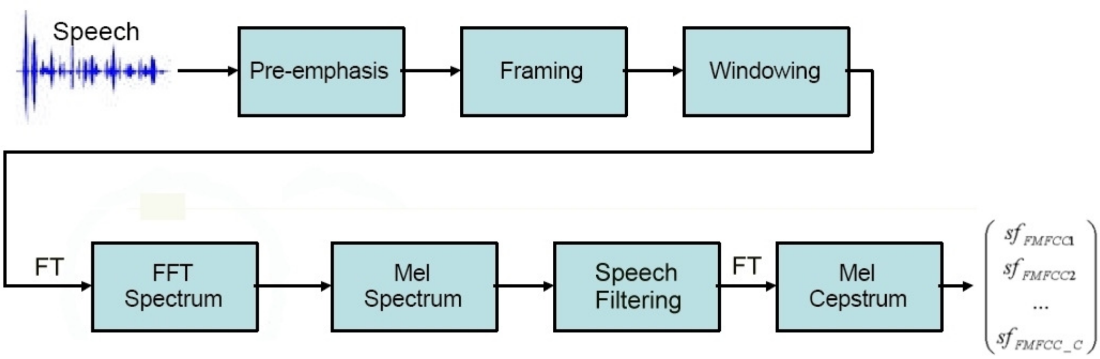


Figure 8: Computation of the FMFCCs [KD2007SPIE]

$$MelCepstrum = FT(BandFilter(MelScaletransformation(FT(S)))) = \begin{pmatrix} sf_{FMFCC1} \\ sf_{FMFCC2} \\ \ldots \\ sf_{FMFCC\_C} \end{pmatrix} \quad (18)$$

4.2.2.10 Pattern search

The AAST is also capable of performing a pattern search in the time domain signal. Here the length of the pattern has to be specified.

4.2.3 Supported Methods for inter-window analysis

In the current version v.1.03 the AAST is capable of performing a X^2 test (Chi-Square test) against equal-, normal- and exponential distributions for each of the introduced features. The result returned from this computation is the absolute distance to the selected distribution over the selected number of windows.

4.2.4 Supported Output formats

Currently three different output file formats are supported: CSV, SVM, and ARFF. The output file is specified with the `--statoutput(-ostat)=<string>` option. The output file format selected with the AAST option `--statoutformat(-statof)=<arff|csv|svm>`. Note that if the `--statconcatoutput(-conc)` option is enabled, the computed feature vectors will be appended to the output file if it already exists (otherwise the file is overwritten).
In case no output format is specified AAST returns its output only to the shell.

Important: The option `--statsselect(-statsel)=<string>` specifies in a comma separated list the features to be computed and output to the file. The keyword 'list' lists available features (use switches `-melcepstrum` and `--filtermelcepstrum` before the selection).

4.2.4.1 The CSV format

The CSV (comma separated values) format [RFC4180] is a basic exchange format for structured data, which can be exported and imported by a large number of different applications (e.g. MS Excel). The implemented CSV format output device expects no additional parameters.

4.2.4.2 The SVM format

While this is not strictly a file format, this output option is modelled after the expectations for input for the *libsvm* [LIBSVM] SVM classification software. For a detailed description of the format see [LIBSVM]. The option `--statclass(-scl)=<cover|stego>` defines the class of the material (for SVM classification using *libsvm*) this will be translated to {-1,1} as expected by *libsvm*.

4.2.4.3 The ARFF format

The ARFF format is the input format for the WEKA data mining suite [WEKA], [WITTEN2005]. This file format with a user definable header is implemented following the description found at: http://www.cs.waikato.ac.nz/~ml/weka/index.html. The option `--statarffclassset(-sacs)=<string>` specifies the set of classes which is allowed for the ARFF output. The option `--statarffclass(-sacl)=<string>` identifies the class for the current audio data (file) for the ARFF output.

4.2.5 Usage of the AAST (version 1.03)

In the 1.03 version of the AAST the tool is a command line tool compiled as static binary for Linux/Unix environments. It is launched with `./aast_static <filename> <numwin> [options]` a complete list and descriptions of all parameters/options for AAST v.1.03 is found in appendix A.

4.2.6 Practical Results

Practical results for the application of AAST v.1.03 in steganalysis and watermarking benchmarking are found in [KD2007SPIE] and [KD2007IH]. In these publications the AAST is used as a trained classifier for different watermarking and steganography algorithms. One important result shown in the papers is the fact that a statistical detection of the selected watermarking and steganography algorithms is possible with AAST. Another important result of these papers is the comparison of the results for statistical detectability (in terms of classification accuracy) for the two classes of watermarking and steganography algorithms.
Besides its application in steganalysis and watermarking benchmarking the current version of AAST was also used as a feature extraction and classification tool for media forensics in microphone and environment detection in [KODL2007ACM].

4.2.7 Summary

The practical results described briefly above (and in more details in the referenced publications) indicate that the AAST is a versatile analysis tool, which has already proven to be of practical use in watermarking benchmarking. Due to the positive results already achieved, the functionality of the AAST will be enhanced in the near future to improve its performance as a detectability benchmark for digital audio watermarking schemes.

5 Further Benchmarking Activities in WVL3

Besides the information about the work on SMBELL/SMBA, source modelling and statistical analyses using the AAST which are the fields of actual research already introduced in sections 3 and 4 of this report, the following subsections will give a brief indication of further actual and ongoing work in WVL3 and a show a connection to WVL4. Here brief information, based on WAVILA publications in the reporting period, are given on 3D watermarking benchmarking and the combination of digital watermarking and perceptual hashing in their evaluation/benchmarking.

5.1 3D Watermarking Benchmarking

In the last few years, a large number of 3D watermarking schemes have been proposed. In [BENNOUR2007] a possible benchmark to evaluate 3D watermarking algorithms is described. A list of objects and basic reproducible attacks against which 3D watermarking system could be evaluated are proposed as well as a way to compute a final score.

5.2 Digital Watermarking and Perceptual Hashing of Audio Signals with Focus on their Evaluation

In [LDK2007] Lang et. al present a theoretical framework, the design and formalisation of an evaluation profile especially for perceptual hashing algorithms. Based on this profile, on one hand, the transparency of the digital watermarking scheme or on the other hand the robustness of the perceptual hashing function can be evaluated.

A new transparency measure called the perceptual hashing difference grade (PHDG) is introduced for the transparency measurement of watermarking embed and attack functions.

6 Summary

In this report the WVL3 activities in the fields of benchmarking methods and tools for digital watermarks, steganography and steganalysis the period M25-M42 of the ECRYPT project are indicated. Since some of the research results were already reported in other publications these were referred to.

One focus of WVL3s work is the evaluation and benchmarking of schemes and algorithms based on the work of WVL1 and WVL2. For this, reliable frameworks for fair benchmarking have to identified or, if necessary, implemented. With SMBA, which was described in detail D.WVL.10, a benchmarking suite is maintained by an ECRYPT partner which is concerned with the so far neglected audio watermarking domain. In this report SMBELL, an enhancement to SMBA, is introduced for the application by end users and for large scale automated testing.

Special attention is paid in this report on source modelling in image steganalysis and the AAST analysis suite and their impact in watermarking and steganography detectability evaluations.

Additionally actual and ongoing research in 3D watermarking benchmarking, where researchers from ECRYPT are the first ones to consider benchmarking activities, and a model for the combination of digital watermarking and perceptual hashing benchmarking, thereby joining the work of WVL3 and WVL4, are presented.

WVL3 will continue its research in watermarking and steganography benchmarking and in steganalysis using the research results identified here as a foundation for upcoming evaluations.

7 Bibliography

[ALLEN1977] J. B. Allen and L. R. Rabiner, "A unified approach to short-time Fourier analysis, synthesis," in Proc. IEEE, pp. 1558–1564, Nov. 1977. Published as Proc. IEEE, volume 65, number 11.

[BENNOUR2007] Jihane Bennour and Jean-Luc Dugelay; Toward a 3D watermarking benchmark, MMSP 2007, IEEE International workshop on multimedia signal processing, October 1-3, 2007, Chania, Greece.

[BHT1963] B. P. Bogert, M. J. R. Healy, and J. W. Tukey, "The frequency analysis of time series for echoes: cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe cracking," in Proceedings of the Symposium on Time Series Analysis, M. Rosenblatt, ed., (Wiley New York, USA), Feb. 1963.

[BORGELT2001] C. Borgelt, H. Timm, and R. Kruse: Probabilistic networks and fuzzy clustering as generalizations of naive bayes classifiers. In B. Reusch and K.-H. Temme, editors, Computational Intelligence in Theory and Practice (Advances in Soft Computing), pages 121-138. Physica-Verlag, Heidelberg, Germany, Heidelberg, Germany, 2001.

[DH2004] J. Dittmann and D. Hesse: Network based intrusion detection to detect steganographic communications channels - on the example of audio data. In Proceedings of IEEE 6th Workshop on Multimedia Signal Processing, Sep. 29th - Oct. 1st 2004, Siena, Italy, ISBN 0-7803-8579-9, 2004.

[DIRENT] http://www.delorie.com/gnu/docs/dirent/dirent.5.html

[FrGo_03] J. Fridrich, M. Goljan: Digital Image Steganography Using Stochastic Modulation. Proc. of EI SPIE, Canada, 2003, 191-202.

[FrGL_05] J. Fridrich, M. Goljan, J. Lukáš: Determining Digital Image Origin Using Sensor Imperfections. Proc. EI SPIE, CA, 249-260, 2005.

[FrGS_03] J. Fridrich, M. Goljan, D. Soukal, D.: Higher-order statistical steganalysis of palette images. Proc. SPIE Electronic Imaging Santa Clara, CA, Jan 2003.

[Frid_04] J. Fridrich: Feature-Based Steganalysis for JPEG Images and its Implications for Future Design of Steganographic Schemes, 6th Information Hiding Workshop, J. Fridrich (ed), LNCS, vol. 3200, Springer-Verlag, pp. 67-81, 2004.

[FrSc_05] E. Franz, A. Schneidewind: Pre-Processing for Adding Noise Steganography. In: Proc. of the 7th International Workshop on Information Hiding, Barcelona, Spain, 2005, 189-203.

[HAND2001] D. Hand, H. Mannila, and P. Smyth: Principles of Data Mining. MIT Press, Cambridge, MA, USA, 2001.

[GETOPT] http://www.gnu.org/software/libc/manual/html_node/Getopt.html

[GoWo_02] R. C. Gonzalez, R. E. Woods: Digital Image Processing. 2nd edition, Prentice-Hall, Inc. 2002.

[HeHQ_05] J. He, J. Huang, G. Qiu: A New Approach to Estimating Hidden Message Length in Stochastic Modulation Steganography. M. Barni et al. (Eds.): IWDW 2005, Springer, LNCS 3710, 1-12.

[HoFV_05] T. Holotyak, J. Fridrich, S. Voloshynovskiy: Blind Statistical Steganalysis of Additive Steganography Using Wavelet Higher Order Statistics. Proc. of the 9th IFIP TC-6 TC-11 Conference on Communications and Multimedia Security, Salzburg, Austria, Sep. 19-21, 2005.

[JOHNSON2005] M. K. Johnson, S. Lyu and H. Farid: Steganalysis of recorded speech, In Proc. SPIE, vol. 5681, Mar. 2005, pp. 664-672, 2005.

[KD2007IH] Christian Kraetzer and Jana Dittmann: Pros and Cons of Mel-cepstrum based Audio Steganalysis using SVM Classification. To appear in the Proceedings of Information Hiding 2007, June 11-13th, 2007, St. Malo, France, 2007.

[KD2007SPIE] Christian Kraetzer and Jana Dittmann: Mel-Cepstrum Based Steganalysis for VoIP-Steganography; Proceedings of SPIE conference, at the Security, Steganography, and Watermarking of Multimedia Contents IV, IS&T/SPIE Symposium on Electronic Imaging, Jan. 28- Feb. 1st, 2007, San Jose, USA, 2007.

[KLAKOW2006] Dietrich Klakow, Lecture Notes Digital Signal Processing (Summer Term 2006), Universität des Saarlandes, Saarbrücken, Germany, http://www.lsv.uni-saarland.de/digital_ss06.htm, 2006.

[KODL2007ACM] Christian Kraetzer, Andrea Oermann, Jana Dittmann and Andreas Lang: Digital Audio Forensics: A First Practical Evaluation on Microphone and Environment Classification; Proceedings of the ACM Multimedia and Security Workshop 2007, Dallas, Texas, September 20th-21st, 2007.

[LANG2005ACM] Andreas Lang, Jana Dittmann, Ryan Spring and Claus Vielhauer: Audio watermark attacks: from single to profile attacks. In proceedings of Multimedia and Security Workshop, MM & Sec'05 (New York, NY, USA, August 1-2 2005), pp. 39 - 50, ISBN 1-59593-032-9, 2005.

[LDK2007] Andreas Lang, Jana Dittmann and Christian Kraetzer: Digital Watermarking and Perceptual Hashing of Audio Signals with Focus on their Evaluation, To appear in the Proceedings of the WAVILA Challenge 2007, St. Malo, France, June 14th, 2007.

[LIBGSL] GNU Scientific library, libgsl, 2006. Available at http://www.gnu.org/software/gsl.

[LIBSNDFILE] http://www.mega-nerd.com/libsndfile/

[LIBSVM] Chih-Chung Chang and Chih-Jen Lin: LIBSVM - a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[LIBXML2] http://xmlsoft.org/

[LyFa_05] S. Lyu, H. Farid: Steganalysis Using Higher-Order Image Statistics. IEEE Transactions on Information Forensics and Security, 2005.

[MCEACHERN1994] R. H. McEachern, “Hearing it like it is: Audio signal processing the way the ear does it,” in DSP Applications, February 1994.

[MICHE2006] Y. Miche, B. Roue, A. Lendasse, and P. Bas: A feature selection methodology for steganalysis. In Proceedings of the International Workshop on Multimedia Content Representation, Classification and Security, Istanbul (Turkey), September 11-13, 2006, Springer Berlin / Heidelberg, 2006.

[MKRM_99] M. K. Mihcak, I. Kozintsev, K. Ramchandran, P.Moulin: Low-Complexity Image Denoising Based on Statistical Modeling of Wavelet Coefficients. In: IEEE Signal Processing Letters 6 (1999), December, Nr. 12.

[NYQUIST1928] Harry Nyquist, Certain topics in telegraph transmission theory, Trans. AIEE, vol. 47, 617-644, 1928.

[Ochi_90] M. K. Ochi: Applied Probability and Stochastic Processes in Engineering and Physical Sciences. Wiley, 1990.

[Penn_05] J. Penne: Auswirkungen steganographischer Operationen auf Korrelationen zwischen Pixeln. Diplomarbeit, TU Dresden, September 2005.

[POHLE2005] Tim Pohle, Elias Pampalk and Gerhard Widmer, Evaluation of Frequently Used Audio Features for Classification of Music into Perceptual Categories, In Proceedings of the Fourth International Workshop on Content-Based Multimedia Indexing (CBMI2005), Riga, Latvia, 2005.

[REYNOLDS1992] D. A. Reynolds, A Gaussian Mixture Modeling Approach to Text-Independent Speaker Identification. Phd thesis, Department of Electrical Engineering, Georgia Institute of technology, USA, 1992.

[RFC4180] Y. Shafranovich: RFC4180, Common Format and MIME Type for Comma-Separated Values (CSV) Files,October 2005.

[ROBINSON2000] D. J. M. Robinson and M. O. J. Hawksford, “Psychoacoustic models and non-linear human hearing,” in Proceedings of the AES Convention, (109), AES, (Los Angeles), 2000.

[RU2005] X.-M. Ru, H.-J. Zhang and X. Huang: Steganalysis of audio: Attacking the steghide. In Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, China, 18-21 August, pp. 3937-3942, 2005.

[Schu95] Ernst Günter Schukat-Talamazzini, Automatische Spracherkennung, Vieweg Verlag, 1995.

[STEVENS1937] S. S. Stevens, J. Volkmann and E. B. Newman, A Scale for the Measurement of the Psychological Magnitude Pitch, The Journal of the Acoustical Society of America, volume 8, issue 3, pp.185-190, Jan. 1937.

[TB2003] T. Thrasyvoulou and S. Benton, Speech parameterization using the Mel scale Part II, 2003.

[VAPNIK1995] V.N.Vapnik, The nature of statistical learning theory, Springer Verlag, New York, 1995.

[WEKA] WEKA Data Mining Suite, http://www.cs.waikato.ac.nz/ml/weka/

[WITTEN2005] Ian H. Witten and Eibe Frank: Data Mining: Practical machine learning tools and techniques, 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

[ZWICKER1999] H. Fastl and E. Zwicker, Psychoacoustics. Facts and Models., Springer, Berlin, second ed., 1999. ISBN 3-540-65063-6.

8 Acknowledgements

Elke Franz likes to thank Thomas Gloe, Dr.-Ing. Dagmar Schönfeld, Jochen Penne and Antje Winkler for helpful discussions on the section “Analysing correlations between pixels”. Andreas Lang and Christian Kraetzer wish to thank Sebastian König for his work on SMBELL. Christian Kraetzer thanks Sebastian Heutling and Christian Zeitz for their work on AAST v.1.01-v.1.03.

9 Appendix A – AAST (v.1.03) parameters

9.1 Mandatory Parameters

The AAST has two mandatory parameters which have to be supplied:

`--filename(-f)=<string>`
Filename containing the audio data to analyse by AAST.

`--numwin(-nw)=<1...2147483647>`
Maximum number of windows to analyse by AAST. Might be reduced depending on the size of the audio file.

9.2 Optional Parameters

`--output(-o)=<string>`
Filename where to output a comma separated list of calculated data.
(obsolete; replaced in v. 1.03 by the configurable output `--statoutput(-ostat)`)

`--sizewin(-sw)=<1...2147483647>`
Window-size in samples (default: 1024)

`--offset(-off)=<1...2147483647>`
Offset in samples in the audio data to start analysis (default: 0 - off)

`--interleave(-in)=<1...2147483647>`
Overlap in samples between consecutive windows (default: 0 - off)

`--absstat(-abs)`
Calculate statistics over absolute sample values

`--usesamplebitinterval(-bit)`
Use an interval/bit-plane (e.g. only the LSBs) of bits for every sample. The two options below specify the interval.

`--lobitinterval(-lobit)=<0...29>`
If sample bit interval is enabled, specifies low bit-plane boundary. (default: 0)

`--hibitinterval(-hibit)=<1...30>`
If sample bit interval is enabled, specifies high bit-plane boundary. (default: 1)

`--perchannel(-ch)`
Calculates statistics per channel separately.

`--silencethreshold(-sil~)=<0...2147483647>`
Silence threshold value, samples below this threshold are not included into the analysis (default: 0 - off)

`--silencesamples(-siln)=<0...2147483647>`
If silence threshold is non zero and at least a number of contiguous samples specified here are found which are below the threshold value, then those samples will be ignored from being analyzed. Note that the number of windows might be reduced. (default: 8)

`--lopatternlength(-lopat)=<0...2147483647>`
Low boundary for the length of patterns. (default: 0)

`--hipatternlength(-hipat)=<1...2147483647>`
High boundary for the length of patterns. (default: 1)

`--statoutput(-ostat)=<string>`
If passed, specifies the filename for the output.

`--statoutformat(-statof)=<arff|csv|svm>`
Defines the output format and file ending of the output file. Valid formats are: svm (libsvm input format), csv, arff (Weka input format)

`--statconcatoutput(-conc)`
If enabled the computed feature vectors will be appended on existing data file (otherwise the file is overwritten).

`--statclass(-scl)=<cover|stego>`
Defines the class of the material (for SVM classification using libsvm).

`--statarffclass(-sacl)=<string>`
Defines the audio data's class for ARFF output.

`--statarffclassset(-sacs)=<string>`
Defines which set of classes is defined for ARFF output.

`--statsselect(-statsel)=<string>`
Comma separated list of features to be computed and output to the output file (or the shell). The keyword 'list' lists available features (use switches `-melcepstrum` and `--filtermelcepstrum` before the selection).

`--svmnormalize(-norm)`
If enabled normalizes the SVM output vectors. (obsolete; use the normalisation function of libsvm instead!)

`--melcepstrum(-mc)`
If enabled calculates the Mel-cepstrum as MFCCs.

`--melcepstrumoutput(-omc)=<string>`
Filename where to output a comma separated list of calculated Mel-cepstrum data for debug reasons. Automatically enables `--melcepstrum` option. If any post-processing of the vectors is desired use `--statoutput` instead!

`--filtermelcepstrum(-fmc)`
If enabled calculates the Mel-cepstrum with filtered (ignored) frequency bands between 2 and 23 (default). The range can be modified by using the two variables below.

`--lofilterboundary(-lofb)=<0...2147483647>`
Defines from which low frequency band to filter during the Mel-cepstrum calculation. (default: 2)

`--hifilterboundary(-hifb)=<0...2147483647>`
Defines up to which high frequency band to filter due Mel-cepstrum calculation. (default: 23)

`--filteredmcoutput(-ofmc)=<string>`
Filename where to output a comma separated list of calculated and filtered Mel-cepstrum data for debug reasons. Automatically enables `--filtermelcepstrum` option. If any post-processing of the vectors is desired use `--statoutput` instead!

`--chisquaretest(-chi2)=<string>`
Comma separated list of algorithms with which results to do a test. The keyword 'list' lists available algorithms.

`--chisquaretestoutput(-ochi2)=<string>`
Filename where to output a comma separated list of calculated chi square test data.

`--quiet(-q)`
Do not output calculated data to stdout.

`--help(-h)`
List all programme parameters.

`--helplong(-???)`
List all programme parameters with a description.

`--version(-v)`
Show version info and authors list.