Linköping Studies in Science and Technology
Thesis No. 1212

Distribution and Individual Watermarking of Streamed Content for Copy Protection

K-G Stenborg

LIU-TEK-LIC-2005:67
Department of Electrical Engineering
Linköping University, SE-581 83 Linköping, Sweden

Linköping 2005

Distribution and Individual Watermarking of Streamed Content for Copy Protection

Department of Electrical Engineering
Linköping University,
SE-581 83 Linköping,
Sweden

ISBN 91-85457-74-4

ISSN 0280-7971

LIU-TEK-LIC-2005:67

Printed by UniTryck, Linköping, Sweden 2005

To infinity,
...and beyond!

Buzz Lightyear

Abstract

Media such as movies and images are nowadays produced and distributed digitally. It is usually simple to make copies of digital content. Consequently illegal pirate copies can be duplicated and distributed in large quantities. One way to deter authorized content receivers from illegally redistributing the media is watermarking. If individual watermarks are contained in the digital media and a receiver is a pirate and redistributes it, the pirate at the same time distributes his identity. Thus a located pirate copy can be traced back to the pirate. The watermarked media should otherwise be indistinguishable from the original media content.

To distribute media content scalable transmission methods such as broadcast and multicast should be used. This way the distributor will only need to transmit the media once to reach all his authorized receivers. But since the same content is distributed to all receivers the requirement of individual watermarks seems to be contradictory.

In this thesis we will show how individually watermarked media content can be transmitted in a scalable way. Known methods will be reviewed and a new method will be presented. The new method is independent of what type of distribution that is used. A system with robust watermarks that are difficult to remove is described. Only small parts of the media content will be needed to identify the pirates. The method will only give a small data expansion compared to distribution of non-watermarked media.

We will also show how information theory tools can be used to expand the amount of data in the watermarks given a specific size of the media used for the watermarking. These tools can also be used to identify parts of the watermark that have been changed by deliberate deterioration of the watermarked media, made by pirates.

Keywords: Watermarking, Fingerprinting, Video distribution, Broadcast, Multicast

Acknowledgments

I started my research as a Ph.D student with a new method for compression of images that would be a generalization of Fractal coding. My interest in watermarking led me to finally work with how to distribute individually watermarked media over multicast. Traces of all research areas can be found s thesis even though the last area is the dominant one.

I would like to thank my supervisor Dr. Robert Forchheimer for always having new ideas and never being afraid to try new things. My thanks also go to **Settopbox konsortiet** which gave me many insights into the industrial side of video distribution. Important influences came during my time in the **ECSEL** research school that led to new and interesting knowledge which has been of great use later on.

I would also like to thank my colleagues in the Image Coding Group, Information Theory Group and Data Transmission Group. No one is forgotten but I would like to mention some special people below.

Peter Johansson with whom I shared office. He has always been ready to help, especially when it comes to mathematical problems. Dr. Jacob Löfvenberg and Dr. Tina Lindgren for their work on fingerprinting that first led me into the watermarking research and Jacob for giving the idea to use channel capacity to describe the watermarks. Jonas Svanberg for always having many ideas on watermarking like the credit card case in example 3. Harald Nautsch and his rock solid knowledge of image coding. Jonas Eriksson and his teacup always signaling when it is time for a coffee-break.

Marianne Svensson, Daniel Wiik and Fredrik Claesson for helping with the final touches of the thesis.

Finally I would want to thank my family and friends!

Thank you all,

K-G

Contents

Chapter 1

Introduction

In this chapter we give the background to the problem with pirate distribution and some ways to handle it. An outline of the thesis is given followed by a description of our contributions to the field.

1.1 Background

Distribution of digital media is an area that opens up many new and exciting opportunities. With the digitalization of the media it is possible to make copies without any decrease in quality. This benefits consumers as well as producers but also allows for illegitimate copying. Distribution over the Internet makes it possible to efficiently distribute such pirate copies to many people at a very low time and money cost. The making of a good distribution scheme that will distribute media without the possibility for the receivers to redistribute it is therefore desirable.

A basic security measure is to use encryption during the distribution of the media. Encryption will prevent anyone else from reading the media during the transmission. When the media has arrived at the receiver the next security problem arises, see figure 1.1. How will we know that the receiver is not a pirate illegally retransmitting the media?

One solution would be to use secure hardware that always treats the media in encrypted form. Regarding video content the TV, monitor or projector

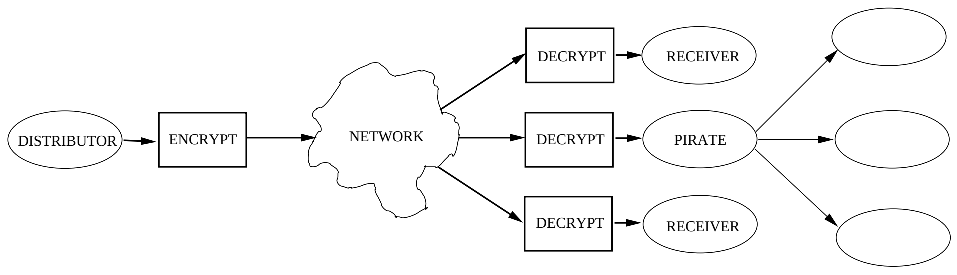

Figure 1.1: The distributor has three receivers and the media is encrypted during transmission. One of the receivers is a pirate that redistributes the media to unauthorized receivers.

used must be able to decrypt the video before displaying it, and can not have any line out feature for sending the decrypted video further. Such secure systems are still unusual and they will not stop pirates from filming the screen and dispersing illegal copies. So how do we stop the receivers from illegally distributing their copies?

So called **digital watermarking** can be used to embed hidden information (a mark) into the video stream. The mark can contain information about the copyright owner or even information about who the receiver is. If the receiver is a pirate and he redistributes the media it can be traced back to him. Thus the pirate will run a risk by illegally redistributing the media. Even when given to a trusted person the media may continue to spread while still being traceable to the original receiver.

Digital watermarking may be a part of a **Digital Rights Management** (**DRM**) system. DRM systems incorporate methods such as encryption, conditional access, media identification and copy control, to deal with all security issues for digital media distribution. In this thesis only aspects concerning watermarking will be dealt with.

Distribution of digital media is often done over networks or in broadcast environments such as for TV. If all received media are expected to have an **individual watermark** (so that redistributed pirate copies can be traced back to the pirate) the distribution must be adopted for that type of scheme. It will not be efficient to transmit an individual media copy to every receiver. Instead a more scalable transmission method must be used that still gives every receiver an individually watermarked media copy.

1.2 Outline of the Thesis

- **Chapter 2** gives an overview of watermarking and defines a general model for our work. A short description of possible attacks from pirates is given.
- **Chapter 3** continues with individual watermarks and describes what characterizes them.
- **Chapter 4** describes embedding of the watermarks. A short description of image and video coding is also given.
- **Chapter 5** describes a way of using fractal block coding to embed watermarks into images.
- **Chapter 6** describes how video containing individual watermarks can be distributed in a scalable way. Known methods are reviewed.
- **Chapter 7** describes a new method called Modified-transform watermarking that is designed for scalable distribution of individually watermarked video.
- Finally **chapter 8** gives the conclusions of the work.
- **Appendix A** contains a table over the different watermarking methods from chapter 6 and 7.

1.3 Contributions to the field

We propose embedding with a fractal block coder that uses channel capacity reasoning to make the watermark robust against attacks. The watermark channel can also be extended so that the watermark contains more data (chapter 5).

We give an extensive overview of different methods for distributing individual watermarks over multicast and broadcast networks (chapter 6).

Our method **Modified-transform watermarking** for distributing individually watermarked content is independent of the network, gives only minor data expansion and does not rely on secure hardware. The watermarks in the method are dense and robust against attacks (chapter 7).

Chapter 2

Watermarking

In this chapter we give a detailed description of watermarking and what it can be used for. The terminology that is used in the thesis is also given.

2.1 History of watermarking

Illegal coping has existed for a long time. Paintings could be very expensive and a good forgery could bring in lots of money. The idea is to fool a buyer that the painting was made by a famous painter. Therefore the signature from the painter was also falsified.

This type of illegal copying (plagiarism) is a different one from what this thesis will be about since the forgery is designed to keep the marking (the artist's signature) while pirates in this thesis want to erase the marking.

Watermarking of papers was first done in Fabriano, Italy in 1292 [1]. During manufacturing a water coated stamp was impressed on the still not dry paper pulp. The watermark in the paper was visible as a lighter pattern when the paper was viewed by transmitted light. The watermark was used by paper makers to identify their products. Watermarking can also be used on postage stamps, currency, and other government documents to discourage counterfeiting [2].

Watermarking of moving images can be traced back a hundred years to the movie company Pathé Frères. To deal with the illegal copying of their movies

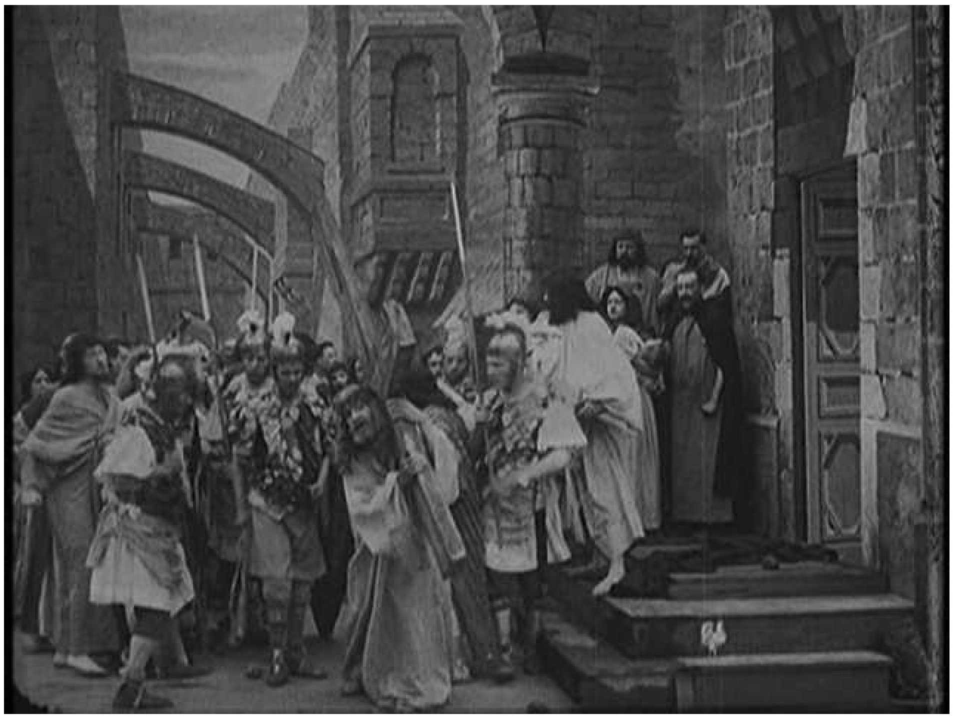

Figure 2.1: An image from the Pathé Frères movie *Vie et passion de notre Seigneur Jésus-Christ* (1907) directed by Ferdinand Zecca. Note the rooster logo painted on the stairs. *Reprinted with permission from Gaumont Pathé archives.*

during the first decade of the twentieth century Pathé Frères painted their logotype (a rooster) on the set-pieces [3]. That way the audience could see that it was a movie from Pathé Frères they were watching. The image in figure 2.1 contains the Pathé logo on the stairs to the right.

A similar type of marking is used today by TV-channels to inform the viewers about what channel they are watching. Today's corporate logo's are much more simple to embed in the image but Pathé Frères way was more elegant.

Images on the Internet may sometimes contain a copyright text in the image. This can also be seen as a type of visual watermark.

Over 50 years ago the first patent for an electronic marking devise was filed [4] [5]. The method could embed an identification code into an audio signal to prove ownership. The method was used by the Muzak Corporation to identify music that they owned. Using a notch filter the sound for some frequency component (for example 1kHz) could be blocked. By turning this filter on and off a mark could be embedded into the audio signal. They used Morse code-types of watermarks and when the mark was "transmitting" the filter was turned on, otherwise the audio signal was not altered. To extract the watermark from the audio, another filter was used to detect the embedded frequency component. That way the Morse code could be extracted from the audio signal.

It was in the 1990's that electronic watermarking became a substantial research topic [1]. Mostly watermarking of digital images, video and audio has been of interest. It is such digital watermarking that we from here on will be considering.

2.2 Watermarking for copyright management

Watermarking is an important technique for copyright protection. Below we give three applications of such use of watermarking.

- **Copyright information** Embedding visible copyright information in an image or video can prevent illegal coping. A pirate that illegally distributes it will always risk being disclosed by someone who has received such a copy.
- **Embedding receiver information** Making the receiver aware of the fact that a mark which can be traced back to him is embedded in the media object will deter from illegal coping. For more information see chapter 3.
- **Copy control** In some DRM systems watermarking is used as a control of copying and access to the media object [6] [7]. For example it may be possible to make three copies of an object before it is copy protected. This kind of copy protection will not be dealt with in this thesis.

Watermarking can also be used in other applications as can be seen in section 2.8.

2.3 Terminology

This section presents our model of the distribution. We will also give the terminology of the different parts of the model.

A **media object** can be a video, image, sound-file, program code or a text that we want to prevent from being illegally distributed. Mostly we will deal with images and video in this thesis but sometimes when we want to be more general we will call them media objects or just objects.

A media object is **close** to another objects if they are indistinguishable from each other.

The **original** is the exact media object without any embedded marks. Such an object should not be transmitted since it does not contain any watermark to prevent it from being copied.

A **watermark** or **mark** is information that can be used to prevent illegal coping. In the literature the watermark is sometimes called label, tag or signature.

A watermark can be **embedded** into a media object. When the embedding is done the media object will be distorted. After the embedding we say that the media object is **watermarked**.

The watermark can be **extracted** from the watermarked media object. Extraction methods will be described in section 2.6.

A **copy** is a media object that contains an embedded mark. The copy should be close to the original and only introduce little distortion. The copy should work just as good as the original for its purpose.

In chapter 3 we will start talking about many **copies** from one original and that every copy contains different individual watermarks. So the copies from one original all contains different information and no-one of them are exactly the same as the original.

The **distributor** is the company or person that sells the media objects. The distributor can be the copyright holder of the media object or just someone with the right to distribute the object.

A **receiver** or **legitimate receiver** is a person who legally buys a copy from a distributor. In the literature the receiver is sometimes called customer, subscriber or user.

A media object is **illegally distributed** if it is distributed by someone else than the distributor. Such a media object is called an **illegal copy**.

A **pirate** is a receiver that illegally distributes a media object and an **unauthorized receiver** is the person who gets the illegal copy.

An **attack** can be performed on a copy by a pirate with the intent to remove the embedded mark. More about attacks can be read in the next section and in section 3.3.

An **attacked copy** is a copy that has been attacked by a pirate in an attempt to remove the watermark.

2.4 Pirate attacks

There is a lot of different attacks that a pirate can use to weaken or remove the watermark from a media object. If the media object is an image it can be subject to some of these attacks:

- Cropping
- Scaling
- Rotation
- Stretching
- Translation
- Skewing
- Filtering
- Noise addition
- Quantization (least significant bits manipulation)

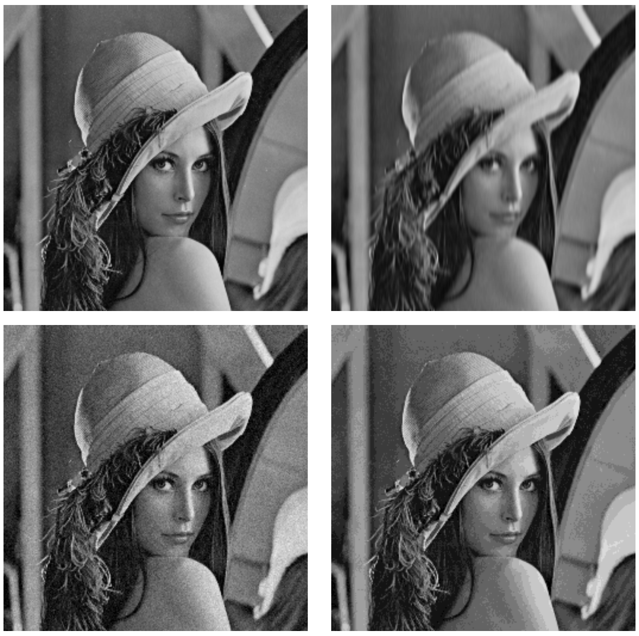

Figure 2.2: (a) Original Lenna image (b) Image attacked with low-pass filter. (c) Image attacked with Gaussian noise. (d) Image attacked with quantization.

- Lossy compression
- A/D-D/A conversion
- Over-marking (Embedding new watermarks)

In figure 2.2 the visual results on an image after some of these attacks are shown.

The pirate does not want to destroy the image he attacks so the attacks should not introduce to much distortion on the images. For video streams the embedding can also be done in the audio and in the temporal space. Therefore efficient attacks on video should not only change the images but also the audio and the time domain.

Another type of attack that works for individual watermarks will be described in section 3.3. In the next two sections specific classes of watermarking will be described and attacks for these cases will be discussed.

2.5 Visible and invisible watermarks

A watermark that is embedded into a media object can either be perceptually visible or invisible. The expressions visible and invisible can for some media objects (e.g. watermarking of audio) seem incorrect. Better expressions for these applications would be perceptible and imperceptible although we will not be using them here.

2.5.1 Visible watermarks

A visible watermark can be seen by the human eye and often it contains a name or logo of a company or copyright information. This type of mark is intended to be easy to read by a receiver and deter pirate distribution since everyone can see the origin of the video. On the other hand, a visible watermark makes it easier for the pirate to see if an attack was successful.

This type of watermarking is often used by stock photo companies that want to promote their images without them being stolen [8]. The ever present logos on TV-channels and the example in figure 2.1 are visible watermarks.

For some applications the visible mark can be disturbing and may discourage people from using the service. For those cases invisible watermarks can be used instead.

2.5.2 Invisible watermarks

For invisible watermarks a copy should be indistinguishable from the original, i.e. the embedding of the watermark should not introduce perceptual distortion to the media object. Since the invisible watermark can not be detected by the human eye we need some type of extraction algorithm to be able to read the watermark. More about how invisible embedding can be done is described in chapter 4.

Invisible watermarks are the type of watermarks that will be investigated in this thesis. Sometimes both visible and invisible watermarks are used. This kind of watermarking is called Dual watermarking [9].

2.6 Private and public watermarking

To extract an invisible mark you will need to know the algorithm. This knowledge can either be private or public. Sometimes the extraction are split up into the more detailed categories: private, semi-private, public and public key watermarking [8].

2.6.1 Private watermarking

For **private watermarking** (also known as non-blind watermarking) the original media object and the watermark is needed for the extraction. That means that only the distributor will be able to extract the embedded watermark. To let someone else perform the extraction the distributor must trust them with the original media object with the risk of the original being illegally redistributed.

Semi-private watermarking is similar to private watermarking but for this case the original media object is not needed, only the watermark. This can be used by systems that automatically check the watermark. In some cases a special unlock key is needed to be able to extract the watermark. The distributor then can give trusted persons the unlock key instead of the original as in the private watermarking case. That way the original can always be kept safe by the distributor.

There also exist programs or web-crawlers that can search the Internet for images and investigate if they contain any watermarks embedded by an algorithm known by the program [10]. Such programs then can tell the distributor where they have found media objects that contain watermarks belonging to the distributor.

Neither private nor semi-private watermarking will let the receiver extract the watermark. This means that if someone receives a media object he can not check the watermark and therefore can not get information about the owner of the object.

2.6.2 Public watermarking

In **public watermarking** (also known as blind or oblivious watermarking) neither the original nor the watermark is needed to be able to extract the watermark. Either an extraction key is needed or the extraction algorithm can be secret in this case. That way the receivers can not extract the watermark by them selves. The extraction key or algorithm will only be known by the distributor and other trusted parties. Using an extraction key is best since relying on a secret algorithm is always risky in case of the revealing of the algorithm.

With **public key watermarking** (also known as asymmetric watermarking) anyone with a copy can extract the watermark from it. Since the system is asymmetric the knowledge of the extraction algorithm can not be used to directly remove the watermark as with symmetric watermarking [11].

A positive effect in this category is that if a person thinks he has come by an illegal copy he can check the watermark and report to the distributor if he wants to. A negative effect is that since the receiver can extract the watermark he can make an Oracle attack [8] by attacking the copy and then checking if the watermark remains. If it remains he can change the attack and test again. Thus the watermark in such systems could be simple to remove for a pirate.

Sometimes public watermarking or public key watermarking are very useful since the presence of the original and the watermark is not needed. This will make the extraction easier and possible to perform by more people.

Since public watermarking embedding always can be made private it is better to try and make the algorithm public [8]. This way one can later on decide if the final algorithm should be public or private.

2.7 Robust and fragile watermarks

Another categorization of watermarks are if they are robust or fragile. In this section we will explain the difference between them.

2.7.1 Robust watermarks

The watermarks that are used for copy protection are robust. It does not necessarily mean that all of them are robust against all pirate attacks but that they aim to be robust.

Ideally the watermark should survive any attacks as long as the media object is possible to use, i.e. as long as an attacked movie have a fair image quality.

Definition 1 *A **robust watermark** should be able to survive attacks up to the point when the attacks have done the copy unusable for its purpose.*

The robustness of the watermark will effect the quality of the copy. The more robust a watermark in a copy is the further from the original media object the copy is. So we need to find a good robust watermark embedding which only causes limited visual degradation. The robustness will also depend on the amount of data in the watermark. More about this in section 3.2

Although it is difficult to achieve robust watermarks in practise we will consider these to be available in later parts of this thesis.

2.7.2 Fragile watermarks

Watermarks that are fragile should be distorted immediately if the media object is manipulated or attacked. One possible use of fragile watermarks are in cameras. With digital cameras no original negative exists that can be used to identify forgeries. Instead a fragile watermark embedded by the camera can tell if the image is the original or if it has been changed [7]. Of course the original image can not be reconstructed from the watermark. But the embedding algorithm can be done in such a way that the watermark can tell what part of the image that has been changed. The watermark can also contain information about which camera that was used.

Security cameras and medical images are possible areas where fragile watermarks can be of use. How a combination of fragile and robust watermarks can be used for copyright protection is discussed in [12].

For Copy control applications (see section 2.2) fragile watermarking can be used. A fragile watermark in an image for example, must be intact otherwise it is impossible to make a copy of the image. If the program deliberately modifies the image no more copies can be made from it. Copied images should also be modified so that they are impossible to copy.

2.8 Other watermarking applications

As mentioned in the last section watermarks can be used for other purposes than proprietary rights. In this section we will give some examples of embedding of marks that we will not deal with any further in this thesis.

- **Steganography** means to embedded secret information in different types of media objects. For example it can be text messages embedded into images that are located on the Internet. Only people that know how to extract the embedded information can read the message. Steganography is the oldest of all data hiding applications and have been used for thousands of years [13].
- **Data monitoring** is a way to save data of what is transmitted from for example a TV-channel. In 1997 it was discovered that some TV-stations in Japan were over-booking their air time for commercials. They got paid by advertisers for hours of commercials that were never aired [7]. Watermarking can be used as part of an automatic monitoring system that store information about what have been aired by the TV-stations.

  Data monitoring can also be used for statistical data collection and analysis [14].
- **Time stamping** with the use of watermarking will make it possible for media objects to contain information about when they were created or last used [15]. The time stamp can be used for copyright protection but also in other instances when it is important that the media has a time mark. For example security cameras will need to have an exact time for the images being recorded.

It is possible to use more than one of these watermarking applications in the same media object by embedding more than one mark [16].

Chapter 3

Individual watermarking

In this chapter we will describe individual watermarking. Individual watermarking is sometimes called fingerprinting or transactional watermarking.

3.1 Introduction to individual watermarks

Individual watermarking means that each receiver will get a copy with an individually embedded mark. If a receiver is a pirate the illegal copy that he redistributes contains an individual watermark or “fingerprint” that will pinpoint him as the pirate. Therefore a smart pirate will try to use an attack to remove the individual watermark. If more than one pirate collaborate they can perform special types of attacks, see section 3.3.

Individual watermarking or **fingerprinting** was first described in 1983 [17]. In fingerprinting the individual marks are described as (mostly binary) **codes**. This research area tries to construct codes that will be strong against attack where parts of the codes have been changed or destroyed. Collaborating attacks are of great concern since two or more pirates should not be able to destroy their fingerprints by mixing their individual copies [18] [19]. Another important aspect is the length of the codes. If they are long it might be difficult to embed them into a media object.

In literature on digital images the expression fingerprint is sometimes used for the data that is retrieved or extracted from images for indexing purposes [20]. Therefore we will use the expression individual watermarking instead of fingerprinting in this thesis.

The legal aspects of individual watermarks is another research aspect. How can we know that it is the receiver that is the pirate and not the distributor that has made a mistake, or worse, the distributor tries to frame the receiver? How to prove the dispute to a third party [21] will not be dealt with in this thesis. Implementation of a solution to that issue is very important if individual watermarking is going to be accepted by the consumers.

3.2 Size of the watermark

The amount of information in watermarks is an important parameter, both for ordinary watermarks and for individual watermarks. When a copy is transmitted and attacked the watermark that is embedded in the copy is affected. Between the embedding and extraction the watermark can be viewed as being transmitted on a channel, the **watermarking channel**. The watermarking channel properties depends on the size of the object and the type of embedding that is used.

Channel capacity [22] can be used to calculate the amount of information that a watermark can contain for a specific watermarking channel. This will be described further in section 5.4.

Categorizing types of marks has been done before [17]. We will split up watermarks into sparse and dense marks depending on the amount of data that they can contain. If the watermarks are sparse or dense depends on the relationship between the watermarking channel and the main channel.

3.2.1 Sparse watermarks

Marks that are sparse are used in research areas like fingerprinting mentioned in last section. Usually we have binary bits that are placed in a number of **positions** of the digital media object. The combination of these bits will be the individual **codeword** for each receiver.

Not all sparse watermarks are binary (only two possible values in each position) but the amount of possible values for each position is still small.

Definition 2 ***Sparse watermarks*** *have a small number of positions compared to the size of the media object and a small number of possible values for every position.*

Usually only a minor part of the media object will have marked positions but that is not always true. The following example describes a simplified version of the embedding algorithm from [23] that will be further investigated in subsection 6.5.3.

Example 1 *We have a video stream of 25 frames/second and every frame can either be a 0-frame or a 1-frame. The 0-frame and 1-frame is similar to the original frame but contains small invisible differences. If we, for a robust fingerprinting codeword, need 15000 bits it will require 10 minutes of video to embed it.*

As can be seen in the example every part of the video is part of a marked position. However the number of positions is still small compared to the total amount of bits in the digital data.

For sparse watermarks we need a large media object (i.e. video, digital radio) for the embedding or we will only be able to embed very little information.

3.2.2 Dense watermarks

For watermarks that are dense, a lot of information can be embedded in a small media object. For example the whole watermark can be contained in an image. Compare this to sparse watermarking where a complete video stream could be needed for one mark. Dense watermarks are sometimes desirable since every frame in a video can contain the watermark and that way the extraction will be much faster. If just one image from a video is illegally distributed the watermark can still be extracted.

Definition 3 ***Dense watermarks*** *have a similar number of positions compared to the size of the media object and a fair number of possible values for every position.*

Let us give an example of a very simple embedding algorithm for dense watermarks:

Example 2 *An image contains* 256×256 *pixels. Every pixel is seen as a position for the embedding. In every pixel we can embed* 8 *different values* ($+4$ *to* $+1$ *and* -1 *to* -4) *by adding the watermark value to the pixel value. The watermark for one image will then contain* $3 \cdot 256^2 = 196,608$ *bits.*

The embedding algorithm described in example 2 is not robust against attacks but it gives a feeling of what a dense watermark is.

If we have a robust embedding algorithm and a given maximum of the embedding distortion, the robustness will decrease if the size of the watermark increases [8].

For embedding of a small amount of information in a large media object dense watermarks can still be used. The extra information in the dense watermark can then be made more robust by using error correcting codes.

3.3 Collaborating attacks

For individually marked media objects there is another type of attack. If two or more receivers collaborate they can blend their copies to compromise the watermarks. For video and images these attacks by collaborating pirates can be done by:

- Using the mean of the different copies
- Combining different parts of the image/video from different copies
- Use the copies to find the positions of the watermark and attack or remove these parts.

Some of these attacks will be tested in section 4.3 and 7.5.2.

If successful the pirates would be able to remove all the marks so that none of the pirates' individual watermarks remain. It would be even worse if some innocent receivers watermark would appear after the extraction. This is known as a **false positive**.

So, a distributor must take this into account when creating the embedding and codes of the watermark. It seems like a reasonable goal that at least one of the pirates' watermark should be identified and that there are no false positives as long as the attacks have not destroyed the media object.

3.4 Advantages of individual watermarks

In this section we will discuss a few different schemes of individual marking to give an idea of what can be used in different cases.

3.4.1 Private individual watermarks

Private (invisible) individual watermarking is the most common scheme. Only authorized persons (for example the distributor) can extract the watermark. To be able to trace a pirate an illegal copy must be found by, or given to, an authorized person.

3.4.2 Public individual watermarks

Consider a case with public key watermarks. Then an illegal receiver can read the mark and maybe even trace the mark to the pirate.

If the mark would contain information that the receiver would want to keep secret he would not likely want to illegally distribute that video.

Example 3 *Say that the receiver has bought a movie from a distributor. The receiver has used his credit card to pay for it and the distributor has embedded the credit card number (= the watermark) into the video. Knowing that the watermarking is public and that the extraction can be done with a public key the receiver will most likely not want to illegally distribute the video. If he does, his credit card number will be disclosed to all illegal receivers.*

Example 3 is not that realistic since very few receivers would use a distributor that embeds their credit card number. But the example could be changed so that something else is embedded like a one-time code that will allow the receiver to see another movies for free.

A disadvantage of individual marks with public key watermarking is pirates collaborating in attacking the watermarks can see if they have been successful. It might even be so that they can see if they have been able to create a valid watermark so that an innocent receiver will be blamed for the piracy.

3.4.3 Combining watermarking and individual watermarking

We can have a mark embedded by public key watermarking so that any person can see who the distributor is. This can be combined with a private individual mark so that only the distributor can see who the pirate is. Then, if a media object belonging to a distributor is found and the distributor is informed, he can extract the individual watermark to identify the pirate.

Chapter 4

Embedding of watermarks

In this chapter we will go deeper into how embedding of non-visual watermarks is done in images and video. The chapter will mostly be concentrating on the spread spectrum method for embedding since it is a simple embedding which is robust against attacks.

4.1 Embedding of invisible watermarks

There are a lot of different methods for embedding invisible watermarks into media objects such as images and video. Instead of describing all of them we will in this chapter give some background and a quick overview of one of the best known methods, the spread spectrum embedding.

Embedding methods for images have many similarities to image coding. Image coding methods, can with small changes, be used to embed watermarks in the images. In chapter 5 we will use fractal coding for watermarking of images while the method presented in chapter 7 is based on JPEG or MPEG. In the next section we will therefore give a quick overview of image and video coding.

4.2 Coding of images and video

Embedding in images usually has a lot in common with encoding and decoding of images. **Encoding** of images is a way to compress the amount of

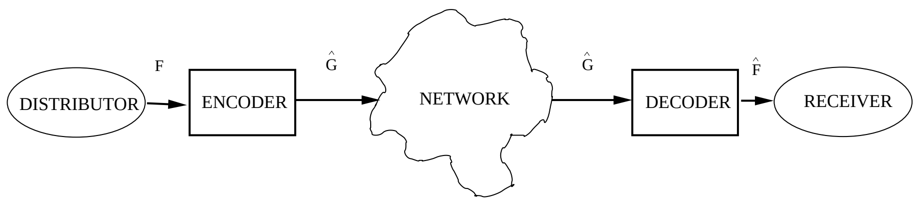

Figure 4.1: A media F is encoded with at the distributor. The lossy compressed media $\hat{G}$ is transmitted over a network. At the receiver the media is decoded to $\hat{F}$.

data needed to describe an image. The compression can be either **lossless** or **lossy**. Lossless means that the image does not contain any distortion after the decoding. Since the embedding of watermarks will distort the image lossless compression can not be used for embedding. Lossy compression will distort the image but a good encoder will make the distortion perceptually invisible to the human eye. Similar low distortion is wanted after embedding of invisible watermarks. After the encoding the data is transmitted or stored. To be able to see the image we need to **decode** the compressed data. Figure 4.1 illustrates the encoding, transmission and decoding of a media object.

Coding and embedding have many similarities but they can also be seen as opposites.

Encoding - Removing insignificant visual information.

Embedding - Adding insignificant visual information.

If we had perfect encoding and perfect embedding the watermark would be removed if the image was coded since the encoder would remove all insignificant information, e.g. the watermark. This would be the perfect attack!

4.2.1 Image coding

There exist many different methods for coding of images. We will give a short overview of one of them, **transform coding**. The transform coding can be split into three parts, **transformation**, **quantization** and **source coding**. In the transformation part the image F is first split into smaller blocks F^j (for example of size 8×8 pixels). These blocks are transformed

$$G^j = TF^jT^t \tag{4.1}$$

as expressed by the transform matrix T. The most common transform in image coding is the DCT (discrete cosine transform) [24]. After the transformation the coefficients are quantized $\hat{G}^j$ and ordered in a suitable way. The quantized coefficients are then source coded (losslessly compressed) for all blocks in the image.

During the decoding the inverse transform is used

$$\hat{F}^j = T^t\hat{G}^jT \tag{4.2}$$

with $T^{-1} = T^t$ since the matrix is usually orthonormal. The transform coding is lossy since information is lost during the quantization, $\hat{F}^j \neq F^j$.

The well known coding standard JPEG uses DCT transform coding. Exactly how quantization and source coding is done in JPEG will not be described here.

4.2.2 Video coding

Coding of video has some similarities to still image coding but there are more images and also audio. We will not deal with the compression of audio in this thesis.

The images of the video are called frames. Since we usually have between 24 and 30 frames/second in video, neighboring frames will be similar. During encoding this fact can be utilized. In the encoder the frames will be divided into I-frames, P-frames or B-frames depending on their order in the video sequence. If we call the frames I, P and B one common sequence is $IBBPBBPBBPBB$ that is then repeated from the start over and over, see figure 4.2.

The **I-frames** (I for intra) are coded in a similar way as in JPEG. The **P-frames** (predictive) and **B-frames** (bidirectionally predictive) are coded using motion-compensated prediction from other neighboring frames. Such

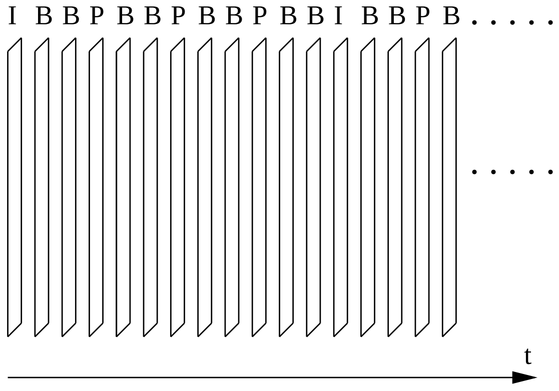

Figure 4.2: A sequence of frames from video. The I-frames, B-frames and P-frames are repeated in one of many possible patterns.

prediction will need less data and therefore the P- and B-frames will achieve much higher compression than the I-frames.

Most video compression standards like MPEG use this type of compression for the frames. Some video coding methods only use I-frames and will therefore not be able to compress as much as MPEG encoders. These I-frame coders are mostly used in the production of video sequences but not during transmission to the receivers.

4.3 Embedding in images

One of the most known embedding methods uses spread spectrum. Spread spectrum watermarking was first introduced in [25]. When embedding a spread spectrum watermark into an image the image is first transformed using DCT

$$G = TFT^t. \tag{4.3}$$

After that the watermark W (pseudo random noise) is inserted

$$G_W = G + W \tag{4.4}$$

in the transform domain. A weighting can be used to make the insertion stronger or weaker in specific parts of the transform domain. Thereafter the embedded image is given by

$$F_W = T^t G_W T \tag{4.5}$$

and F_W will be close to F.

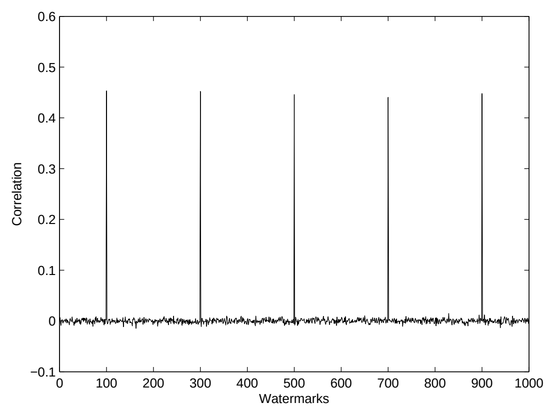

Figure 4.3: Correlation between the attacked watermark and the existing watermarks. We can see that receiver number 100, 300, 500, 700 and 900 are the pirates that have collaborated during the attack.

To extract the watermark we need the difference between the watermarked image and the original image. The extracted watermark is then correlated with all different watermarks to identify which watermark that has been extracted. This is therefore a private watermarking method.

We will now give a short example of how spread spectrum watermarking holds up to a collaboration attack.

Example 4 *We have 1000 different watermarks. Five pirates have collaborated and taken the mean of the pixel values from their individually watermarked copies. We extract the attacked watermark and correlate it with the other watermarks. The result is shown in figure 4.3.*

As can be seen in figure 4.3 all five pirates can be identified. This method is therefore robust against this particular attack. The spread spectrum watermarking is also dense.

4.4 Embedding in video

When embedding watermarks in video the embedding can be done in the I-frames, the P- and B-frames and in the audio. The embedding in the images should if possible be dense so that only one frame is needed to extract the watermark. The audio should contain the whole watermark so that if a pirate only redistributes the audio it will be possible to extract the watermark. In this thesis we will only deal with embedding in the I-frames of the video sequence and therefore watermarking of video will be very similar to watermarking of still images.

If the video stream is encoded we should not need to decode the stream to embed the watermark. It is much more efficient if the watermark can be embedded directly into the stream so that only the parts which are needed are decoded during embedding [26]. Such an embedding method can be seen as a type of transcoding which does not change the bit rate much but instead embeds the mark.

Chapter 5

Embedding using Fractal Coding

In this chapter we will describe a method to embed a watermark into a still image using ideas from fractal block coding. The method will be enhanced so that we will be able to see if an attack has been done on the image. We will also show how channel capacity calculations can be used to increase the amount of data in the watermark.

5.1 Introduction

Many watermarking schemes have used coding algorithms as base for their embedding. We will here give a description of how fractal block coding can be used for embedding and extraction of binary watermarks in images.

5.2 Fractal block coding

Jacquin [27] proposed a block based approach to the concept of fractal coding. In his **fractal block coding** method the image is divided into non-overlapping range blocks R_i of size $N \times N$. The image is also divided by (possibly overlapping) domain blocks D_j of size $2N \times 2N$. For each range block

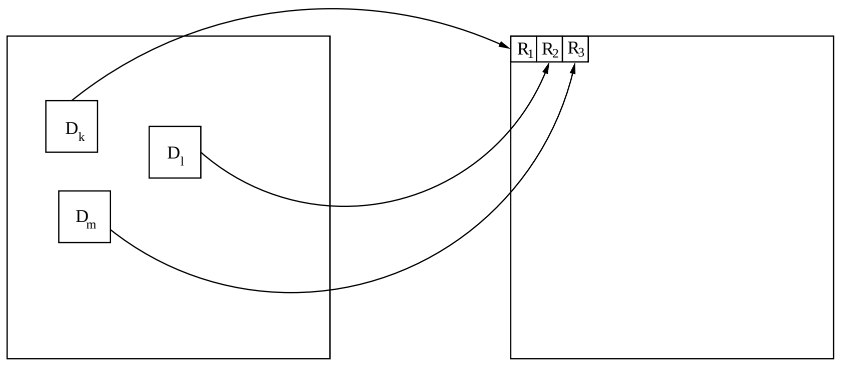

Figure 5.1: The content in range block R_1, R_2 and R_3 comes from domain block D_k, D_l and D_m.

the most similar domain block is found. The content of the domain block is then mapped to the range block, see figure 5.1. The contractive mapping can be written as $sI_l(S(D_j))+c$ where S is a spatial contraction, I_l is one of eight isometric transforms, s is the contrast scaling and c is the luminance shift. The encoding is done by searching for the mapping for each range block that minimizes the mean square error $< (R_i - (sI_l(S(D_j)) + c))^2 >, \forall j, l$. The quantized parameters s_i, $I_{l,i}$, c_i and the location of the domain block is the encoded signal for the range block R_i.

The decoding is done by performing the contractive mappings on a random image of the same size as the original image. The resulting image is used to perform the mapping a second time. By continuing to iterate in this way the resulting image will converge to what is called the **attractor**. The attractor is the decoded image. In the attractor all the range blocks are exact mappings from a domain block. If the attractor would be encoded with the same fractal block coder the resulting decoded image would be the same as the original attractor. It is this fact that can be used to embed and extract watermarks.

5.3 Watermarking with Fractal block coding

The first article on embedding using fractal block coding was published in 1996 [28]. In this method two non-overlapping regions of the image $F_{original}$ are selected. One of these regions will be used for range blocks and the other

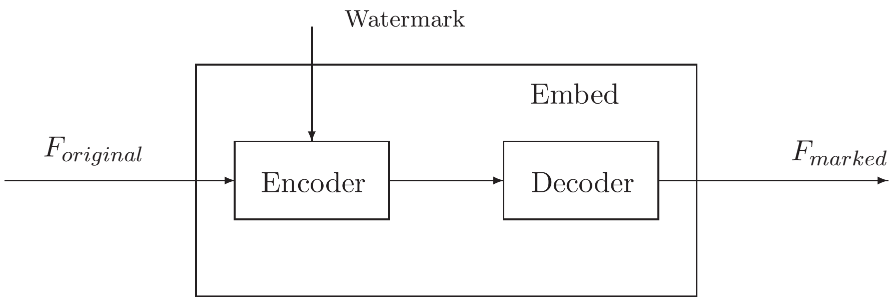

Figure 5.2: Embedding the watermark.

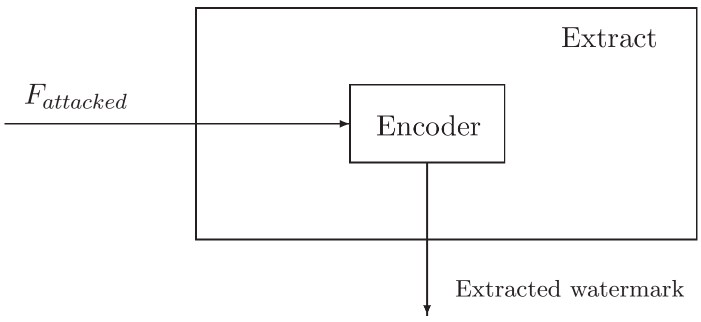

Figure 5.3: Extracting the watermark.

for domain blocks. The domain region is divided into two disjunct parts D_0 and D_1. For every range block in the range region one binary symbol can be embedded by selecting if the domain block should be searched for in D_0 or D_1. Which binary symbol that is embedded in every range block is given by the binary watermark. Since the range region and the domain region is non-overlapping the attractor is reached after only one mapping in the decoder. The attractor is now the embedded image, F_{marked}. The embedding is described in figure 5.2.

To extract the watermark from a possibly attacked embedded image F_{attack} we search both D_0 and D_1 to find the best match for every range block in the range region. If the best match occurs in D_0 the extracted symbol for that range block is 0 and otherwise 1. The extracted symbols for all range blocks are collected into the extracted watermark. The extraction is described in figure 5.3.

This method has many disadvantages. To extract the watermark we must know the location of the range region as well as D_0 and D_1. The amount of data in the watermark is restricted by the number of range blocks in the range region. Since the range region and the domain region are non-overlapping and the domain blocks are of size $2N \times 2N$ compared to the range blocks $N \times N$ the range region must be kept small not to cause high distortion. Therefore this method will give a limited amount of data in the watermark.

Another paper [29] from the same year lets both the range region and the domain region consist of the whole image. To embed one bit in one range block the domain block was either searched for close to the range block (D_0) or the rest of the image (D_1). The attractor that is given from the iterative decoding of this fractal block coded signal is the embedded image, F_{marked}.

This method can embed more data into the image and the range and domain regions are well defined. One disadvantage is that the best domain block usually can be found close to the range block. Embedding of a 1 can therefore cause high distortion. This problem is solved in [30] where the two domain block regions are given by the location of the upper-left pixel in the domain block. If the pixel is in an even numbered column it belongs to D_0 and otherwise D_1.

Other papers on watermark embedding have also taken ideas from fractal block coding. In [31] the range blocks are classified into self-similar and non-self-similar range blocks. In the embedding the LSB (Least Significant Bit) of the pixels in a range block is set to 0 or 1 depending on range block class and bit value to encode. In this method the image is not encoded using fractal block coding. Since only the LSB is changed an easy attack is to choose the LSB randomly for all pixels. Therefore this and all other LSB methods can not be considered to be robust against attacks.

In [32] the domain blocks of the same size as the range blocks are used. The watermark is embedded by changing some blocks to give new exact mappings with no distortion given by the mapping. To conceal these exact self-similar mappings, a scheme is tested where only some of the low frequency coefficients in the DCT of the range and domain blocks are altered. The DCT coefficient version is more robust against noise.

Another version of the fractal block coding embedding [33] uses the isometric transforms to split the domain region into D_0 and D_1. Four of the isometric transforms are used for D_0 and the other four are used for D_1.

All of these methods use the same concept namely that the domain region is divided into two subregions and that the watermark controls the choice of domain region for each range block. The watermark is binary and during extraction both domain regions are searched to find the best match and thereby the embedded symbol. Observe that the parameters in the fractal block coder do not need to be quantized since no compression is needed between the encoder and decoder in figure 5.2.

5.4 Channel capacity description

In this section we will investigate how channel capacity can be used to describe the amount of data that the watermark can contain. We will only give a quick overview of channel capacity here but more information can be found in [22].

Definition 4 *Let X be the input and Y the output of a communication channel. The probability of receiving a given output symbol from a specific input symbol is given by $p(y|x)$. Then the* ***channel capacity*** *for a discrete memoryless channel is defined by*

$$C = \max_{p(x)} I(X;Y) \tag{5.1}$$

with $I(X;Y)$ being the ***mutual information*** *between the input values X and the output values Y. The mutual information is given by*

$$I(X;Y) = H(Y) - H(Y|X) \tag{5.2}$$

there we have the ***entropy***

$$H(Y) = -\sum_{y} p(y) \log_2 p(y) \tag{5.3}$$

and ***conditional entropy***

$$H(Y|X) = -\sum_{x} p(x) \sum_{y} p(y|x) \log_2 p(y|x). \tag{5.4}$$

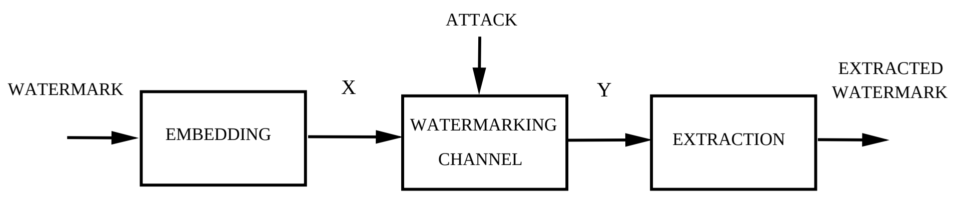

Figure 5.4: The watermarking channel. A watermark is embedded into a media. While the watermark is embedded it will be affected by attacks done to the media. The extracted watermark might therefore contain errors.

The channel coding theorem [22] states that all rates R below capacity C are achievable. So if we have a given channel capacity C we know that there always exist codes that can give any rate $R \leq C$ with arbitrarily low probability of error. For $R \geq C$ the probability of error will not be arbitrarily low. So the channel capacity is the limit of what rate a channel can achieve. To reach or come close to the channel capacity error correcting codes will be needed.

In the following subsections we will investigate the channel capacity for three different types of channels. This watermarking channel describes what can happen to the watermark between the embedding and extraction. Errors in the watermarking channel are caused by attacks to the copy by pirates. Figure 5.4 illustrates the watermarking channel.

First we will give the channel capacity for the fractal coding methods described in the last section. After that we will investigate two extensions of the method and its watermarking channels. The last of them can be seen as a generalization of the previous two channels.

5.4.1 Binary watermarking channel

In section 5.3 we have a binary watermarking channel with only two different input symbols $X_i \in \{0, 1\}$ for every position. The possible output symbols are also two, $Y_i \in \{0, 1\}$. If the image is attacked errors will occur in the watermarking channel and the output symbols for some positions will not be the same as the input symbols. This type of channel is called a **Binary**

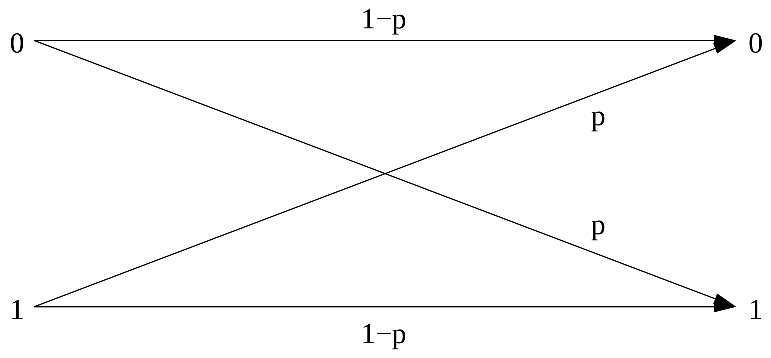

Figure 5.5: Binary Symmetric Channel (BSC)

Symmetric Channel (BSC). If the probability that a symbol will change in the watermarking channel is p the BSC will be described by figure 5.5.

The probability p will depend on the probability r that a range block will be mapped from an incorrect domain block. The probability of incorrect mapping, $0 \leq r \leq 1$, is then the probability that an attack has been successful. If the attack has succeeded at a certain position we assume that the incorrect domain block will have equal probability to come from D_0 as it has from D_1. This will then give us

$$p = \frac{r}{2}. \tag{5.5}$$

The channel capacity for BSC [22] is given by

$$C = 1 - H(p) \tag{5.6}$$

where H(p) is the entropy

$$H(p) = -p \log_2 p - (1-p) \log_2 (1-p). \tag{5.7}$$

This gives the channel capacity

$$C = 1 + \frac{r}{2} \log_2 \frac{r}{2} - (1 - \frac{r}{2}) \log_2 (1 - \frac{r}{2}) \tag{5.8}$$

for our type of watermark channel.

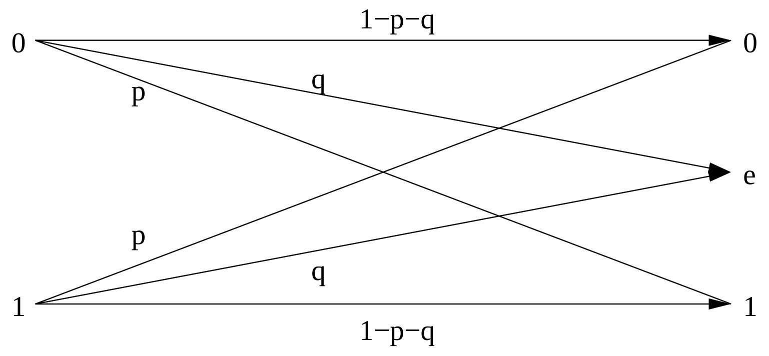

Figure 5.6: Binary Symmetric Erasure Channel (BSEC)

5.4.2 Extended watermarking channel

It is possible to extend the fractal block coding embedding as we first described in [34]. Let the number of domain regions be $d \geq 2$. For every position (range block) we let a pseudo random key decide which two domain regions that will be D_0 and D_1. Then we will have the same input $X_i \in \{0, 1\}$ as before but a larger possibility of outputs. Let us call all regions which are not D_0 or D_1 for D_e with e standing for "erasure". This give us $Y_i \in \{0, 1, e\}$. This type of watermark channel is called a **Binary Symmetric Erasure Channel** (BSEC) [35]. Figure 5.6 describes BSEC.

The probability p and q will depend on the probability of incorrect mapping r. For successful attack on one position we assume that an incorrect domain block will have equal probability to come from any of the d domain regions. We will then get

$$p = \frac{r}{d} \tag{5.9}$$

$$q = r\frac{d-2}{d}. \tag{5.10}$$

From (5.1)-(5.4) the channel capacity for BSEC can be calculated as

$$C = (1-q)(1-\log_2(1-q)) + (1-p-q)\log_2(1-p-q) + p\log_2 p \tag{5.11}$$

which for our watermarking channel gives

$$C = (1 - r\frac{d-2}{d})(1 - \log_2(1 - r\frac{d-2}{d}))+ \tag{5.12}$$

$$+(1 - r\frac{d-1}{d})\log_2(1 - r\frac{d-1}{d}) + \frac{r}{d}\log_2\frac{r}{d}.$$

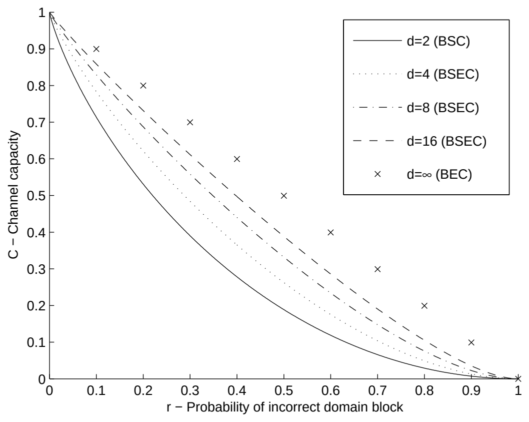

Figure 5.7: Channel capacity with $d = 2, 4, 8, 16$ and ∞.

In figure 5.7 we can see how the channel capacity increases with d. That means that the amount of information in the watermark which is not destroyed by the attack is increasing. For the case $d = 2$ we have a BSC and for

$$\lim_{d\to\infty} C = 1 - q = 1 - r \qquad (5.13)$$

we have reached a **Binary Erasure Channel** (BEC) [22]. The channel capacity for BEC is also shown in figure 5.7. We can see BEC as an upper limit for the channel capacity for this watermarking algorithm since BEC is BSEC with $p = 0$.

When we have BSEC with $d > 2$ we can use error correcting codes to approach the channel capacity. If we do not use error correction codes the watermarking can be seen as both having a fragile and a robust part. The fragile part will tell us if the image has been attacked in some positions ($Y = e$). That information can then be used during identification of the pirate. So BSEC will work to the advantage of the distributor even if error correcting codes are not used.

If we increase d the image quality may decrease. At the same time the robustness of the mappings will probably be effected in a negative way. Therefore we must try to find a suitable size of d.

The BSEC could be used on other watermarking methods which have the same attack response (totally random if position is “lost”). We have so far not found any other embedding method suitable for BSEC.

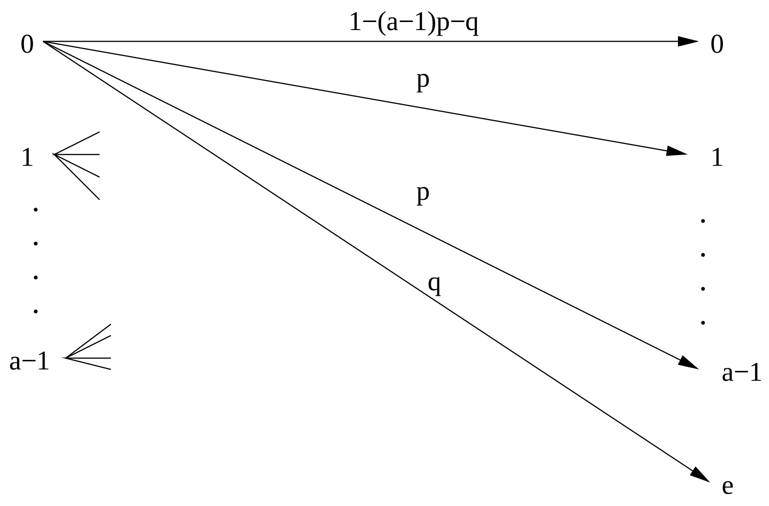

Figure 5.8: N-ary Symmetric Erasure Channel (NSEC)

5.4.3 N-ary input

Since the number of domain regions d has increased the number of input values can also be increased $2 \leq a \leq d$. For every position (range block) we let a pseudo random key decide which a domain regions that will be $D_0, \dots D_{a-1}$. Then we will have the input $X_i \in \{0, \dots, a-1\}$ and the output $Y_i \in \{0, \dots a-1, e\}$. Since we have not found this type of channel in the literature we have decided to call it **N-ary Symmetric Erasure Channel** (NSEC).

Figure 5.8 describes NSEC. For NSEC we have a input symbols and $a+1$ output symbols.

The probability p and q will depend on the probability of incorrect mapping r. If we once again assume that an incorrect domain block will have equal probability to come from any of the d domain regions we will get

$$p = \frac{r}{d} \tag{5.14}$$

$$q = r\frac{d-a}{d}. \tag{5.15}$$

From (5.1)-(5.4) the channel capacity for NSEC can be calculated to

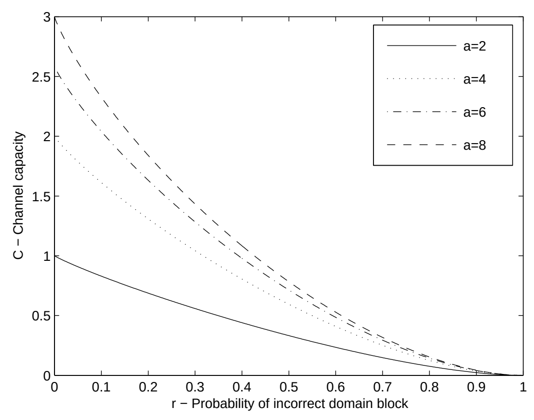

Figure 5.9: Channel capacity for NSEC with $d = 8$ and $a = 2, 4, 6, 8$.

$$C = (1-q)(\log_2 a - \log_2(1-q)) + \tag{5.16}$$
$$+(1-(a-1)p-q)\log_2(1-(a-1)p-q) + (a-1)p\log_2 p$$

which gives

$$C = (1 - r\frac{d-a}{d})(\log_2 a - \log_2(1 - r\frac{d-a}{d})) + \tag{5.17}$$
$$+(1 - r\frac{d-1}{d})\log_2(1 - r\frac{d-1}{d}) + r\frac{(a-1)}{d}\log_2\frac{r}{d}.$$

This is a general expression of the channel capacity for our watermark channel. If $a = 2, d = 2$ we have a BSC (5.8), and if $a = 2, d \geq 2$ we have a BSEC (5.12).

The channel capacity for NSEC is shown in figure 5.9 with $d = 8$ and $a = 2, 4, 6, 8$. As can be seen the capacity increases when a increases. If we further investigate (5.17) it is seen that this is always true. So from a channel capacity point of view it is best to have a value of a that is as large as possible. That means that we should use $a = d$. To use a BSEC is therefore not the best way to increase the channel capacity, instead NSEC should be used. Since for the case $a = d$ there is no erasure symbol at the output ($q = 0$) a better name for that case would be **N-ary Symmetric Channel** (NSC). NSC with as large d as possible will give the best capacity!

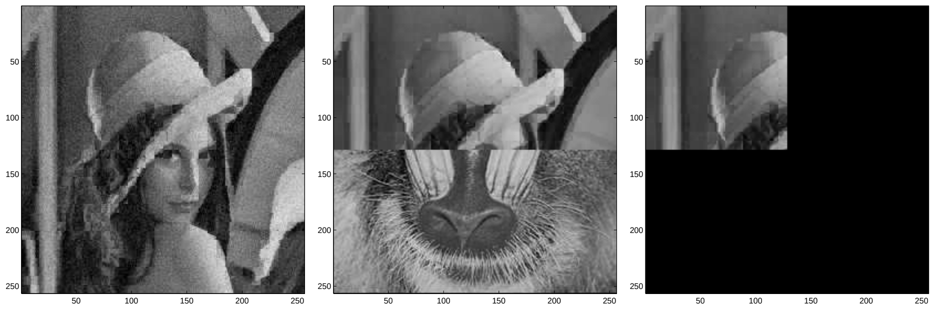

Figure 5.10: Three types of attacks. (a) Added Gaussian noise. (b) Edit image to another image. (c) Cropped image.

If the size of the watermark is suitable for BSEC it will be a better channel to use than NSEC. This comes from the fact that the knowledge about erased positions q for a given d is highest for BSEC, compare q for BSEC (5.10) with q for NSEC (5.15). So from a channel capacity point of view NSEC is best but if we want to maximize the knowledge about erased positions BSEC is best. Because of this we will further investigate $d = 2$ (BSC) and $d = 8$ (BSEC) in the next section.

5.5 Embedding using BSC and BSEC

In this section we will make our own embedding using fractal block coding. First we will use a BSC watermarking channel similar to what has been done before. After that the extension with BSEC will be investigated.

5.5.1 BSC watermarking

In the following experiment the gray scale Lenna image with 256×256 pixels was used. The amplitude range of each pixel is $[0, 255]$. The size of the range blocks was set to 8×8. This gives a binary watermark of length 1024. To better illustrate the damage done by the attacks the watermark will be represented by a binary image of an "E" [1] containing 32×32 pixels, figure 5.11 (a).

[1]The "E" comes from the logotype of the ECSEL research school that the author was attending when he started with this research.

Type	Spatial contraction	Domain region	d	Extraction
1	Sub	All	2 (BSC)	Public key
2	Sub	Local	2 (BSC)	Public key
3	Average	Local	2 (BSC)	Public key
4	Average	Local	8 (BSEC)	Public
5	Average	Original	8 (BSEC)	Private

Table 5.1: The different types of embedding tested.

Three different attacks will be investigated:

Noise Adding Gaussian noise with expected value zero and standard deviation 10 to the embedded image, figure 5.10 (a).

Edit Editing the upper half of the embedded image into the baboon image, figure 5.10 (b).

Crop Cropping the embedded image and only keeping 1/4 of the image, figure 5.10 (c).

These attacks are not designed specifically against fractal block coding embedding. To be robust against a designed attack the position and size of the range and domain blocks should be kept secret. Fractal block coding can in itself be an efficient attack according to [36].

We use the index of the location of the domain block to describe if the block belongs to D_0 or D_1. A number of different parameters were changed to see what setting would be most robust. Table 5.1 lists the different parameters used for the embedding. Embedding type 1-3 are BSC and will be described in this subsection. Type 4-5 are BSEC and will be described in the next subsection. The spatial contraction of the domain blocks to the size of the range block can be done by sub-sampling or averaging. The domain blocks region can be the whole image or only a region close to the corresponding range block. Type 1-3 correspond to public key watermarking if the algorithm is known since neither the original image, the watermark nor any key is needed to extract the watermark.

In table 5.2 we list the result of the extraction after the three different attacks on the five types given in table 5.1. The “Correct symbol” column gives the number of symbols that is correctly extracted after the attack. If all symbols

Type	Attack	Correct symbol	Correct block $(1 - r)$
1	Noise	0.6611	0.3428
1	Edit	0.6387	0.2363
1	Crop	0.5723	0.0479
2	Noise	0.6982	0.4014
2	Edit	0.7070	0.4180
2	Crop	0.6113	0.1719
3	Noise	0.7354	0.4541
3	Edit	0.7344	0.4238
3	Crop	0.6221	0.1777
4	Noise	0.4453 (0.8492)	0.3818 (0.7281)
4	Edit	0.4893 (0.8743)	0.4268 (0.7627)
4	Crop	0.2861 (0.7399)	0.1777 (0.4596)
5	Noise	0.5303 (0.8960)	0.4785 (0.8086)
5	Edit	0.5547 (0.9030)	0.5000 (0.8140)
5	Crop	0.3477 (0.7964)	0.2510 (0.5749)

Table 5.2: Result from different attacks.

were extracted correctly this will be 1 and if no one was extracted correctly it will be 0. Since the correct symbol does not mean that the symbol was extracted from the correct domain block we also list correct block in a similar way.

If we compare the results from type 2 and 3 we see that it is slightly better to use averaging for the spatial contraction compared to sub-sampling. For the noise attack the difference is most notable and that comes from the fact that the domain blocks are more robust against that attack if averaging is used.

Figure 5.11 and 5.12 show the binary watermark image before embedding and after extraction for embedding type 1 and 3. The location of the symbols in the watermark image corresponds to the location of the range blocks that

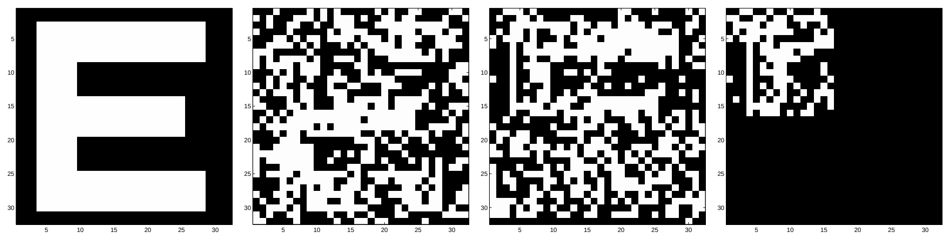

Figure 5.11: Embedding type 1. (a) Binary watermark image. (b) After attack with noise. (c) After attack with edit. (d) After attack with cropping.

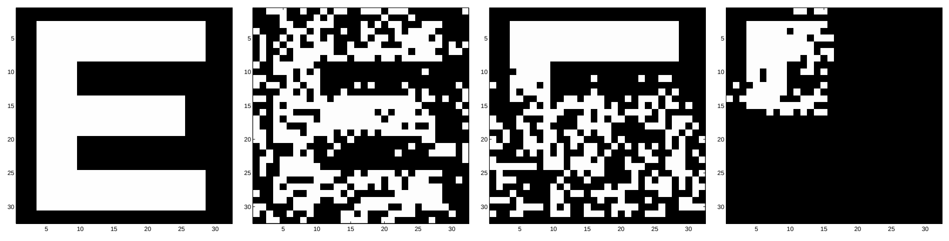

Figure 5.12: Embedding type 3. (a) Binary watermark image. (b) After attack with noise. (c) After attack with edit. (d) After attack with cropping.

they are embedded in. This is not an ideal way to embed the watermark since the characteristics of the watermark can become visual in the embedded image. It can also be less visually robust against some attacks. Instead the location of the symbol and the location of the corresponding range block should be pseudo-random. In this example these binary watermark images is used only to illustrate the damage caused by the different attacks.

Comparing figures 5.11(c) and 5.12(c) it can be noticed that for type 3 the watermark is not damaged in the upper part. This comes from the fact that neither the range blocks nor their domain blocks are affected by the attack since the domain region is local. For type 1 that uses the whole image as domain region also the upper part is damaged by the attack. Similar results can be found in figures 5.11(d) and 5.12(d.). A local domain region is therefore preferable.

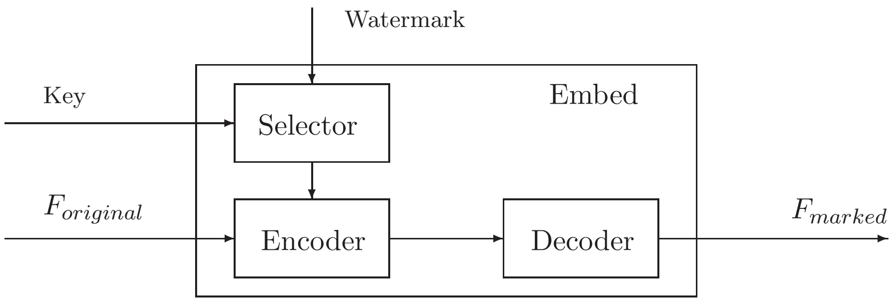

Figure 5.13: Embedding the watermark with a key.

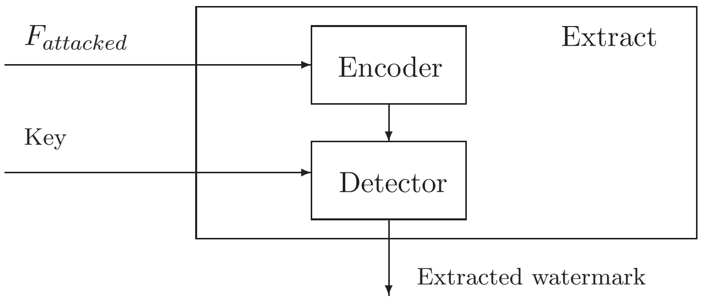

Figure 5.14: Extracting the watermark with a key.

5.5.2 BSEC watermarking

In this subsection we will use $d = 8$ (embedding type 4-5). The index of the location of the domain block will once more be used to decide which domain region it belongs to. We will have eight domain regions but will only use two of them as input symbols. This watermarking channel is then a Binary Symmetric Erasure Channel. Which two domain regions that are used is selected by a key, see figure 5.13. This key is also used to detect the watermark in the extraction, see figure 5.14. If a symbol extracted for a range block is not one of the two possible for that range block, given by the key, that position in the watermark is detected as erased by an attack.

In table 5.2 we list the result of the extraction of embedding type 4 and 5. We know that a symbol that is detected as erased is not the correct symbol and that the mapping does not come from the correct block. If we ignore these erased positions we will get the results within parenthesis in table 5.2. This

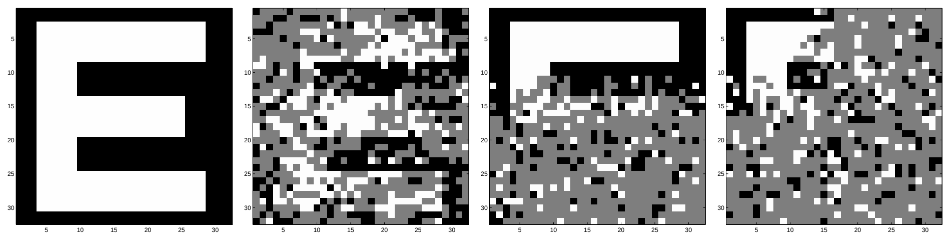

Figure 5.15: Embedding type 4. (a) Binary watermark image. (b) After attack with noise. (c) After attack with edit. (d) After attack with cropping.

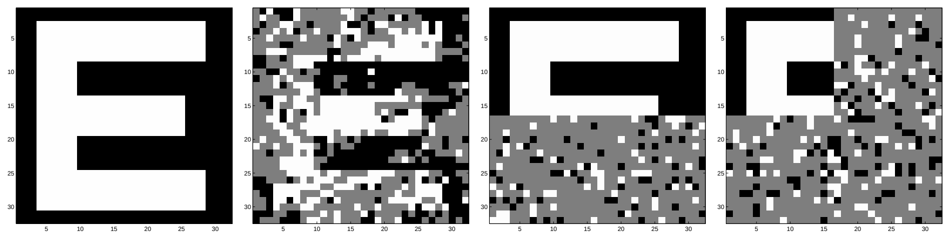

Figure 5.16: Embedding type 5. (a) Binary watermark image. (b) After attack with noise. (c) After attack with edit. (d) After attack with cropping.

will increase the number of correct symbols among the considered symbols and the same for correct blocks.

The extraction of the symbol for type 4 is public since only the key is needed. One problem with this method is that the increased domain region (four times as large as for ordinary binary embedding) will increase the possibility to find an incorrect domain block for the range blocks. This can especially be a problem if the domain block regions are close (and therefore similar) to each other.

Figure 5.15 shows the watermark image for embedding with type 4. The gray pixels comes from the symbols that are detected as erased. In figure 5.15(d) we see that most of the cropped area is found to be erased. That information can be used for automatic correction of the watermark.

Embedding type 5 always uses the original image for the domain region. This means that to extract the watermark the original image must be used. So this is a private watermarking. Similar to type 4 we use BSEC with eight domain regions and a key to select two of them for each range block. In figure 5.16(c) and (d) we can see that the watermark is correct for the part of the image that has not been edited or cropped. This comes from the fact that no attack can destroy the domain regions since they come from the original image.

5.6 Conclusions

In this chapter we have shown how fractal block coding can be used to embed watermarks into images.

We have extended the number of domain regions and used channel capacity reasoning for describing the watermarking channel. It is seen that the N-ary Symmetric Channel will give the largest amount of data in the watermark. If we instead are interested in knowing which positions that have been erased by attacks the Binary Symmetric Erasure Channel should be used.

Finally we have tested some methods for watermarking with fractal block coding using the Binary Symmetric Channel and the Binary Symmetric Erasure Channel. We have found that to be robust against cropping and editing attacks it is best to have the domain regions close to the range block.

Chapter 6

Distribution of individual watermarks

In this chapter we will describe distribution methods and discuss how individually watermarked copies can be distributed in a scalable way. Methods to distribute individual watermarks are reviewed.

6.1 Distribution

For many years the distribution of TV (terrestrial, satellite and cable) was done by analog signals. Home video formats like VHS and Laser disc [1] were also analog. Nowadays most of the TV that is distributed is digital and it follows standards such as the DVB [37]. Home video formats like DVD and VCD are also digital. Therefore we will only deal with digital distribution in this thesis.

When using computer networks digital distribution is natural and the media being distributed is split into **packets**. We will see the distribution chain as a **server** owned by a distributor, **the network** and a number of **clients** each belonging to a receiver. In the network there are **routers** that will transmit the packets between the server and the clients.

[1]Actually the audio in Laser disc could be analog or digital (like AC-3 or DTS) but the image frames were always stored by frequency modulated analog signals.

Packets can be transmitted through a network using three different ways, unicast, multicast and broadcast [38]. These methods will be described in the following subsections.

6.1.1 Unicast

In **unicast** one packet is sent from the server to the client in the network. If the distributor wants to send the same information to ten receivers the server has to send the same packets ten times, once to every client. So this is not a very efficient way to transmit the same data to many receivers. Figure 6.1 illustrates unicast distribution.

Unicast is the transmission method that is most frequently used today but in the future more efficient methods will hopefully be used.

6.1.2 Multicast

In **multicast** the server will only need to send each packet once. The clients of authorized receivers will belong to a multicast group. In the network the routers will know which of its downstream branches that contain members in the multicast group. A router will multiply the packets and transmit them on all branches with clients in the multicast group. That way all authorized receivers will receive the same packets that the distributor transmits. Figure 6.2 illustrates multicast distribution.

6.1.3 Broadcast

In **broadcast** the server will only send each packet once. The routers in the network will multiply and forward the packet downstream so all routers and eventually all clients will get it. It does not matter if the receivers are subscribed to the distributor or not, they will all get the packet. If the packet is encrypted non-subscribed receivers will not be able to decrypt it. Figure 6.3 describes broadcast distribution in a network.

Broadcast distribution is in many ways similar to TV broadcasting over terrestrial and satellite. Everyone should be able to get access to the transmitted packets as long as they have the appropriate equipment to receive the data. Video distribution over multicast and broadcast is described in [39] and [40].

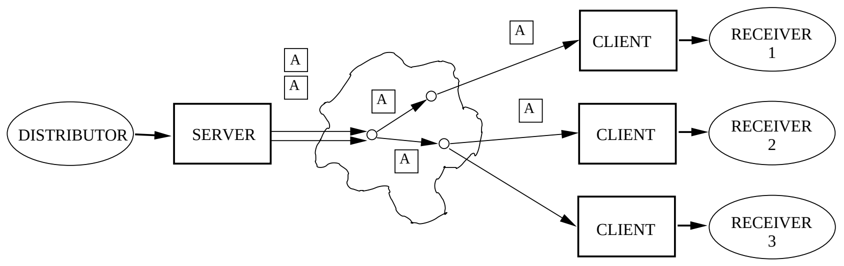

Figure 6.1: Unicast distribution. The server is going to distribute packet A to receiver 1 and 2. To do that the server must transmit packet A two times, once to receiver 1 and once to receiver 2.

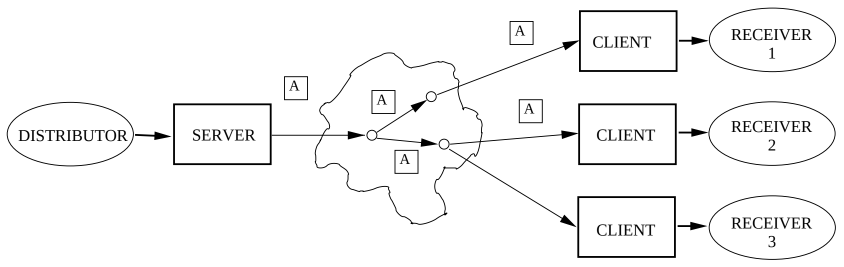

Figure 6.2: Multicast distribution. The server is going to distribute packet A to receiver 1 and 2. The distributer transmits the packet only one time. The first router forwards the packet to both downstream routers. Router two forwards to receiver 1 and router three forwards the packet to receiver 2.

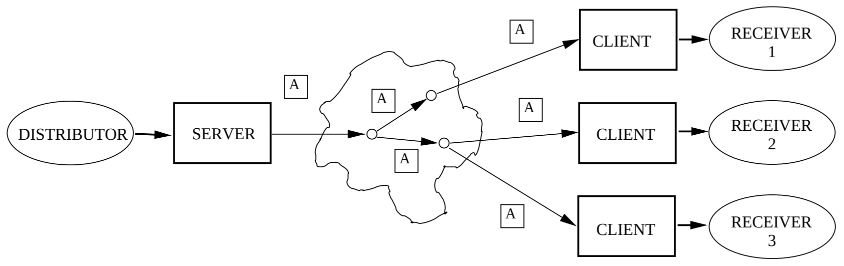

Figure 6.3: Broadcast distribution. The server is going to distribute packet A to receiver 1 and 2. The distributer transmits the packet only one time. The routers forwards the packet on their downstream branches. All receiver gets the packet (even receiver 3).

6.2 Watermarking and distribution properties

The distributor wants to use multicast or broadcast to transmit the media object to the receivers. At the same time he wants to individually watermark each copy, so that every receiver will get an individual copy. But in multicast and broadcast the packets that reach the receivers are all the same. So how can we transmit individual watermarks using multicast or broadcast?

Different approaches for doing this will be presented in section 6.3 to 6.5 and in chapter 7. In the following subsections we will discuss some properties that is important for the distribution of individual watermarked objects. In appendix A these properties will be used to summarize the different methods.

6.2.1 Individual watermarking location

One of the most important part of an individual watermark distribution is of course the embedding of the watermark. Individual watermarks does not need to be embedded in the usual way. Instead other types of manipulation of the copies can create the individuality. The location of this individual watermarking can be the server, the network and/or the client.

In section 6.3 methods that create the individual watermarks in the server will be described. Network and client based methods are described in section 6.4 and 6.5.

6.2.2 Dependence of the network

Another important aspect of the distribution method is if a specially designed network is needed or if any ordinary network will do. If for example the embedding is done in the network the method is dependent on it. A method that is independent of the network is easier to implement using existing network infrastructure. The best type of distribution methods are such methods that not only can be used in networks but also for TV broadcasting.

6.2.3 Data expansion

To introduce the uniqueness most methods need to expand the amount of data that is transmitted. This will then increase the needed bandwidth which is often a problem. Methods that can transmit the data without any or only a small amount of data expansion and still achieve a robust individual watermark are desired.

For some methods special decryption keys are transmitted individually to each receiver. The data transmission of these keys must also be taken into account then the data expansion is calculated.

6.2.4 Sparse or dense watermarks

The individual watermark can be sparse or dense, section 3.2. Dense watermarks are preferable since less data is needed to identify the pirate. It is desired that each frame in a video stream contains the individual watermark, so only one frame is needed during extraction.

6.2.5 Robustness against collaborating pirates

Collaborating pirates can try to remove the individual watermarks using their copies. Preferably the embedding should be robust against that type of attack and the position of where the watermark is embedded should be difficult to find.

6.3 Server based individual watermarking

In this and the two following sections we will describe some methods for distributing individually watermarked media objects. In this section a method that makes the individual marking in the server is presented. It is the location of the individual watermarking that we use for our classification whether the method is server, network or client based. Therefore methods that only embed regular non-individual watermarks in the server will be classified as network based embedding or client based embedding.

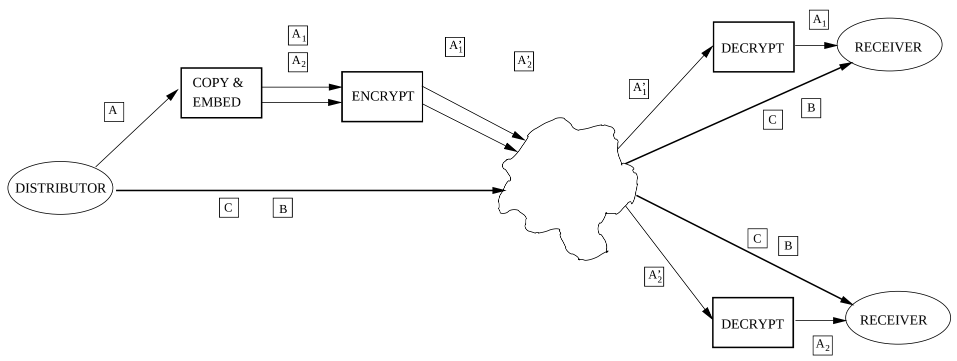

Figure 6.4: Scheme that combines unicast and multicast. Some packet (B and C) are transmitted using multicast. Packet A is made into two copies embedded with individual watermarks. These packet copies are encrypted with individual keys and transmitted by unicast to one receiver each. That way parts of the stream will be individual for each receiver.

6.3.1 Combined unicast and multicast

In 1997 Wu and Wu [41] described a method to transmit individually watermarked data to many receivers without all of the data being transmitted by unicast. The solution is simply to split the data into one multicast part (or broadcast part) and one unicast part. The packets in the multicast part can be read by all receivers while the packets in the unicast part are only transmitted to and read by one receiver. The unicast part is embedded with an individual watermark and encrypted in the server before it is transmitted to the corresponding receiver. Figure 6.9 describes the method.

The watermarking method used is an extension of the spread spectrum embedding. The authors have tested seven different locations for the mark from only the DC part of the luminance in I-frames to both luminance and chrominance of I-, P- and B-frames. They use dense watermarks and only one embedded frame is needed to extract the watermark.

We can note that to be scalable to many receivers only a small part of the data can be transmitted by unicast. The smallest embedding results in less than 1% being transmitted with unicast while the largest embedding gives that 90% of each receivers streams are transmitted with unicast. But if we have 1000 receivers the smallest embedding would still give a total data expansion of 10 times compared to video without any individual watermarking. This method will therefore only work for transmission to small amounts of receivers.

The table in appendix A describes how this methods deals with the properties given in section 6.2. A new and better version of this distribution method is presented in [42]. In this version the bandwidth requirement can be reduced by up to 87% compared to an all unicast distribution.

6.4 Network based individual watermarking

In this category the individual watermarking is done in the network. To achieve this a smart overlay network needs to be implemented. Some methods for doing this will be presented below.

6.4.1 Watercasting

In 1999 Brown, Perkins and Crawford introduced **Watercasting** [43]. This combination of individual watermarking and multicast uses an overlay network with special routers to handle the watermarking. Every packet that is going to be a watermarked position is made into n different copies. Each of these packet copies are embedded with an individual mark but contains the same basic information (no perceptual difference). All of these copies are then transmitted from the server to the network. The overlay network routers will not act like ordinary routers since they will not forward all packet copies. Instead the routers will forward all but one of the packets to each downstream router. That way all underlying routers will receive different set of packet copies. Figure 6.5 describes the method in more detail. From the last overlay router only one copy of each packet is transmitted to the subnet of receivers.

For the next packet the same thing is done but now other packet copies are intentionally dropped in the network. It is this dropping of copies that

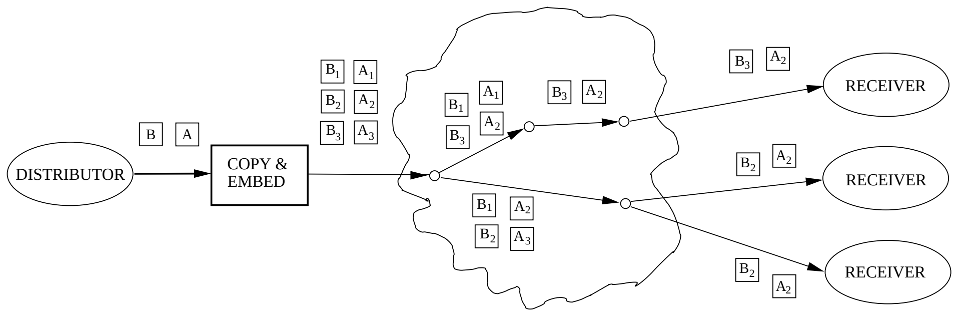

Figure 6.5: Watercasting. At the server the packets are copied and embedded with different marks, A_1, A_2 and A_3. The first router forwards packet A_1 and A_2 to the second router and packet A_2 and A_3 to the third router. From the last router to the receiver only one packet of each copy is transmitted. The same is done with packet B but now other packets are dropped. Receivers on different subnets will receive different sets of packets.

results in an individual watermarked set of packets giving an unique media object. So even if the watermark embedding is done at the server it is the active network that gives the individual watermarking.

If the depth of the tree of overlay routers is d the minimum numbers of copies of each packet n is $n \geq d$. If that is fulfilled all last hop routers will receive at least one copy of each packet. The needed bandwidth will be larger at the beginning and will then decrease during transmission and finally only one packet from each set of packet copies are transmitted to the receiver.

The tree topology must be stored and also the information about which packet copies that were dropped for every packet. The dropping can be done in a repeated way so that information will be easier to store.

One problem with this method is that all receivers on the same subnet will receive the same packets. Another problem is that the method does not seem to be able to handle collaborating pirates from different subnets.

6.4.2 Hierarchical tagging

Caronni and Schuba [44] described two different methods for distributing individual watermarks in an efficient way. One of the methods called **Hierarchical tagging** is based upon the idea that there is more than one distributor. The original distributor has to his help a hierarchical network of sub-distributors that will distribute the media object until it reaches the receiver. These sub-distributors will be required to embed individual watermarks for all of their sub-distributors. From the last sub-distributor to the receiver the media is individually watermarked.

The idea is that all of these watermarks should be possible to extract from the copies. That way the distributor can extract the watermark that tells him who his sub-distributor was. The sub-distributor can extract the information about the next hand in the transmission chain and so on until the receiver is identified.

This method is dependent on the existence of a network of distributors. The next method will use a similar method and implement it in a network.

6.4.3 WHIM - Backbone

In 2000 Judge and Ammar [45] created a system that "watermarks in a hierarchy of intermediaries", **WHIM**. This method contains two parts, the **Backbone (WHIM-BB)** and the **Last Hop (WHIM-LH)** (see next section). Figure 6.6 describes both WHIM-BB and WHIM-LH.

For WHIM-BB an overlay network is created that contains special routers that can embed watermarks. The network is adapted for multicast. If for example images are transmitted each of these overlay routers will embed a watermark in the image. The watermarks are unique for the routers and each router will always embed the same watermark in the images. That way an image will contain identification watermarks from all routers that it passed during transmission from the distributor to the receiver. This can be used to trace the transmission chain from the server to the last router.

Similar to Watermarking the individual watermarks will only be able to identify the subnet of the receiver. This will be fixed by WHIM - Last Hop. The properties of WHIM-BB are given in appendix A.

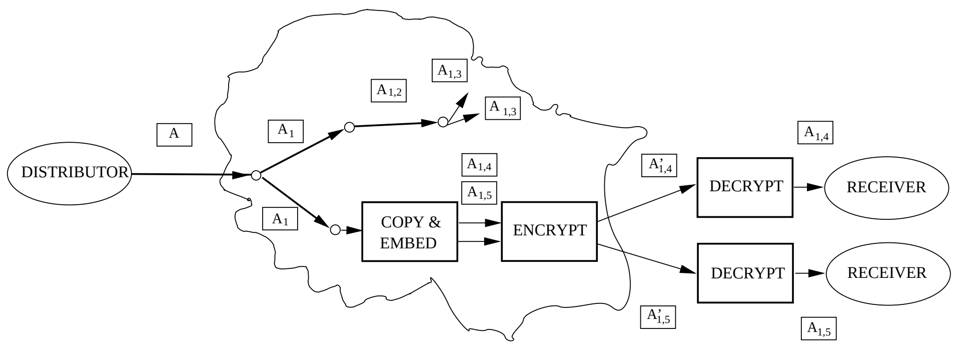

Figure 6.6: WHIM-BB - The routers in the network embeds their watermark in the transmitted images. WHIM-LS - The last router makes copies of the images and marks them individually. Encryption is done before the images are transmitted with unicast to the different receivers. Each received image contains information about which routers it has passed and a watermark individual to the receiver.

6.4.4 WHIM - Last Hop

Another part of the WHIM [45] method is the **WHIM - Last Hop**. This takes care of the last part of the transmission, from the last overlay router to the receiver. This method is also illustrated in figure 6.6.

WHIM-LS is constructed for the distribution from the last overlay router to the receiver. The router will individually watermark and encrypt the media before transmitting it to the subscribed receivers. That way all receivers in a subnet will get individual watermarked copies.

If a WHIM-LS watermarked copy is found it might be a problem to locate the subnet that the receiver belongs to. By combining WHIM-LS with WHIM-BB information about the whole transmission chain from the distributor to the receiver will be embedded in the media. Both of these methods use dense watermarking and can be made robust against collaborating pirates.

One of the big problems with the WHIM methods are that all the embedding will probably make the network slow. However, the main problem is that a special overlay network is needed for the distribution. These methods are therefore dependent on the network and can for example not be used for TV broadcasting.

If WHIM-LS is used for all overlay routers we will end up with a method similar to the Hierarchical tagging method from subsection 6.4.2.

6.5 Client based individual watermarking

To let the individual watermarking be done in the client is the category that most distribution methods use. One of the earliest methods to use this was the original Divx [2] format. When using a Divx disc an individual watermark was embedded in the video stream [7]. The Divx format relied on a trusted hardware. But to rely only on secure hardware can be dangerous. In this section we will describe methods that use client based individual watermarking without relying on secure hardware.

6.5.1 Joint source fingerprinting

Luh and Kundur [46] propose a similar scheme to [41] and [42] called **Joint source fingerprinting**, **JSF**. They split the video stream into a *semantic class* and a *feature class*. The semantic class is encrypted and transmitted with broadcast so that all receivers get and decrypt the same data. The feature class is made into multiple copies, one for every receiver. The feature class is then used for the individual watermarking and the embedding is done at the distributor. The feature class for all receivers is also encrypted and transmitted by broadcast. Every receiver has only one decryption key so he can only decrypt his own feature class part. That way every user will get an individual watermark. The distribution is described in figure 6.7. Because everything is broadcasted this method is categorized as client based, since

[2]Divx (Digital Video Express) was released in 1998 in USA. This was a pay-per-view rental version of the DVD home movie format. The format failed and came to an end in June 16, 1999. The DivX that is known today began in March 2000 as a compression method that used the name DivX;-) to mock the original Divx. The smiley was eventually dropped from the name and that has since then lead to much confusion.

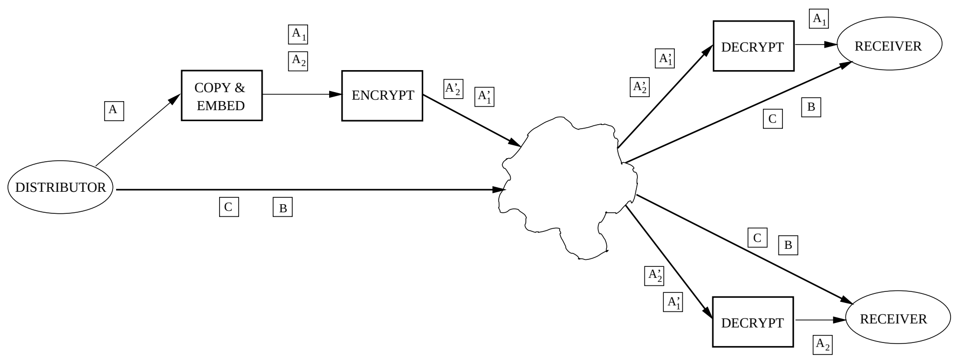

Figure 6.7: Joint source fingerprinting. Some packet containing the semantic class (B and C) are transmitted using broadcast. Packet A contains feature class information that is individual for all receivers. These feature class packets A are encrypted with individual keys and transmitted with broadcast. Each receiver can only decrypt one of the A_i packets. That way parts of the information will be individual for each receiver.

the uniqueness comes from the decryption keys in the client. The authors also discuss that the distribution of the feature class can be done by unicast in cases where unicast transmission is possible (not for TV broadcasting for example). For that case the method becomes a server based individual watermarking.

In the algorithm the semantic part has a low frame rate and will therefore have choppy motion that will give it no commercial value on its own. When combined with the feature class the video will not contain any visual degradation. The authors claim that the method will be robust against collaborating pirates because the quality of the video will be distorted by colluding attacks.

One problem when using broadcast is that the number of possible receivers must be small enough to let the individual information be part of the broadcast stream. The amount of data in the feature class is said to be small but if we are aiming for distribution to millions of receivers the bandwidth needed for the feature class will be huge. No description of how to combine the algorithm with a video coder to get compression is given.

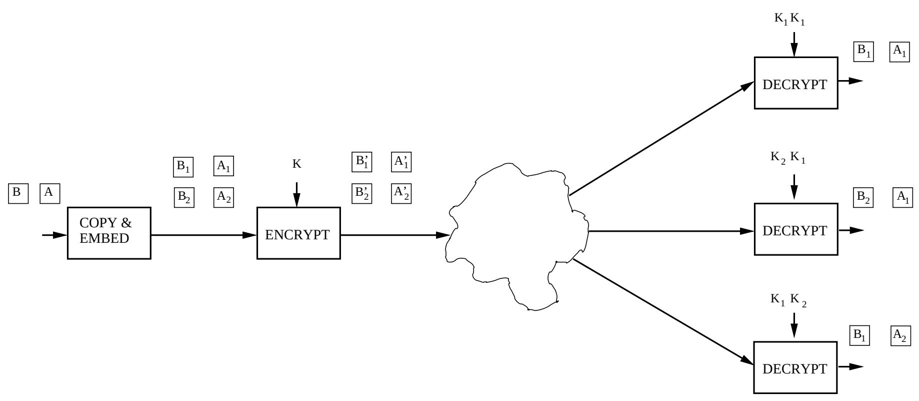

Figure 6.8: Packet based embedding. In the server the packet A is duplicated and the copies are individually watermarked. The packet copies A_1 and A_2 are encrypted with different keys and transmitted. The receivers can only decrypt one of the packet copies. After more packets are distributed the receivers will get different sets of packets combined to unique watermarks.

6.5.2 Packet based embedding

In 2001 Parviainen and Parnes [47] described a method that uses multicast to transmit individually watermarked radio. For every packet that is transmitted two copies (with different watermarks) are made. The packet copies are encrypted with different keys and all packets are transmitted with multicast. Every receiver has a unique set of decryption keys. These keys can only decrypt one of the two copies of every packet. That way every receiver will get access to different sets of watermarked packets, i.e. individually watermarked radio streams. See figure 6.8 for more details.

Compared to JSF the amount of transmitted data is not increased when the number of receivers increase. The needed bandwidth will always be double compared to a system with no individual watermarking. For the system to work well a good way of distributing the decryption keys is needed.

The watermarking will be sparse since the watermarks from many packets must be extracted before the individual set of marks can be identified. The system is not very robust against collaborating pirates since the positions of the individual watermark (every packet is one position) is known.

We can also expand the number of copies of each packet so that less amount of data is needed to identify the pirate. Such extension will cause higher bandwidth.

The main drawbacks of this method is the increase in transmitted data and the fact that we need a large part of the video before a pirate can be identified.

6.5.3 Frame based embedding

In 2002 Chu, Qiao and Nahrstedt [23] introduced a similar idea to [47]. Instead of using different copies of packets they use different copies of each frame. So every frame in a video stream exist in two versions embedded with different watermarks. The frames are encrypted with different keys. The video stream that is distributed will then contain two versions of each frame. Each receiver will be able to decrypt only one of these frames and the decrypted video stream will then contain an individual watermark.

The main problem with this method is that in video coders such as MPEG there is a dependence between the frames (P- and B-frames). Since the frames will be different the distortion will propagate. To solve this problem the coder can be constructed to transmit only I-frames but that will give low compression. Another possibility is to only embed in the I-frames but then the watermark will be even more sparse.

The sparse watermark is otherwise also a problem. The authors need between 14 minutes and 1939 minutes of video for the embedding of one watermark. The length depends on number of receivers and how many colluding pirates the mark should be robust against. By creating 8 copies of each frame the number of minutes needed will become 4 to 443. Such an extension will also increase the needed bandwidth with the same amount.

The method can be extended so that the watermark that is embedded in every frame contains information about the copyright holder. That way each frame contains a watermark with information about the distributor and the video stream contains an individual watermark.

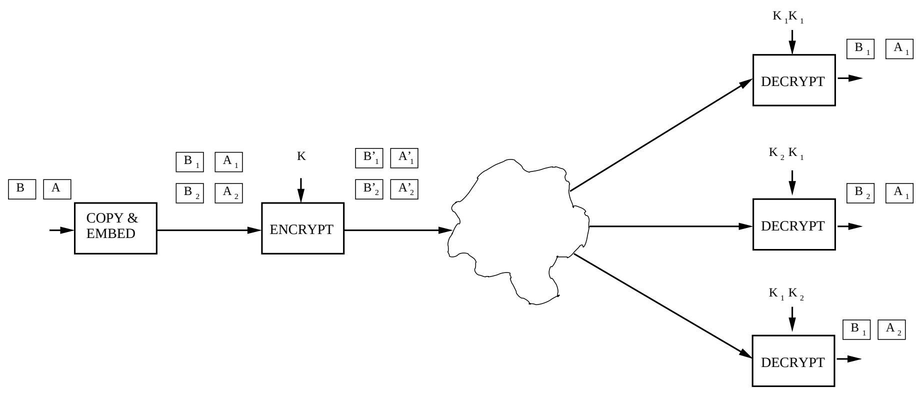

Figure 6.9: Frame based embedding. In the server the frame A is duplicated and the copies are individually watermarked. The frame copies A_1 and A_2 are encrypted with different keys and transmitted. The receivers can only decrypt one of the frame copies. After many frames are distributed the receivers will get different sets of frames combined to unique watermarks.

A **copy attack** [48] is an attack that, without any prior information about the watermark technology, can copy a watermark from one image to another. If used by a pirate on a frame based embedded video a watermark can be copied from one frame to another. If the same two watermarks are used on every frame the attack could totally destroy the individual set of marks that the receivers get. Ways of making the watermark copy-resistant to deal with copy attacks for this type of watermark schemes are described in [49].

6.5.4 Bulk tagging and COiN-Video

The second method described by Caronni and Schuba [44] is called **Bulk tagging**. This is a method with many similarities to packet based embedding and frame based embedding. One of their ideas is to transmit two versions of image blocks. That way one image can contain many positions for the individual watermark and the method is therefore not as sparse as Frame based embedding. The authors discuss that the scheme also can be used in distribution on a CD.

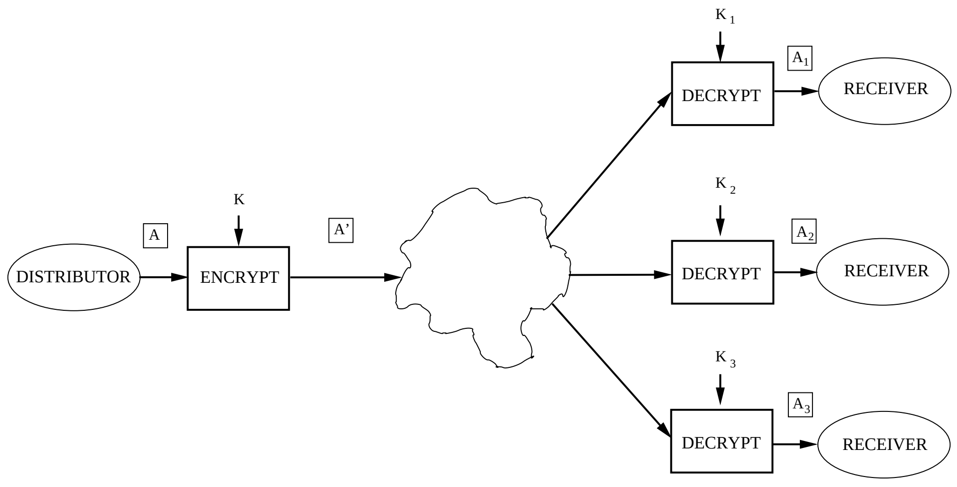

Figure 6.10: Chameleon. All packets are encrypted and transmitted with broadcast. Each receiver has an individual decryption key that will distort the decrypted media slightly. That way the information will be individual for each receiver.

A similar method is given in **COiN-Video** [50] by Konstantas and Thanos. For example they split the image in six blocks and transmit six copies of each block. That way the whole individual watermark can be contained in one frame of the video. The properties of Bulk tagging and COiN-Video are given in appendix A.

Boneh and Shaw [19] give a theoretical derivation of the size needed in the public data and in the private data for a similar distribution scheme.

6.5.5 Chameleon

Anderson and Manifavas [51] published their method **Chameleon** in 1997. This method is based upon a stream cipher and was originally developed for audio. The idea is that all receivers have different decryption keys. The encrypted audio is transmitted using broadcast and the decryption keys will decrypt it in different ways. So after the decryption the stream has been slightly distorted and it is this distortion that is the individual watermark. Figure 6.10 illustrates the method.

The authors work with 16-bit audio signals and the mark is embedded into the least significant bits. Their example is not so good since the embedding is not robust against attacks on the least significant bits. No compression is used so they need a lot of bandwidth to transmit the audio signal. The properties of chameleon are given in appendix A.

An extension of how to implement the Chameleon cipher is given in [52]. Tamper proof smart cards are used to contain the key. This improves the security by making it difficult for the receivers to make their own keys. Another implementation of Chameleon is given in [53].

More schemes based upon the Chameleon principle will be given in chapter 7.

6.6 Additional categorization

In the previous three sections we have presented the methods categorized as server based, network based and client based. In this section we will present another categorization of the methods for distribution of individually watermarked media.

- The network based system is still one category containing Watercasting, WHIM and Hierarchical tagging.
- Methods for which the receivers get unique parts of the stream form one category. For JSF and the combined unicast and multicast methods each receiver will get data that no one else receives. These methods are therefore contained in this category.
- For some methods the receivers can only decrypt parts of the transmitted data. This parts will then be combined to individual watermarks. Methods in this category are packet based embedding, frame based embedding, bulk tagging and COiV-Video.
- Finally we have one category with methods for which the receivers have unique keys that will treat the transmitted streams differently. The difference from last category is that the transmitted data is not duplicated. So far only Chameleon belongs to this category but more methods will be presented in the next chapter.

The methods in the first of the new categories have the weakness that they are dependent on the network. Methods in the second and third category will transmit much data that can not be used by all receivers. Consequently we will have data expansion for the methods in these categories. Methods in the fourth category do not need to increase the bandwidth in the same way. So methods that belong to the fourth category seem to be the type of method that we want for individually watermarked media.

6.7 Conclusions

We have presented a number of methods for distribution of individually watermarked media and categorized them as server based, network based and client based. An additional type of categorization has also been given.

The methods have been investigated to see where in the distribution chain the individual watermarking is done, if they are dependent on the distribution network and if they cause data expansion. Furthermore we have investigated if the watermarks are dense or sparse and if they are adopted to deal with collaborating pirates.

We note that none of the given methods can provide a robust embedding with quick identification that does not expand the data needed to be transmitted. Such a distribution system would be most interesting and a proposal for such a method is given in the next chapter.

Chapter 7

Modified-transform watermarking

In this chapter we introduce an individual watermark distribution scheme that is scalable and has a collusion resistant embedding. A new type of attack is described and measures of how to make the embedding scheme robust against that attack is presented.

7.1 Introduction

In the previous chapter we described some methods of distributing individually watermarked media over multicast or broadcast. We described different properties and how the methods worked. Simply put we can say that we want to have a distribution and embedding method that:

- Is independent of the network and can be used in TV broadcasting
- Gives no or only minor data expansion in the network
- Has dense watermarks
- Is robust against collaborating pirates
- Does not rely on secure hardware

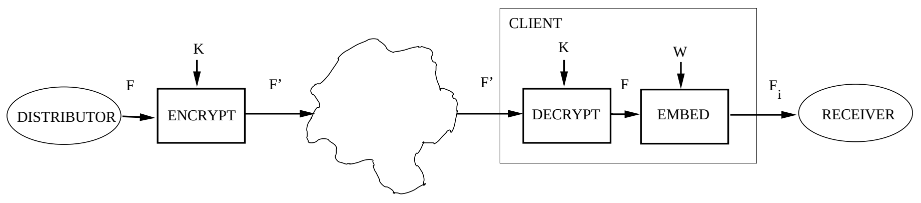

Figure 7.1: The frames are encrypted in the server. After transmission the client will decrypt the frame and thereafter embed the watermark.

In this chapter we describe how this can be achieved. The new method can be seen as a combination of encryption and marking. One way to combine these is to use secure hardware to both decrypt and embed in the client, see figure 7.1. There will always be ways that the original frames can be retrieved from such clients and then the watermarking will be lost. Consequently we can never rely on secure hardware. Instead we will use a similar idea to the Chameleon scheme but with better watermarking.

Before we describe our own method we will introduce two similar approaches made in the last years.

7.1.1 Mask blending

Emmanuel and Kankanhalli [54] have developed an interactive protocol that address security issues for both the distributor and the receiver. Contained in their DRM scheme is a simple and efficient way of embedding and distributing individually watermarked video. The idea is to mask the video at the distributor and transmit the same material to all receivers. To be able to view the masked video properly the receiver will need to unmask it. The unmasking frame is slightly different for all receivers and therefore the unmasked video will contain an individual watermark that can be traced back to the pirate.

The masking of the frames is done in the transform domain of the MPEG stream for faster embedding in already compressed video. The unmasking will be done in the spatial domain by the client. An unauthorized receiver that tries to view the masked video will only see a distorted version of it.

In this scheme two marks are actually embedded. One watermark to trace pirates and one for the receiver's concerns.

How the masking affects the compression ratio is not mentioned in the article. The data expansion will probably not be large compared to coding without Mask blending.

7.1.2 JFD

In 2004 Kundur and Karthik [55] created **Joint fingerprinting and decryption (JFD)** which is an extension of the Chameleon algorithm. In JFD the whole image is transformed using DCT. A set of perceptually significant regions in the low- and mid-frequency domains are identified. These sets are portioned into subsets. During encryption the subsets are sign scrambled, meaning that the sign of the coefficients are changed depending on a key. This encrypted image is transmitted using multicast or broadcast. The receiver will only be able to decrypt a portion of the encrypted subsets. The remaining scrambled subsets are then the embedded watermark. Every receiver has individual decryption keys that descramble different coefficients and therefore the copies will have individual watermarks. The individually watermarked copies will contain some distortion but they will all be close to the original image.

The encrypted image is still possible to receive and view but the scrambling makes the image quality poor and therefore of little commercial value.

The method is not adapted for video compression since it is the whole image that is DCT transformed in the embedding. Therefore the embedding will probably affect the compression factor in a negative way during video encoding.

In appendix A we can see how Mask blending and JFD holds compared to the other methods given in chapter 6.

7.2 Generalization of distribution methods

In this section we will give a general description of the watermarking methods Mask blending and JFD. Later on we will present a new method that also

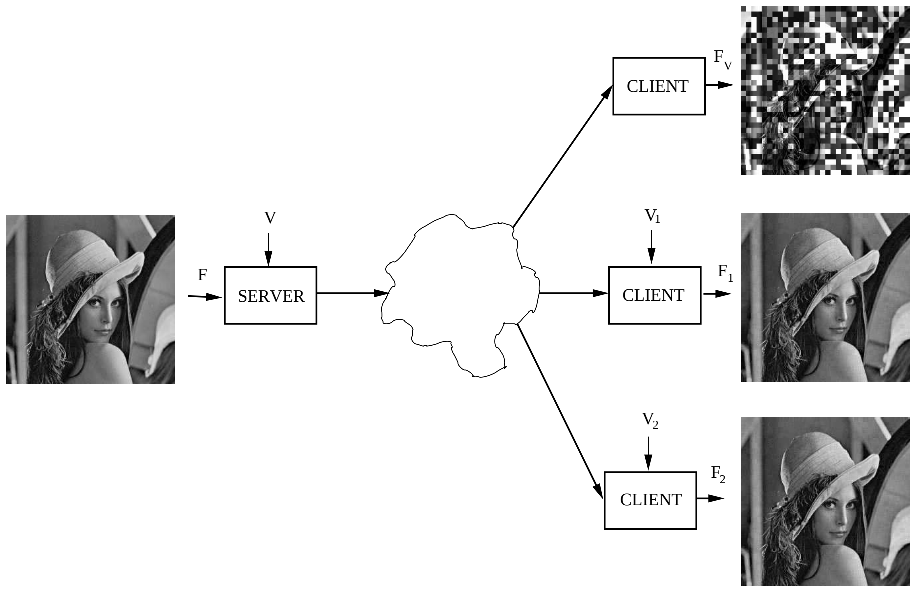

Figure 7.2: Distribution system. The offset watermark V is embedded at the server. If an unauthorized receiver tries to view the image it will be distorted. Authorized receivers will use their individual keys V_i to remove the offset watermark. Their images F_i will then contain an individual watermark.

can be described by this generalization. This description falls under the fourth category given in section 6.6. The method is illustrated in figure 7.2.

We have a frame of the video stream F. In the server at the distributor we have a type of embedding method "$\triangleleft$" that we use to embed an **offset watermark** V to the frame. The new frame

$$F_V = F \triangleleft V \tag{7.1}$$

is transmitted over the network. If a receiver views F_V he will only see a heavily distorted version of F. To see a proper image the offset watermark must be removed by the client. The removal "$\triangleright$" is a kind of inverted embedding with $(F \triangleleft V) \triangleright V = F$.

The removal is done with an individual key

$$V_i = V \oplus W_i \tag{7.2}$$

where W_i is an individual watermark unique for each receiver i. We will sometimes call V_i the **decoding key**. The operation "$\oplus$" will depend on the method but it is constructed so that the decoding key V_i will always be close to the offset watermark V. After the removal we have the received image

$$F_i = F_V \triangleright V_i = F \blacktriangleleft M_i \tag{7.3}$$

with M_i being the individual information that has been embedded into F. The embedding "$\blacktriangleleft$" can be a new type of embedding or the same embedding "$\triangleleft$" that was used for the offset watermark. For some of the methods (e. g. Mask blending) $M_i = W_i$ and for other methods M_i will both depend on W_i and other non-unique information, e.g. V. So the individual watermark W_i is embedded in the received frame F_i. The embedding of W_i will slightly distort the image but not affect the visual quality, i.e. F_i will be close to F.

The key V_i can be included in the client or distributed in a secure way so that no other receivers than i can read his key. If the keys are distributed we can change offset watermarks and decoding keys at some given times to keep the system more secure.

For Mask blending "$\triangleleft$" is an adding of noise to the image and for JFD "$\triangleleft$" is the scrambling of the signs of the transform components. Chameleon in the previous chapter was made for uncompressed audio but otherwise this generalization works for that method too. For Chameleon the "$\triangleleft$" is the encryption of the uncompressed audio. Other types of embedding are also possible to use for similar distribution methods.

7.3 The test pattern attack

For the Mask blending and JFD method there is a special type of attack that pirates can use to destroy the watermarks. We call this adapted attack the **test pattern attack**. If the distributor at some point distributes a known frame this can be used to reverse engineer the embedding. The known frame can be a test pattern, an often used image or maybe just a blank frame. Let us call this known frame F. Then the pirate have his F_i frame, the distributed F_V frame and the known F frame. If the embedding is known it could be possible to use (7.1) to retrieve V from F_V and F. In a similar way (7.3) can be used to retrieve W_i (or M_i) from F_i and F.

A pirate that knows the offset watermark V can use it instead of V_i in (7.3) and that way retrieve the original frame F without any individual watermark. This can be done for all frames in the video stream until the offset watermark V is changed by the distributor.

If the pirate instead knows W_i he can retrieve V from V_i using (7.2). The offset watermark V can then be used to receive F as described above. Depending on the embedding method it might be possible to use W_i to directly retrieve F from F_i. Both V and W_i can be used to retrieve the original frames F in the video stream and none of them should therefore be possible to retrieve by the receiver.

For Mask blending the test pattern attack will find the mask and the pirate will therefore be able to remove the mask from other frames. For JFD and Chameleon the test pattern attack will find the locations of the positions where the individual embedding is done. Thus a pirate can concentrate his next attack to those positions and probably be able to remove the individual watermark.

One way to deal with the test pattern attack is to avoid transmitting known frames but that is difficult to achieve. Another way is to often change V and V_i but that will expand the amount of transmitted data. A better way to deal with it is to have an embedding method that will make it impossible to retrieve V and W_i even if F_i, F_V and F are known. In the next section we will present a method to achieve this.

7.4 Embedding by Modified-transform

Usually watermarks are embedded in images in the transform domain. Some methods use the spatial domain for the embedding. We will use a third location for the embedding, the transform itself (see figure 7.3). The embedding in the image is done during the transformation between the spatial and transform domain. We call this type of watermarking and distribution scheme **Modified-transform watermarking**, **MTW**.

7.4.1 Embedding and distribution

We will only use the I-frames for the embedding in this work. For MPEG encoding a frame F is split into blocks of 8×8 pixels F^j for $j = 1, \ldots, s$, with s

$$\begin{tikzcd} F \arrow[r, bend left, "T"] & G \arrow[l, bend left, "T^{-1}"] \end{tikzcd}$$

Figure 7.3: The embedding can be done in the frame F (spatial domain) or in the transformed frame G (transform domain). We will instead embed in the transform T.

being the number of blocks in one frame F. These blocks are then transformed using DCT. The DCT matrix T will then generate the transformed image block by

$$G^j = TF^jT^t \tag{7.4}$$

for all $j = 1, \dots, s$. The offset watermark V is split up into 8×8 blocks V^j. The individual watermarks W_i are also divided into blocks W_i^j.

The embedding of block j is now done by embedding the transform matrix in the encoder with

$$T_{V^j} = T + V^j \tag{7.5}$$

at the distributor. The transformed image block will then be

$$G_V^j = T_{V^j}F^jT_{V^j}^t. \tag{7.6}$$

which will be quantized $\hat{G}_V^j = Q(G_V^j)$ and source coded before transmission. The video stream is transmitted to all receivers.

In the decoder at the receiver i the transform matrix for block j is changed by

$$T_{V_i^j} = T + V_i^j \tag{7.7}$$

where

$$V_i^j = V^j + W_i^j \tag{7.8}$$

is the individual decoding key that receiver i has available. The individual watermark W_i^j is much smaller in energy than the offset watermark V^j. By using this decoding key the decoded image block will be

$$\hat{F}_i^j = T_{V_i^j}^{-1}\hat{G}_V^j(T_{V_i^j}^t)^{-1} \tag{7.9}$$

which both contains a quantization distortion and the embedded watermark. The embedded information contains not only W_i^j and V^j but also T and F_j. This can be seen if we assume that the quantization error is zero, $\hat{G}_V^j = G_V^j$. We will then have

$$F_i^j = (T + V^j + W_i^j)^{-1} G_V^j ((T + V^j + W_i^j)^t)^{-1} = \tag{7.10}$$

$$= (T + V^j + W_i^j)^{-1}(T + V^j)F^j(T + V^j)^t((T + V^j + W_i^j)^t)^{-1} =$$

$$= (I - (T + V^j + W_i^j)^{-1} W_j^j) F^j (I - (W_i^j)^t (T^t + (V^j)^t + (W_i^j)^t)^{-1}) =$$

$$= F^j - (T + V^j + W_i^j)^{-1} W_j^j F^j$$

$$-F^j (W_i^j)^t (T^t + (V^j)^t + (W_i^j)^t)^{-1}$$

$$+(T + V^j + W_i^j)^{-1} W_j^j F^j (W_i^j)^t (T^t + (V^j)^t + (W_i^j)^t)^{-1}$$

with I being the identity matrix of size 8×8. As we can see F_i^j will both contain F^j and the information in the three last terms that contains W_i^j, V^j, T and F^j. The individual embedded mark will be controlled by W_i^j but information from V^j, T and F^j are also embedded!

A receiver without any decoding key V_i^j will only be able to use an ordinary MPEG decoder. This will give him

$$\hat{F}_V^j = T^t \hat{G}_V^j T \approx T^t G_V^j T = \tag{7.11}$$

$$= T^t (T + V^j) F^j (T + V^j)^t T =$$

$$= (I + T^t V^j) F^j (I + (V^j)^t T) =$$

$$= F^j + T^t V^j F^j + F^j (V^j)^t T + T^t V^j F^j (V^j)^t T$$

since $T^t = T^{-1}$. The energy in the offset watermark V^j is much larger than the energy in the individual watermark W_i^j. Therefore $\hat{F}_V^j$ will be much more distorted than $\hat{F}_i^j$ as can be seen in (7.11) and (7.10). This visual distortion in $\hat{F}_V^j$ will make the video stream unusable for unauthorized receivers that decodes $\hat{G}_V^j$ without the decoding key.

7.4.2 Extraction

All I-frames that have been decoded with the individual key $\hat{F_i}$ will contain W_i and V as given by equation (7.9) and (7.10). So every I-frame in the video can be traced back to receiver i. Since the embedding is done during the transformation, W_i and V can not be directly extracted from a found frame F_{found}. A simple way to find a pirate is instead to correlate F_{found} with $\hat{F_i}$, $\forall i$. This method is therefore a private watermarking since we will need the original frame to create $\hat{F_i}$, $\forall i$ during the extraction.

To make the correlation faster we can start with correlating one or a few blocks and then continue the correlation only with the frames that have the best match. Finding better and faster extraction methods is left as a proposal for future work.

7.4.3 Robustness against attacks

If a pirate tries to use the test pattern attack to eliminate the individual watermark he will end up with F, $\hat{F_i}$ and $\hat{F_V}$. We see that W_i and V can not be directly calculated from (7.10) and (7.11). The best way to approach it would probably be to try to calculate a starting point and then continue by having some intelligent iterative algorithm search for V in (7.11). Given an approximation of V the same method can be used to find W_i in (7.10). It is not known if such an algorithm would be successful. If more than one of the original frames are known it will make the retrieval of V and W_i easier but it will still be difficult. The MTW can therefore be seen as robust against the test pattern attack. In section 7.6 we will suggest how to make the embedded information even harder to retrieve by a test pattern attack.

Another approach that collaborating pirates can use is to take their individual decoding keys V_i and combine these. This has some similarities with the research area Traitor tracing [56]. In Traitor tracing the keys are used to decrypt the media object. By combining their individual decryption keys the pirates can try to make a new valid decryption key that can not be traced back to any of them. In our case it is not the key but the attacked media object that is used to identify the pirates.

The decoding key V_i contains both the offset watermark and the individual watermark as given by (7.8). If a number of pirates take the mean of their decoding keys they will get

$$V_{mean} = \frac{1}{n}\sum_{i=1}^{n} V_i = V + \frac{1}{n}\sum_{i=1}^{n} W_i. \tag{7.12}$$

If the individual watermark has $E[W_i] = 0$ the W_i part in V_{mean} will be smaller the more pirates that collaborates.

That way the pirate copy will contain less distortion than the individually watermarked copies contain. The pirate copy will be closer to the original media object than the legal copies. This fact is not desired and might be a good argument for piracy - "*The pirate copies have better quality than the legal copies*". Therefore W_i must be constructed in such a fashion that

$$\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} W_{i_{k,l}} \neq 0 \tag{7.13}$$

for all elements k, l. One way to achieve this is that every element in W_i has a given sign. If the noise is uniformly distributed between 0 and A we will get

$$|\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} W_{i_{k,l}}| = \frac{A}{2} \tag{7.14}$$

for all elements k, l.

How the noise and embedding affects the frame (of size $N_1 \times N_2$) is illustrated in figure 7.4. If we see the images as points in an $N_1 \times N_2$ space we have five different points. Point 1 is the original image F. Using ordinary lossy coding we will get the compressed and distorted image in point 2. In this scheme we instead add the offset watermark during compression and that results in image $\hat{F}_V$, point 3. As we can see point 3 is far from point 1 and 2. If we only remove the offset watermark in the decoder we will get the image in point 4. Because of the offset watermark the coding has been changed and therefore point 4 contains more distortion than point 2. But the receivers can not reach point 4, instead they must use their individual decoding keys and will by using them get a frame $\hat{F}_i$ in the subset 5. As we can see the pirates can not reach point 4 by collaborating since all copies are located in subset 5.

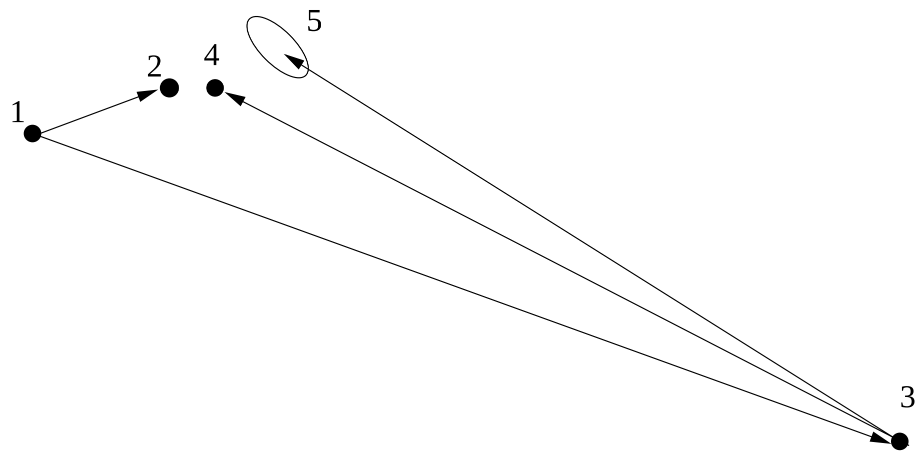

Figure 7.4: Images in $N_1 \times N_2$ space. Point 1 is the original image F, point 2 the coded image $\hat{F}$ and point 3 the image containing error noise $\hat{F}_V$. Point 4 comes from point 3 if the offset watermark is removed. The individually watermarked images $\hat{F}_i$ are located in the subset 5.

7.5 Experimental results

Since the watermarking is only done in the I-frame of the video we will use still images during our test. The coding method used is a simplified JPEG-coder. To further simplify the experiment we will use gray scale images.

7.5.1 Parameters

What we want to achieve is the following:

1. The method should give large distortion to the image if it is decoded without the individual key, i.e. F_V should be far from F.
2. The compression ratio should still be good during transmission.
3. The embedded individual watermark W_i should be robust against attacks.
4. The distortion to the image should be small.

One simple way to achieve the first two points is to let the offset watermark V strongly affect the DC level during transformation. This will give distinct visual changes and bring out block effects in F_V. At the same time the compression ratio will not be increased much since it is mostly a few element in every block that have been altered. So most of the quantized elements will not have been affected by the offset embedding.

By embedding in the upper row of the DCT matrix the DC element will be strongly affected during transformation. Other elements that will be affected are the upper row and left column in the transformed block matrix.

The third point will only be affected by the embedding of the individual watermark W_i. It should be embedded in such a way that elements that are difficult to remove are changed. Remember that W_i will not affect the compression ratio of the video stream but will affect the size of the individual decoding keys. If the keys are distributed the size of them will be of great concern. For large keys the needed bandwidth for the video distribution will increase and that is not a desired scenario.

The individual watermark will also affect point four together with the offset watermark as given by (7.10). The distortion from the encoding will not only depend on the compression ratio but also if the quantization and embedding of the offset watermark are adapted to each other. If the quantizer can not handle the embedded transform-blocks G_V^j the distortion from the encoding will be large.

Since we use a JPEG coder with only a simple quantization method we have not tried to optimize the parameters. We have only aimed at finding values that achieve the four points so that we can see if the method is useful.

For the offset watermark blocks V^j we use Gaussian noise. We have given the noise in the upper row a higher amplitude compared to the other rows. The upper row also contains an added constant that is different for each block. This constant will have a large effect on the DC level during transformation. The noise in the individual watermark blocks W_i^j is uniformly distributed as described in the last section.

7.5.2 Results

The number of individual marks in our experiment is 1000. We have used one single attack and three collaborating attacks to test our scheme. These attacks have been used:

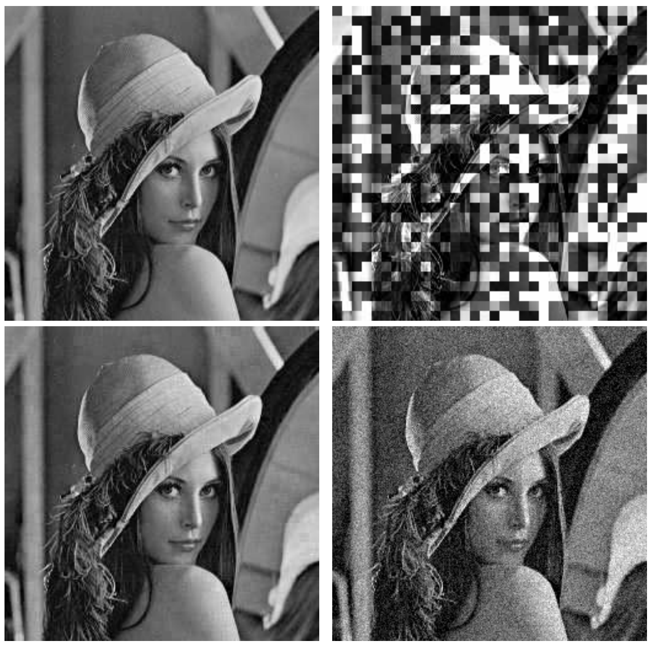

Figure 7.5: (a) Coded image in ordinary way without any embedding at the server or client. (b) Decoded image without key. (c) Decoded image with key containing individual watermark. (d) Attacked image (attack number 1).

1. **Noise** - Single pirate attack.
2. **Mean of images** - Collaborating between five pirates.
3. **Mean of decoding keys** - Collaborating between five pirates.
4. **Randomly picking pixels from copies** - Collaborating between five pirates.

We used the gray scale Lenna image of size 256 × 256 pixels as the original frame. In figure 7.5 (a) we can see what a frame will look like if it was encoded and decoded in an ordinary way (compression ratio 8.6, SNR 23.85 dB). Figure 7.5 (b) is the image after an unauthorized receiver has decoded the image without any decoding key F_V (SNR 3.76) and figure 7.5 (c) is the correctly decoded image F_i containing an individual watermark (compression ratio 8.3, SNR 23.08 dB). A pirate has taken this image and performed a noise attack (attack no. 1 by Gaussian noise with mean zero and standard deviation 20). The resulting image (SNR 14.31 dB) is shown in figure 7.5 (d).

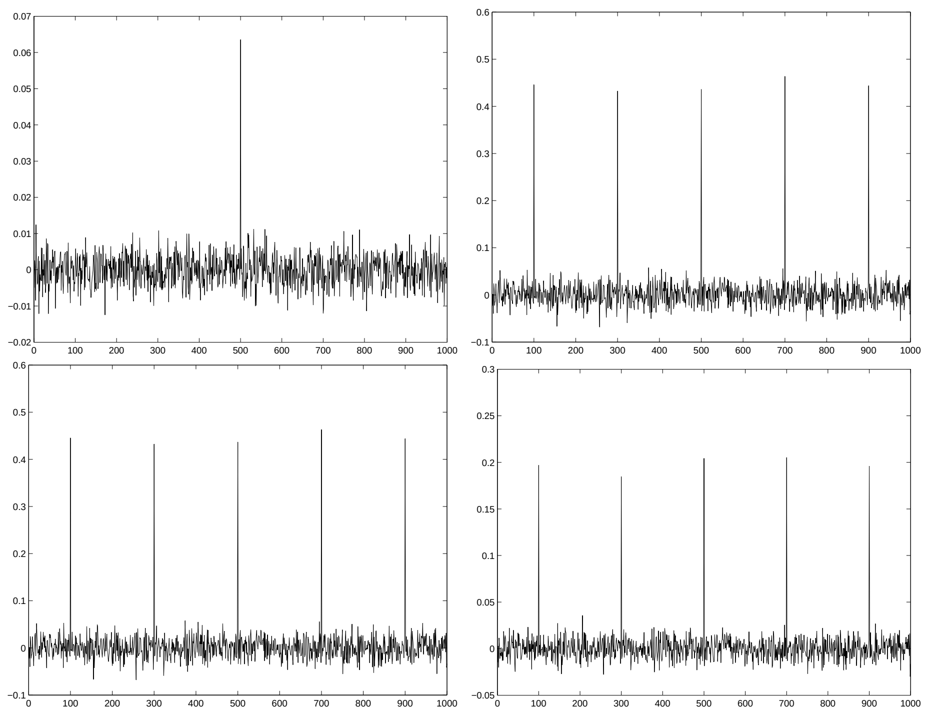

Figure 7.6: Correlation between the attacked image and the existing individual watermarked images. (a) Single pirate attack using noise. (b) Five collaborating pirates taking the mean of the images. (c) Five collaborating pirates taking the mean of their individual keys. (d) Five collaborating pirates randomly picking pixels from their five copies.

As we can see the compression ratio has only decreased with about 5% by the inclusion of the offset watermark during transformation in the encoder. This is an acceptable increase of bandwidth if it means that all receivers will get an individual key embedded into their copies.

The signal to noise ratio is changed from 23.85 dB to 23.08 dB because of the coding of the offset watermark and the individual watermark. We can test how much of that distortion that depends on the offset watermark and how much that depends on the individual watermark. If we only use the offset watermark during the decoding (point 4 in figure 7.4) the SNR will be 23.53 dB. So for this case the individual watermark will give more distortion than the coding of the offset watermark, but most of the distortion comes from the lossy image compression.

In figure 7.6 (a) we can see that the pirate (receiver number 500) can easily be identified during correlation between the attacked image and the different individual watermarked images. To get a proper system we should have a threshold that can be used for the decision if a receiver is the pirate or not. Since we are embedding in video, more than one I-frame can be used during the correlation to get a more exact identification of the pirates.

The correlation after attack number 2 is shown in figure 7.6 (b). The pirates have blended their copies by taking the mean value of every pixel. As we can see all five pirates are identified. For attack number 3 (figure 7.6 (c)) the pirates have not attacked the image directly, instead they have taken their individual keys and taken the mean of them. This new key has then been used during the decoding. Once more all five pirates can be identified and actually attacks number 2 and 3 give the same results. So even if the pirates can read their individual key this will not give them any advantages. Finally we have attack number 4 shown in figure 7.6 (d). In this attack the pirates have, for each pixel, decided at random which pirate to pick the pixel value from. Once more all of the pirates are identified.

The images from attack number 2, 3 and 4 will have a SNR similar to that of the different copies. So the noise distribution in the individual watermarks is of a nature that will not make collaboration a tool for getting better quality in the images.

In appendix A the MTW can be compared with the other methods for distributing individual watermarked copies.

7.6 Further improvements

In the described scheme we embed one individual watermark W_i in the client. We can also use

$$\hat{F}^j_{i,1,2} = (T + V^j + W^j_{i,1})^{-1} \hat{G}^j_V (T + V^j + W^j_{i,2}) \tag{7.15}$$

during the decoding. For that case we will need two unique decoding keys $V_{i,1} = V + W_{i,1}$ and $V_{i,2} = V + W_{i,2}$ for every receiver. The problem is that a pirate could exchange $V_{i,1}$ and $V_{i,2}$ during the decoding. That will still give a good image quality but the individual watermark will be removed.

#	Encoding with embedding of offset watermarks
1	$G^j_V = (T + V^j)F^j(T + V^j)^t$
2	$G^j_{V,1,2} = (T + V^j_1)F^j(T + V^j_2)^t$
3	$G^j_{V,1,2,3} = (T + V^j_1)F^j(T + V^j_2)^t + V^j_3$
4	$G^j_{V,1,2,3,4} = (T + V^j_1)(F^j + V^j_4)(T + V^j_2)^t + V^j_3$

Table 7.1: Embedding of one, two, three and four different offset watermarks in the encoder.

To embed more watermarks and that way increase the degrees of freedom does seem like a good way to make the method robust against the test image attack. The more matrix elements that are involved during the encoding and decoding the more difficult it will be to retrieve the embedded information for a pirate. One way to solve this is to embed more that one offset watermark during the encoding. In table 7.1 four different ways of embedding the offset marks are given. The first method embeds in the usual way. The second use different marks for the transformation of the rows and the columns. For the third method an embedding in the transform domain is included. Finally for the fourth method an embedding is also done in the spatial domain, giving a total of four embedded offset watermarks.

During the decoding we can embed as many individual watermarks as we have different offset watermarks. Table 7.2 gives four different ways of embedding the individual watermark depending on if we have one, two, three or four different decoding keys. We can also chose to embed fewer individual watermarks than offset watermarks. Then one or more of the decoding keys are not individual and can be used by all receivers.

Another improvement of the scheme is encryption. In the MTW method we use a type of weak encryption that only distorts the copies but unauthorized receivers can always see what the images describe. The distorted image does not have any commercial value but we have not distorted the audio. So an unauthorized receiver can still listen to the sound from the video stream. Therefore it is best to encrypt the data during transmission.

#	Decoding with embedding of individual watermarks
1	$\hat{F}_i^j = (T + V^j + W_i^j)^{-1}\hat{G}_V^j(T + V^j + W_i^j)$
2	$\hat{F}_{i,1,2}^j = (T + V_1^j + W_{i,1}^j)^{-1}\hat{G}_{V,1,2}^j(T + V_2^j + W_{i,2}^j)$
3	$\hat{F}_{i,1,2,3}^j = (T + V_1^j + W_{i,1}^j)^{-1}(\hat{G}_{V,1,2,3}^j - V_3^j - W_{i,3}^j)(T + V_2^j + W_{i,2}^j)$
4	$\hat{F}_{i,1,2,3,4}^j = (T + V_1^j + W_{i,1}^j)^{-1}(\hat{G}_{V,1,2,3,4}^j - V_3^j - W_{i,3}^j)(T + V_2^j + W_{i,2}^j) - -V_4^j - W_{i,4}^j$

Table 7.2: Embedding with one, two, three and four different decoding keys in the decoder.

7.7 Conclusions

In this chapter we have described two known methods for distributing individually watermarked video but only transmitting one video stream. We have also presented the test pattern attack that can erase the individual watermarks for these methods. Therefore we have constructed a new method that is more robust against the test pattern attack.

The new distribution and embedding method is independent of the network and will only give minor data expansion compared to distribution of non-watermarked video. The embedded watermarks are dense and only one I-frame is needed to identify the pirate. No secure hardware is needed and the watermarking is robust against collaborating pirates. We call the method Modified-transform watermarking since it is the transform that is modified and will give the individual watermarks at the receiver.

Some ideas on how to improve the Modified-transform watermarking have also been given.

Chapter 8

Conclusions

In this thesis we have dealt with watermarking of images and video. We have described different kinds of watermarking properties and applications. For copyright management individual watermarks are a strong tool. These watermarks can identify the pirates if illegally distributed copies are found.

In chapter 5 we have described how fractal block coding can be used to embed watermarks in images. The known embedding method has been extended to be more robust. We use channel capacity reasoning to increase the knowledge about if the watermark has been attacked. Channel capacity calculations can also be used to increase the amount of data in the watermark.

How to distribute individually watermarked media in a scalable way is of great interest. In chapter 6 we have given an extensive description of known methods for such types of distribution of watermarked media. We find that all of these methods have drawbacks that will make them difficult to use in practical situations.

In chapter 7 we have described two more individual watermarking methods that are better suited for video distribution than the ones given before. But these methods will have difficulties with attacks that use knowledge about the original frame of the video.

Finally we have constructed a new method for distributing individually watermarked copies called **Transform-modified watermarking**. This method does not rely on secure hardware. It is independent of the network, gives minor data expansion, has dense watermarks and is robust against collaborating attacks. This method will make it difficult to retrieve the watermarks even if the method is known and the original frames are used.

Appendix A

Table of distribution methods

In chapter 6 and 7 some methods for distributing individually watermarked media is described. The table on the next page summarizes the properties of these methods.

Method	**Individual watermarking location**	**Dependent on the network**	**Data expansion**	**Sparse or dense watermarks**	**Robust against collaboration**
Combined unicast and multicast	Server	No	Large	Dense	Fair
Watercasting	Network	Yes	Fair	Sparse	No
Hierarchical tagging	Network	Yes	No	Dense	-
WHIM Backbone	Network	Yes	No	Dense	Yes
WHIM Last hop	Network	Yes	No	Dense	Yes
Joints source fingerprinting	Client (Server)	No	Large	Dense	Yes
Packet based embedding	Client	No	Double	Sparse	Fair
Frame based embedding	Client	No	Double	Sparse	Fair
Bulk tagging	Client	No	-	Dense	No
COiN-Video	Client	No	Six times	Dense	No
Chameleon	Client	No	-	-	Fair
Mask blending	Client	No	Minor?	Dense	Yes
JFD	Client	No	-	Dense	Fair
MTW	Client	No	Minor	Dense	Yes

Table A.1: Classification of methods for distributing individually marked video.

Bibliography

[1] Hartung F. and Kutter M. Multimedia watermarking techniques. In *Proceedings IEEE: Special Issue on Identification and Protection of Multimedia Information*, volume 87(7), pages 1079–1107, July 1999.

[2] Berghel H. Watermarking cyberspace. *Communications of the ACM*, 40(11):19–24, 1997.

[3] Abel R. *The red rooster scare - Making cinema american, 1900-1910.* University of California Press, Berkeley and Los Angeles, California, USA, 1999.

[4] Hembrooke E. F. Identification of sound and like signals. United States Patent, 3,004,104, 1961.

[5] Cox I. J. and Miller M. L. Electronic watermarking: The first 50 years. In *IEEE Fourth Workshop on Multimedia Signal Processing*, pages 225–230, October 2001.

[6] Bloom J. A. *et al.* Copy protection for dvd video. *Proceedings of the IEEE*, 87(7):1267–1276, July 1999.

[7] Cox I. J., Miller M. L., and Bloom J. A. Watermarking applications and their properties. In *Proceedings of the International Conference on Information Technology: Coding and Computing - ITCC2000*, pages 6–10, March 2000.

[8] Kutter M. and Petitcolas F. A. P. A fair benchmark for image watermarking systems. In *Proceedings of the SPIE Security and Watermarking of Multimedia Contents*, volume 3657, pages 226–239, Januari 1999.

[9] Mohanty S. P., Ramakrishnan K. R., and Kankanhalli M. A dual watermarking technique for images. In *Proceedings of the 7th ACM International Multimedia Conference, ACM-MM'99*, pages 49–51, October 1999.

[10] Marc Spider from Digimarc, http://www.digimarc.com/.

[11] Eggers J.J., Su J.K., and Girod B. Asymmetric watermarking schemes. In *Tagungsband des GI Workshops "Sicherheit in Mediendaten"*, Berlin, Germany, September 2000.

[12] Fridrich J. A hybrid watermark for tamper detection in digital images. In *Proceedings of the Fifth International Symposium on Signal Processing and its Applications, ISSPA'99*, pages 301–304, Brisbane, Australia, August 1999.

[13] Kahn D. The history of steganography. In *Proceedings of the International Workshop on Information Hiding*, volume 1174 of *Lecture Notes in Computer Science*, pages 1–5, May 1996.

[14] Kalker T., Depovere G., Haitsma J., and Maes M. Video watermarking system for broadcast monitoring. In *Proceedings of the Electronic Imaging'99*, January 1999.

[15] C-C. Chang, K-F. Hwang, and M-S. Hwang. Robust authentication scheme for protecting copyrights of imagesand graphics. *IEE Proceedings-Vision, Image and Signal Processing*, 149(1):43–50, Februari 2002.

[16] Mintzer F. and Braudaway G. W. If one watermark is good, are more better? In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'99*, pages 2067–2069, 1999.

[17] Wagner N. R. Fingerprinting. In *Proceedings of the 1983 Symposium on Security and Privacy*, pages 18–22, April 1983.

[18] Blakley G. R., Meadows C., and Purdy G. B. Fingerprinting long forgiving messages. In *Proceedings of Crypto '95*, pages 180–189. Springer-Verlag, 1985.

[19] D. Boneh and J. Shaw. Collusion-secure fingerprinting for digital data. *IEEE Transactions on Information Theory*, 44(5):1897–1905, September 1998.

[20] Johnson N., Duric Z., and Jajodia S. On fingerprinting images for recognition. In *Proceedings of the Fifth International Workshop on Multimedia Information Systems, MIS'99*, Palm Springs, USA, October 1999.

[21] Memon N. and Wong P. W. A buyer-seller watermarking protocol. *IEEE Transactions on Image Processing*, 10(4):643–649, April 2001.

[22] Cover T. A. and Thomas J. A. *Elements of Information Theory.* Wiley series in telecommunication. Wiley-Interscience, 1991.

[23] Chu H. H., Qiao L., and Nahrstedt K. A secure multicast protocol with copyright protection. *ACM SIGCOMM Computer Communications Review*, 32(2):42–60, April 2002.

[24] Sayood K. *Introduction to Data Compression.* Morgan Kaufmann Publishers, second edition edition, 2000.

[25] Cox I., Kilian J., Leighton T., and Shamoon T. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, 6(12):1673–1687, December 1997.

[26] Hartung F. and Girob B. Digital watermarking in raw and compressed video. In *Proceedings SPIE 2952: Digital Compression for Video Communication*, pages 205–213, October 1996.

[27] Jacquin A. E. A novel fractal block-coding technique for digital images. In *Proceedings of the IEEE international Conference on Acoustics Speech and Signal Processing ICASSP'90*, volume 4, pages 2225–2228, 1990.

[28] Davern P. and Scott M. Fractal based image steganography. In *Proceedings of the International Workshop on Information Hiding*, volume 1174 of *Lecture Notes in Computer Science*, pages 279–294, May 1996.

[29] Puate J. and Jordan F. Using fractal compression scheme to embed a signature into an image. In *Proceedings of the SPIE Photonics East'96 Symposium*, pages 108–118, November 1996.

[30] Zhao Y., Ma Q., and Yuan B. Digital watermark based on lifs. In *Proceedings of the IEEE International Conference on Image Processing ICIP'00*, pages 1281–1284, 2000.

[31] Hanqiang C. and Guangxi Z. A watermark method based on fractal self-similarity. In *Proceedings of the IEEE International Conference on Image Processing ICIP'00*, volume 1, pages 99–102, 2000.

[32] Bas P., Chassery J.-M., and Davoine F. Self-similarity based image watermarking. In *Proceedings of the European Signal Processing Conference EUSIPCO'98*, volume 4, pages 2277–2280, September 1998.

[33] Wu H-C and Chang C-C. Hiding digital watermarks using fractal compression technique. *Fundamenta Informaticae*, 58(2):189–202, November 2003.

[34] Stenborg K-G. Signature embedding based on fractal block coding. In *Proceedings of the Fourth Conference on Computer Science and System Engineering CCSSE'02*, pages 61–66, Norrköping, Sweden, October 2002.

[35] Wozencraft J. M. and Jacobs I. M. *Principles of Communication Engineering*. Waveland Press, Inc., 1965/1990. Reissue.

[36] Rey C., Doërr G., Csurka G., and Dugelay J-L. Toward generic image dewatermarking? In *Proceedings of the IEEE International Conference on Image Processing, ICIP'02*, volume 3, pages 633–636, 2002.

[37] Digital Video Broadcasting, http://www.dvb.org.

[38] Stevens W. R. *TCP/IP Illustrated, Volume 1 - The Protocols*. Addison-Wesley, 1994.

[39] Perrig A., Canetti R., Tygar J. D., and Song D. Efficient authentication and signing of multicast streams over lossy channels. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, pages 56–73, 2000.

[40] Wu D. *et al.* Streaming video over the internet: Approaches and directions. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(3):1–20, March 2001.

[41] Wu T-. L. and Wu S. F. Selected encryption and watermarking of mpeg video. In *International Conference on Image Science, Systems, and Technology CISST'97*, June 1997.

[42] Zhao H. V. and Liu K. J. R. Fingerprint multicast in secure video streaming. *IEEE Transactions on Image Processing*, to appear Q4, 2005.

[43] Brown I., Perkins C., and Crowcroft J. Watercasting: Distributed watermarking of multicast media. In *Networked Group Communication NGC'99*, pages 286–300, November 1999.

[44] Caronni G. and Schuba C. Enabling hierarchical and bulk-distribution for watermarked content. In *Annual Computer Security Applications Conference ACSAC'01*, December 2001.

[45] Judge P. and Ammar M. WHIM: Watermarking multicast video with a hierachy of intermediaries. In *Proceedings of the International Workshop on Network and Operation System Support for Digital Audio and Video NOSSDAV'00*, June 2000.

[46] Luh W. and Kundur D. New paradigms for effective multicasting and fingerprinting of entertainment media. *IEEE Communications Magazine*, pages 77–84, June 2005.

[47] Parviainen R. and Parnes P. Large scale distributed watermarking of multicast media through encryption. In *International Federation for Information Processing Communications and Multimedia Security Joint working conference IFIP TC6 and TC11*, May 2001.

[48] Kutter M., Voloshynovskiy S., and Herrigel A. The watermark copy attack. In *Proceedings of SPIE: Electronic Imaging 2000, Security and Watermarking of Multimedia Content II*, volume 3971, January 2000.

[49] Baaziz N. and Sami Y. Attacks on collusion-secure fingerprinting for multicast video protocols. In *Proceedings of the First International Conference on Distributed Frameworks for Multimedia Applications (DFMA'05)*, pages 210–216, 2005.

[50] Konstantas D. and Thanos D. Commersial dissemination of video over open network: Issues and approaches. Technical report, Object Systems Group, Center Univeritaire d'Informatique of Univ. of Geneva, 2000.

[51] Anderson R. and Manifavas C. Chameleon - a new kind of stream cipher. In *Fourth Workshop on Fast Software Encryption FSE'97*, pages 107–113, January 1997.

[52] Luh W. and Kundur D. *Multimedia Security Handbook*, chapter 19: Digital Media Fingerprinting - Techniques and Trends, pages 577–603. CRC Press, 2005. Furht B. and Kirovski D., Eds.

[53] Briscoe B. and Fairman I. Nark: Receiver-based multicast non-repudiation and key management. In *Proceedings of the 1st ACM conference on Electronic commerce*, pages 22–30. ACM Press, 1999.

[54] Emmanuel S. and Kankanhalli M. S. A digital rights management scheme for broadcast video. *ACM-Springer Verlag Multimedia Systems Journal 8*, pages 444–458, 2003.

[55] Kundur D. and Karthik K. Video fingerprinting and encryption principles for digital rights management. *Proceedings of the IEEE*, 92(6):918–932, June 2004.

[56] Chor B., Fiat A., Naor M., and Pinkas B. Tracing traitors. *IEEE Transactions on Information Theory*, 46(3):893–910, May 2000.