Two-dimensional Coding and Detection for data storage on patterned media

H.W. de Jong

March 26, 2010

Graduation Committee:
Dr. Ir. J.P.J. Groenland
Dr. Ir. L. Abelmann
Prof. Dr. Ir. C.H. Slump
Prof. Dr. M.C. Elwenspoek
Dr. O. Zaboronski
Abstract

An output signal from the magnetic force microscope is used for the detection of bit values in patterned media. In a simulated signal build of a combination of lorentzpulses the pulse distance can be changed. The two-dimensional intersymbol interference will influence the ability of the detection of the bits. Conventional detection methods are adapted for the two-dimensional situation and a comparison of this methods is made by comparing bit error rates of the detectors in the presence of noise and jitter.
## Contents

1 Introduction 7

2 Conventional Detection and Coding 9
   2.1 Introduction .......................................................... 9
   2.2 One-dimensional techniques ........................................ 9
      2.2.1 Threshold Detection ........................................... 9
      2.2.2 Peak detection .................................................. 10
      2.2.3 Partial Response Maximum Likelihood ......................... 10
      2.2.4 Viterbi Detection .............................................. 11
   2.3 Two-dimensional techniques ....................................... 12
      2.3.1 M-Algorithm ..................................................... 12
      2.3.2 Cross Talk Cancellation ....................................... 12
      2.3.3 2D Viterbi Detection .......................................... 12
      2.3.4 Image Processing techniques .................................. 12
   2.4 Patterned Media Solutions ......................................... 13
      2.4.1 Iterative Decision Feedback Detection ....................... 13
      2.4.2 Modifying Viterbi Algorithm ................................ 13
   2.5 Error Correction Codes ............................................. 13
      2.5.1 Low Density Parity Check codes .............................. 14
      2.5.2 Run Length Limited codes .................................... 14
   2.6 Methods in this work ................................................ 14

3 Detection on patterned media 17
   3.1 Goal ................................................................. 17
   3.2 Properties of the MFM signal ...................................... 18
   3.3 Pulse description ................................................... 19
      3.3.1 One-dimensional ................................................. 19
      3.3.2 Two-dimensional ............................................... 19
   3.4 Simulation parameters ............................................... 21
      3.4.1 Sample rate ...................................................... 21
CONTENTS

3.4.2 Pulse period ................................................. 21
3.4.3 Pulse alignment ........................................... 21
3.4.4 Non-linearities of the read-out (perspective) .......... 23
3.4.5 Borders ....................................................... 24
3.4.6 Noise ......................................................... 24
3.5 Error rate of the detection .................................. 25
3.6 Clocking problem ............................................. 25

4 Detector description ........................................... 27
   4.1 Threshold detector ......................................... 27
   4.2 Peak detector ............................................... 28
   4.3 Triple detector .............................................. 29
   4.4 Decision Feedback Equalization ......................... 32
   4.5 Detectors on patterned media .............................. 35

5 Image Processing Techniques .................................. 37
   5.1 Spot Detection ................................................ 37
   5.2 Feature extraction .......................................... 38
      5.2.1 Point spread function ................................. 38
      5.2.2 Euclidean Distance .................................... 38
      5.2.3 Convolution ............................................. 40
      5.2.4 Laplacian of the Gaussian ............................ 43
   5.3 Classification ............................................... 44
      5.3.1 Non-local maximum suppression ....................... 44
   5.4 Image processing on patterned media .................... 45

6 Results and Discussion .......................................... 47
   6.1 Patterned media simulations ................................ 47
      6.1.1 Jitter Influence ....................................... 48
      6.1.2 Medium Noise Influence ............................... 49
      6.1.3 Pulse distance ......................................... 50
      6.1.4 Lower sample rate ..................................... 50
      6.1.5 Simple coding .......................................... 52
   6.2 Triple detector .............................................. 52

7 Conclusions and Recommendations .............................. 57
   7.1 Conclusions .................................................. 57
      7.1.1 Detector performance .................................. 57
      7.1.2 Detector choice ......................................... 57
      7.1.3 Coding .................................................. 58
      7.1.4 Computation Power ..................................... 58
   7.2 Recommendations ............................................. 58
      7.2.1 Worst-case patterns .................................... 58
      7.2.2 Computation Power ..................................... 59
Chapter 1

Introduction

In the current information era the demand for more and more storage capacity is constantly growing. The capacity of harddrives is still increasing, but will reach a physical limit. Magnetic pulses are written close to each other and will soon reach a limit where they can’t be distinguished.

In the regular storage media the information storage is performed in one dimension. Tracks on a harddrive are written without awareness of surrounding tracks. Only intersymbol interference is taken into account and intertrack interference is being neglected.

Within the group of TST-SMI exploratory bit patterned storage media are under research in order to have a two-dimensional read and write method. In this two-dimensional storage medium the bits are written in square patterns on a magnetic surface. The magnetic field of each bit written on the medium is influenced by the fields of the surrounding bits. Two dimensional intersymbol interference is present in these fields.

By using the knowledge of the influence of fields onto each other a detection mechanism can be designed. An optimum should exist where a bit density as high as possible could be reached on an acceptable error rate in detection. In this work a bit patterned medium is simulated. Detectors derived from one-dimensional detection mechanisms have been applied as well as image processing techniques for object recognition. In the simulations the bit error rate is measured by varying medium noise and jitter.
2.1 Introduction

In conventional storage media a one dimensional signal is used in detection. Different detection techniques can be used for detecting bit values in a signal with influence from medium noise and jitter. A general overview of different techniques is given in this chapter.

In the storage on 2D patterned media, these techniques might be useful in the further research for optimal detection and coding.

2.2 One-dimensional techniques

2.2.1 Threshold Detection

In many detection algorithms the decisions are based on a threshold. For example, having a signal with two possible values: a 0 and a 1, the threshold can be placed at 0.5. All signal values above 0.5 are determined as a 1, all signals below as a 0.

This way of detection is very easy to implement, but errors can easily occur while reading from the media. Different noise sources will influence the signal and also influence the decision.
2.2.2 Peak detection

A peak detector will make decisions based on the occurrence of local peaks in the signal. A regular method is differentiating the signal, which results in zero crossings on the peaks. A rectifier decides if the signal is above a pre-defined value. When both occur, a peak is detected. An example can be found in figure 2.1.

![Diagram of peak detection channel](image)

Figure 2.1: Example of peak detection on one-dimensional signal, from [2]

2.2.3 Partial Response Maximum Likelihood

Partial Response Maximum Likelihood (PRML) [1] is used in many data storage media. Compared to the threshold detector, PRML uses the shape of the signal, being aware of linear ISI in the signal. As can be seen from the name, a PRML system consists of two parts, a partial response system combined with a maximum likelihood detector.

The influence of ISI will change the shape of the signal. Peaks will be lower because of the transitions before and after the current signal. The PRML system now can successful detect a bit pattern in a weak signal. The detector uses the relative differences in the signal, which are amplified, making it possible to distinguish differences in the signal.

That way it is possible to detect a bit pattern in a weak signal description by looking at a relative difference between the signal. A simple threshold detector would use absolute values and place all bit values in the same category, whereas they have distinguishable differences.
Partial Response

The basic idea of Partial Response (PR) is to use a controllable amount of ISI rather than eliminate it. The analog signal from the media should have a constant magnitude, which can be reached by a Variable Gain Amplifier (VGA). Because of the bandwidth in which the PR works, all components beyond this bandwidth have to be filtered which is done with an analog amplifier.

Maximum Likelihood

The maximum likelihood (ML) part of the detector does not make immediately decisions on the stream of samples that is put into it. The ML detector analyzes the data and then chooses the most probable sequence.

The ML-detector calculates the distance between (a part of) the current stream of samples and several streams of samples that are known and possible. This results in a set of distances, from which the smallest, the most likely, can be chosen.

An example of the steps in the PRML is given in figure 2.2.

![Figure 2.2: Example of PRML, from [2]](image)

2.2.4 Viterbi Detection

The Viterbi Algorithm [6] is a kind of likelihood method that calculates the most likely path based on the euclidean distances of the different possible paths. Every combination of output bits has a likelihood which is used by the Viterbi Algorithm. One of the bits in the output might have a big disturbances due to some noise, so give a different output value. When it is unlikely for the output to have that value, based on the other bits in the combination, it will be corrected. When paths become longer, the Viterbi Algorithm needs a lot of computation, this makes the Viterbi Algorithm less suitable for such paths.
2.3 Two-dimensional techniques

2.3.1 M-Algorithm
The paper from Zadeh [3] describes a solution for a multitrack magnetic recording medium. In this description the input data is given as a two dimensional signal. In the channel model it is assumed there is only interference from adjacent tracks.

The several standard methods as Partial Response and Viterbi detectors are described, however, the problem with such detectors is the high number of states for only a small number tracks to be considered at one readout. The M-Algorithm is a detector with a reduced complexity. From all paths the $M$ best paths are stored. An advantage of the $M$-algorithm over the Viterbi Algorithm is the reduced number of paths, which makes it more suitable for longer paths and decreases the computations.

In Tosi [4] also a description for a system using the $M$-algorithm is given. This paper describes a partial response detection for a multi-track system.

2.3.2 Cross Talk Cancellation
In the thesis of Immink [5] the Cross Talk Cancellation (XTC) is discussed. With this technique three tracks are read out simultaneously and filtered versions of the two outer tracks are subtracted from the middle track in order to cancel the influence from the two outer tracks on the middle track. A quite simple method for improving the detection.

2.3.3 2D Viterbi Detection
The paper of Kato [7] to which is referred in Immink [5] has some interesting points about 2D Viterbi detection, but these are just separated Viterbi detectors. The 2D viterbi detection described in Immink made some improvements on the Multitrack Viterbi Algorithm it is referring to. These refinements include taking into account weighing of contributions and taking into account other contributions.

Also non-linear ISI can be handled, by using the output of a previous calculated value of a state. The complexity of the algorithm has also been decreased.

2.3.4 Image Processing techniques
By plotting the output signal of a patterned medium as an image, this readout signals look like images with bright and dark dots on a gray background. With image processing techniques the dots in these signals can be detected.
In chapter 5 more about these methods is given.

2.4 Patterned Media Solutions

Several papers are published about the two-dimensional detection in patterned media storage (PMS). An overview is given below.

2.4.1 Iterative Decision Feedback Detection

In Keskinoz [8] a description for a 2D detector for a patterned media is worked out in detail. The paper has 2 parts, one for an Iterative Decision Feedback Detection (IDFD) system and a 2D Generalized Partial Response (2D-GPR) with 1D Viterbi.

These techniques require more computations and have some requirements. IDFD requires all readings to be available to detect the information symbols. The 2D-GPR method performs better than IDFD under the same computational load, whereas IDFD could achieve a higher SNR, employing more iterations.

2.4.2 Modifying Viterbi Algorithm

Nabavi [9] describes a modified version of the Viterbi Algorithm for bit-patterned media. This algorithm improves the BER while the complexity is not significantly increased. There are the same number of states, but the number of branches between these states increases.

The track misregistration (TMR) or read head offset is also taken into account, where as expected the modified Viterbi is more tolerant than the normal Viterbi.

2.5 Error Correction Codes

Errors will always occur in writing and reading data, due to noise and interference. To be able to correct errors, the user data is encoded with error correction codes (ECC).

Errors can occur in two types: single-bit errors and bursts of errors. A single-bit error can occur through a short noise event, which results in an extra pulse or a missing pulse. Bursts of errors usually occur through defects of the medium.

The use of ECC improves the reliability of data recovery. There are different ECC designed to correct a finite number of corrupted bits. Of course encoding requires more bits for the information to be stored on the disc, but the storage
capacity of a medium can be improved with the ECC.

### 2.5.1 Low Density Parity Check codes

To improve the performance of the system, it is important to use a code which is optimized for the channel properties. Combinations of bits which are hard to detect have to be avoided in order to keep the error rate as low as possible.

The use of the right codes already inserts some redundancy in the bits stored on the media. In [10] several methods for the application of Low-Density Parity-Check (LDPC) codes are given, as well as the construction of optimal codes for different channel models.

A consequence on the use of complex coding is the complexity of the decoding that has to be done after the detection of the signal.

### 2.5.2 Run Length Limited codes

Run Length Limited (RLL) codes are codes which have a minimum and a maximum number of a value used after each other. For magnetic storage these values are a transition or no transition, a 1 or a 0. RLL Codes are typically referenced as \((m/n)(d,k)\) codes. \((m/n)\) Means: \(m\) user bits are mapped on \(n\) encoded bits, where \(n \geq m\). \(d\) is the minimum allowed number of consecutive ”0”s between two ”1”s \((d \geq 0)\). And \(k\) is the maximum number of consecutive ”0”s between two ”1”s \((k \geq 0)\).

In the paper from Kato [7] a multi-track recording system with the use of 2D-PRML and 2D-RLL is described.

### 2.6 Methods in this work

While the storage medium in this work is in an exploratory status, the detection is started with a threshold detection on the two dimensional signal. Also two-dimensional versions of the peak detector and a decision feedback detector are applied.

A method using 3x3 samples per bit, which is designed within the TST-SMI group is implemented and some image processing techniques are applied on the signal and a comparison with the regular methods is made.

The results of the detectors are measured by the error rate of the detected bits. By varying the medium noise and the jitter on the medium, a comparison between the methods is made.

In the first comparison between the detectors no specific coding methods on
Chapter 2. Conventional Detection and Coding

the input bits are applied. Only a simple coding method designed in the
TST-SMI group is used for comparison with an uncoded situation. In this
coding the worst-case patterns with bits of the same value are prevented
from being present in the input signal. In every 3x3 bitpattern always one
bit has value 1 and one bit is a 0.

The current implementation of this coding decreases the bit density by about
20 percent, by filling the 3x3 bitpatterns with a 1 and 0 without taking the
other bit values into account. A more advanced implementation could reduce
this density decrease.
Detection on patterned media

Because of the exploratory status of this storage medium candidate, many design decisions have not been made yet. The readout of the signal might be influenced by changes in parts of the system. In order to perform the simulations, assumptions have been made which are described in this chapter.

3.1 Goal

The goal of this master assignment is to compare detection mechanisms on the aspects of medium noise and jitter. On the storage medium the bits are stored in a two-dimensional bit pattern. On conventional storage media the bits have only influence from the previous and next bit on the same track, the intersymbol interference (ISI). In a two-dimensional medium the bit patterns would have influence from the surrounding bits in two dimensions, so called 2D-ISI. Which is in fact an equivalent to Intertrack Interference (ITI) in conventional media.

Other important aspects in comparing detectors is the clocking problem. A clock-less detector, where bits are detected autonomously would be very practical. To be concurring with conventional storage media the information should be detected on a high speed, with a sample rate as low as possible and a high bit-density on the medium.

In the two-dimensional storage and detection field more research has also been done for example by Wood et. al. in [14]. In that paper the feasibility of a magnetic recording at 10 terabits per square inch is investigated. A conventional medium is used, but more advanced write and read methods
are necessary in order to reach a higher bit density. Every bit is stored on only 2 grains on the medium. The information across different tracks is used in the signal processing. The paper concludes that a fine grained medium is necessary in order to be able to reach the high density for 10 terabits per square inch and a practical two-dimensional detection scheme should be developed for detection of the bits with accurate timing and positioning.

In the next chapters various types of detection are described, including the conventional one-dimensional methods. Also techniques from the image processing field are taken into account, the signal can be seen as a picture with black and white dots on it. This can be described as an object recognition problem in image processing.

### 3.2 Properties of the MFM signal

The simulated signal is based on the experimental recording setup with a magnetic force microscope (MFM) from the TST-SMI group. It is in detail described in [12] and [13]. A CoNi/Pt multilayer is used as the recording medium and the periodicity of pulses goes down to 150 nm. The field is scanned over the $x$ and $y$ direction. Writing the information is achieved by approaching a MFM-tip to the medium in the presence of an external magnetic field in the $z$ direction. The read signal is a result from the vibrating MFM tip in the stray field of the magnetized dots. The signal is proportional to the second $z$ derivative $V_{MFM}(x) \approx (d^2 H_z(x)/dz^2)$ integrated over the tip volume.

The MFM signal can be simulated by a superposition of the individual MFM responses. When the individual pulses are written close to each other, the responses of these pulses overlap, influencing the output signal. This is the (two-dimensional) intersymbol interference which influences detection.

An example of a typical MFM output signal can be found in figure 3.1. It should be mentioned that while this image looks like a surface with dots on it, this is not how the recording medium surface looks like, but a result from the vibrating tip.

The magnetostatic details are not described further here. In the simulations the signal is build from overlapping pulses, which characteristics are described further.
3.3 Pulse description

3.3.1 One-dimensional

Rather than taking the full magnetostatic details into account, we can in first instance approximate the signal by a superposition of Lorentz pulses. This considerably speeds up calculation time. The Lorentz function is defined as $y = \frac{1}{1+x^2}$. ISI is introduced by an overshoot of the pulse, also described by a Lorentz pulse. The resulting pulse description for one bit is a combination of three Lorentz pulses:

$$\text{pulse}(x) = \sum_{i=1}^{3} \frac{v_i}{1 + \left(\frac{x-x_i}{p_{w,50}/2}\right)}$$  (3.1)

Where
- $v_i$ = Amplitude of a Lorentz pulse
- $p_{w,50}$ = Lorentz pulse width, measured at 50% height of $v_i$
- $x_i$ = Origin of the $i$th Lorentz.

A single pulse used in the simulations can be seen in figure 3.2.

3.3.2 Two-dimensional

The two-dimensional situation is just an expansion of the one-dimensional pulse from equation 3.1. The pulse is now described in two dimensions:

$$R = \sqrt{X^2 + Y^2}$$

$$\text{pulse}(R) = \sum_{i=1}^{3} \frac{v_i}{1 + \left(\frac{R-R_i}{p_{w,50}/2}\right)}$$  (3.2)
$v_i = \text{Amplitude of a Lorentz pulse}$

$pw_{50_i} = \text{Lorentz pulse width, measured at 50\% height of } v_i$

$R_i = \text{Origin of the } \text{ith Lorentz.}$

In figure 3.3 an impression of this pulse description can be found.

In this pulse the overshoot around the pulse center can be seen, in figure 3.3(a) this overshoot is slightly darker than the surrounding. In the spatial view of figure 3.3(b) this overshoot can be seen as negative values.

This pulse description is almost always valid for the specific types of recording media. In specific cases the overshoot in the pulse description will change, influencing also the 2D-ISI. In this work only simulations for the given pulse description are performed.
3.4 Simulation parameters

3.4.1 Sample rate
The signal which will be used in the detectors is a sampled version of the MFM response of the magnetic field. With a high sample rate there is a better description of the pulses on the medium, more samples are available per bit.

In simulations the sample rate is varied. From earlier work within the TST-SMI Group a sample rate of 3x3 samples per bit has been used in detection. The threshold detector uses only one sample per bit and for the remaining detectors also higher sample rates have been used for comparison of the detector performance on varying sample rates.

With higher sample rates more computation power is needed for the detection. Whether or not this is acceptable depends on the achieved gain in error rate.

3.4.2 Pulse period
The bits written on the medium are placed at some distance from each other, the pulse period. The description of the pulse is given in section 3.3. While varying the pulse period, also the influence from one pulse on the neighbouring pulses will vary. Specific patterns can have such an influence on surrounding pulses that bit values are hard to detect.

A small pulse period will result in a high bit density on the medium, while more errors will occur in detection. With a slightly larger distance between pulses the 2D-ISI will change. The error rate can be reduced and compensate for the loss in bit density. An optimum therefore will exist.

In figure 3.4 examples of pulse distances can be seen for a one dimensional signal. For a two dimensional signal the same principle holds for both dimensions. These figures show a signal with two pulses of the same value. In case of two opposite values, the pulses will also influence, but increase the bit value of the neighbouring value, as is given in figure 3.5.

3.4.3 Pulse alignment
The written pulses can be organized in different patterns. Two main patterns can be distinguished: a square pattern and a hexagonal pattern. An important difference between the square and hexagonal ordering is the equal distance between all direct neighbouring pulses in the hexagonal signal, so that these influences are also equal. In the square pattern the pulses will have different influences on each other. At this moment only the square type of ordering is used in the MFM and in this work only the square pattern is used in the detector.
Chapter 3. Detection on patterned media

(a) Big distance between pulses  
(b) Small distance between pulses  
(c) Combined signal of two pulses of fig 3.4(b)  

Figure 3.4: Example of different pulse distances.

(a) Two opposite bit values  
(b) Combined signal of bit values  

Figure 3.5: Pulses with opposite values.
In figure 3.6 examples of the square and hexagonal pulse alignment can be found. With the use of a hexagonal alignment the distance between the rows is smaller, to keep the distance to the pulses in the neighbouring rows the same as the pulses on the same row.

![Square and Hexagonal Pulse Alignment](image)

**Figure 3.6: Square and hexagonal pulse alignment**

### 3.4.4 Non-linearities of the read-out (perspective)

A typical output of the MFM, as can be seen in figure 3.1, shows a signal with pulses which are not equally spaced. For the detection now a localization problem arises. The pulses are not distributed homogeneous on the field. Solutions for this localization problems are a detector that finds the pulses autonomously. Without the knowledge of the position of the pulses the detector will find these pulses and the bit values. Another option is a transformation of the signal before detection is applied. In figure 3.1 the signal looks like a field with a perspective transformation. With a description of these non-linearities of the output signal a geometric transformation can be applied resolving the localization problem.

An example of a transformation can be found in figure 3.7. In figure 3.7(a) the original MFM output signal can be found. In figure 3.7(b) a rotation of 4 degrees is applied to the signal and in figure 3.7(c) a perspective transformation. The resulting image is a more homogeneous distribution of the pulses over the total field. During this transformations incomplete rows and columns are removed by cropping the image, this is indicated by the dashed lines in the figures.

The signal looks like a hexagonal pattern, but is in fact a 45 degrees rotated square pattern.
3.4.5 Borders

In the typical output signal of the MFM, given in figure 3.1, pulses are situated on the borders of the image. It is hard to detect these pulses, especially when transformation or filtering is applied on the signal. In the simulated signal this problem is solved by keeping the pulses a half pulse period from the border. Applying transformations does not result in disturbing the pulses on the borders.

A number of pulses situated on the border is also not taken into account in the calculation of the bit error rate (BER). The pulses on the borders are not surrounded by pulses on all sides and don’t have intersymbol interference from all sides. The detection for the bit values of these pulses usually will be easier compared to the pulses in the rest of the signal. The overhead of this border pulses will be two bits on all sides. With the current pulse description ISI and ITI will not have influence on more than two concurring neighbours.

3.4.6 Noise

Noise appears in different ways and can be divided in data dependent and data independent noise. The data dependent noise is in this work the ISI and ITI. With the use of an adapted detector or coding scheme this is noise which can be dealt with. The data independent noise can be divided into two types: Medium noise and Jitter.

Medium noise

Medium noise is a result from the inequalities in the magnetic field caused by differences in magnetic moment or tip/medium distance. In the simulations medium noise is modelled by adding white gaussian noise to the signal.
every simulation this noise is varied to create a signal-to-noise ratio (SNR)/
bit error rate (BER) plot. A signal with a low SNR is given in figure 3.8(b).
By looking at images of the signal dots are still detectable by the human eye,
where for a detector the sample values might be below the given threshold
value.

Jitter

Small misalignments in writing and reading the signal is called jitter. Dis-
placement of the peaks of the pulses influences the detection. Jitter can be
-described as a noise on the position of each pulse. A detector should be able
to detect the pulses while there is some jitter in the signal. An example of a
signal disturbed by jitter can be found in figure 3.8(c). The pulses are not
placed on the square raster, so the intersymbol interference will vary for all
pulses.
In the simulations the jitter level is varied to compare the influence of it on
the bit error rate of the detectors.

![Images of input signals](image.png)

Figure 3.8: Examples of input signals used for detection

### 3.5 Error rate of the detection

In a 2D data storage system the goal is to reach a bit density as high as
possible with a low error rate. We wish to store data at the highest possible
density, without causing raw error rates above $10^{-4}$. A BER of $10^{-4}$ is chosen
because error correction codes which will be applied after the detection are
able to correct up to raw error rates of $10^{-4}$.

### 3.6 Clocking problem

In the description of the detectors the position of the pulses in the signal is
assumed to be known. The detection is applied around this specific position.
In the typical MFM signal shown before, the position of the pulses is not
on a specific raster due to rotation and a geometric distortion. With a
detector that finds the pulses itself the knowledge of this information won’t be necessary and the problem with rotation and distortion doesn’t need to be solved before the detection is applied.

The detectors have been implemented with some boundary conditions. As input to the system a signal is given with a known number of written bits. The detectors assume the bits to be distributed homogeneous, which makes it possible to divide the signal in regions for every bit.

One of the problems in this autonomous detection is the noise influence on the signal. Extra pulses might occur or pulses become below threshold. The influence of jitter is a problem when the peaks are written close to each other, they might get written so close near each other that they won’t be distinguishable and will appear as just one pulse. This makes detection very difficult.

As a result from the 2D-ISI extra pulses might occur in the signal. A mechanism where a minimal distance is required between the samples of the detected bits can reduce the number of errors resulting from this influence.

The distortions visible in the output signal of the MFM influence the linear distribution of the pulses over the signal. The detection could compensate this by evaluating each detected pulse and adapt the expected pulse for the last few detected pulses. When the pulse period becomes wider this is compensated during detection and does not lead to an error. Such an error detection mechanism would cost extra computation power. Another possibility to compensate for the non-linearities is to apply transformation techniques. If the non-linearities of the MFM signal can be described this can be used for transformation.

In simulations these distortions are not taken into account. The pulses are expected to be distributed homogeneously on the medium.
Detector description

This chapter describes the conventional detectors used for the simulations in this work. In appendix A the code for all detectors can be found. In the detectors the expected position of the pulses is assumed to be on a square pattern.

4.1 Threshold detector

The threshold detector checks whether the expected peakpoint of the field is above a certain threshold. An important parameter in this detector is the threshold value. With a high threshold the number of false positives due to noise is minimized. A low threshold value will give more influence from the noise. The threshold taken in the simulations is zero, exactly the middle between the positive and negative pulse values.

![Threshold detector raster on three different signals.](image)

(a) Input signal without jitter and noise  (b) Input signal with jitter  (c) Input signal with 5x5 bits of value 1

Figure 4.1: Threshold detector raster on three different signals.

In figure 4.1 examples of the threshold detector on a simulated signal can
be found. The sample values are the crossings of the dashed raster lines. In this figure errors occur, as a result from the intersymbol and intertrack interference.

This detector takes the value of the signal on fixed sample points, resulting in a lot of influence from jitter on it. This shift of peaks will result in lower sample values at the expected points. With high jitter, a sample value can be below threshold and the bit value will flip. In figure 4.1(a) and 4.1(b) an input signal without and with jitter can be seen, both with the same bit pattern. In figure 4.1(b) a lot of pulses are not on the expected peak points.

In the 5x5 bitpattern in figure 4.1(c) all bits have value one and the interference has such an influence on the bits in the center, that the values on the threshold detector will flip. The crossings of the raster lines in the figure are the sample points of the threshold detector.

### 4.2 Peak detector

The peak detector searches for a maximum in the expected peak window. A scheme of the steps in the peak detector can be found in figure 4.2.

By finding this maximum less influence from jitter is expected. Intersymbol interference and intertrack interference is the same as with the threshold detector. The interference can have such an influence that a positive peak will be flipped to a negative peak.

![Schematic view of peak detector](image)

Figure 4.2: Schematic view of peak detector

In figure 4.3 and 4.5 the outputs of the steps of the peak detector applied on an MFM signal can be found. In figure 4.3 a large pulse period is used, figure 4.5 shows a smaller pulse period. The rectifier is applied, which transforms the signal to three values: -1, 0 and 1. The rectifier decides if a peak is high enough to be defined as a bit pulse or that it is a peak of the noisy background. The decision of the rectifier is based on a threshold value. The signal from the rectifier is given in figure 4.3(b).

The input signal is also differentiated. In this differentiated signal, given in figure 4.3(c), zero crossings occur on the tops of the peaks. In case
Chapter 4. Detector description

of this two dimensional signal, the signal should be differentiated in both $x$ and $y$ direction. If both differentiations have a zeros crossing, a local maximum is detected. The combination of the rectified and differentiated signals completes the detection. A zero crossing of the differentiator indicates a maximum, or minimum, and a rectified signal of $-1$ or $1$ defines if the peak is high enough to be assigned as a written pulse.

Instead of using the differentiations in both directions also the gradient can be used. The gradient of a two-dimensional signal is defined as:

$$\nabla f(x, y) = \left[ \frac{\partial f(x, y)}{\partial x} \frac{\partial f(x, y)}{\partial y} \right] = \begin{bmatrix} f_x(x, y) \\ f_y(x, y) \end{bmatrix}$$

For each position the gradient points to the direction of the steepest ascent. And the magnitude $\|\nabla f(x, y)\|$ is proportional to the steepness. In figure 4.4 the differentiations in both directions can be seen as well as the magnitude.

Figure 4.3: Signals of the steps of the peak detector with big pulse period

Figure 4.4: Differentiations in $X$ and $Y$ direction and magnitude

4.3 Triple detector

The triple detector is a detector designed in the TST-SMI group. The triple-detector is a detector based on a sample rate of 3 samples per pulse. In the
Chapter 4. Detector description

2D situation 3 samples for every dimension are taken, resulting in a signal with 3x3 samples per pulse. These samples are not depending on a clock.

In the one-dimensional situation a signal as in figure 4.6 is used. The signal is a sampled version of the continuous function in figure 4.6(a). For every pulse three samples are taken. Samples are indicated by the dashed lines and numbered $s_1 \ldots s_9$. For the first peak the samples $s_1, s_2$ and $s_3$ are compared. Sample $s_2$ has the largest absolute value, indicated by a circle. Assuming that the peak occurs in the center of three values, the next peak should occur in the samples $s_4, s_5$ and $s_6$. For figure 4.6(a) the maximum is at sample $s_5$ and detection can continue for the last pulse. This will be detected on sample $s_8$.

In figure 4.6(b) jitter occurs on the signal. The second pulse is shifted. The maximum absolute value of the second pulse is no longer at sample $s_5$, but at sample $s_4$. For detection of the last pulse, the samples $s_6, s_7$ and $s_8$ are taken into account. Sample $s_6$ is hardly visible, the value is just on the x-axis. From these three samples $s_8$ is the sample with the maximum absolute value.

If the signal from figure 4.6(b) consists of more pulses, the next samples for detection would be $s_{10}, s_{11}$ and $s_{12}$. The pulse on sample $s_8$ is assumed to be in the center of three samples, so sample $s_9$ is ignored.

Since three samples per bit are taken, the triple detector is theoretically insensitive to jitter. When the jitter misaligns the bits, as shown in figure 4.6(b), the triple detector will detect the maximum of that specific field in one of the other three samples.

This method can also be applied on a two-dimensional signal. Now the absolute maximum is found in two dimensions. The 2D signal has samples in $x$ and $y$ direction. Starting in the upper left corner, the first 3x3 samples
to be examined are $s(1, 1)$ till $s(3, 3)$. From these 3x3 samples the absolute maximum is taken as the bit value.

The next 3x3 samples to be examined are based on the index of the detected maximum on the previous row and column, like in the 1D situation. In both the first row for the $y$-samples always 1 till 3 are used, in the first column always the $x$-samples 1 till 3 are used. The detector is assumed to know the number of rows and columns of the bit pattern.

For this two-dimensional situation a 5x5 bitpattern is given in figure 4.7. Circles around the samples indicate a detected maximum. In figure 4.7(a) a jitterless signal is given and the peaks are detected on the raster. In figure 4.7(b) a signal highly influenced by jitter is given, the detected maxima are indicated by the circles.

The performance of the triple detector depends a lot on the initial settings of the detector. A change in resolution or distance between peaks without adapting the sample rate will influence the performance. The startpoint of the signal should be defined clearly, having the first signal peak within the first three samples. If the first value is detected wrong this might result in an error burst, causing a high bit error rate due to an error propagation. The detected bits might be right, but shifted in one or more places.

The detector can be adapted to use more or less samples per bit. The error rate is expected to be lower due to the higher number of samples in which the chance of sampling exactly on the pulse peak is higher. In the simulations this higher sample rate is not examined, while only the concept of this method is compared with other detectors.
4.4 Decision Feedback Equalization

The decision feedback equalization (DFE) is an iterative detection method where the influence from one pulse to the surrounding pulses, the 2D-ISI, is taken into account. With an approximation of the pulse written on the medium, this (expected) interference can be removed from the surrounding pulses. Specific bitpatterns will result in highly influenced pulses by the interference of neighbouring pulses. Pulses might be cancelled or even flipped, which can be seen in figure 4.1(c). The cancellations will be reversed with this detector, making detection of bits in highly influenced signals possible.

The DFE detector used in this work is a two-dimensional implementation of the DFE described in Wang [2]. An implementation of a two-dimensional DFE can also be found in the paper of Keskinoz [8].

In figure 4.8 a schematic view of the DFE can be found. The decision block detects the value of a pulse on the expected peak point of the pulse and uses this value for the pulse description to be subtracted from the signal. With the removal of a pulse, also the 2D-ISI is removed from the surrounding pulses. Now detection can be continued for the next pulse.

When a wrong pulse is detected, this wrong value will be subtracted from the signal, which has influence on the detection of the next pulses. This might result in error bursts. Because pulses on the borders are not influenced by pulses from all sides, these will be better distinguishable. Starting detection with these pulses will prevent the detector from beginning with wrong detected pulses.
Chapter 4. Detector description

In a 1D situation the signal, as a function of the time, is as given in figure 4.9. Three pulses with ISI are given as the input signal. The removal of the expected pulse from the input signal, makes the next pulse better distinguishable, as can be seen in figure 4.9(b) and 4.9(c).

For the 2D signal the situation is similar. In figure 4.10 the steps of the detector can be seen. A signal with a 5x5 bit pattern is used, where all bits are a one (which was also shown by the threshold detector in figure 4.1(c)). Figure 4.10(a) is the input signal of the detector. In figure 4.10(b) the pulses of the first row are subtracted from the signal, removing also the influence from these pulses on the surrounding pulses in the next row. This makes the pulses towards the center better distinguishable. In figure 4.10(c) two more pulses are removed and in figure 4.10(d) the signal is given just before the detection of the pulse in the center.

By comparing figure 4.10(a) with figure 4.10(d) it can be seen that the center value of this 5x5 bit pattern is detected correctly with the DFE detection. In case of a threshold or peak detection this value would be detected wrong due to 2D-ISI of the surrounding pulses.

The DFE detector removes the pulse from the input signal at a certain position. In the presence of jitter, the pulse will be removed from the signal.
Chapter 4. Detector description

(a) Input signal

(b) After first row removal

(c) After two more pulse removals

(d) Detection of center pulse

Figure 4.10: DFE detector signal during several detection steps, no jitter influence.
on a slightly different position than the pulse actually occurs in the signal. The influence on neighbouring pulses will not be removed the way it has influenced these neighbours during writing. There will be influence of jitter on the performance of this detector.

In figure 4.11 the same detection is applied as in figure 4.10, now on a signal with jitter. It can be seen in figure 4.11(d) that after removal of pulses a more noisy signal is left, a result of the remaining ISI. The pulses are removed from the expected peak point, but due to jitter the written pulses are shifted and the influence from ISI is also different than expected.

Figure 4.11: DFE detector signal during several detection steps, with jitter influence.

4.5 Detectors on patterned media

The detectors described in this chapter are used in the detection of the bits on a simulated MFM signal as well as the typical MFM signal given in figure
3.1. By varying the pulse period, signal to noise ratio, jitter and sample rate comparisons can be made between the performances of these detectors. The results of these simulations are given in chapter 6.

With the variation of the pulse period, also the shape of the pulse can be varied. By changing the shape of the pulses also the 2D-ISI changes and detector performance is influenced. The shape variation is not taken into account in the simulations. Only the comparison between detectors with one specific pulse is taken into account.
Chapter 5

Image Processing Techniques

In all methods described in chapter 4 impressions of the signal are shown as grayscale images. On these images dots appear and depending on the presence of noise and jitter, these dots are more or less distinguishable. Instead of using the common detection methods derived from the one-dimensional methods, specific image processing recognition and detection methods can be used. The dots can be detected with object recognition algorithms and so the bit values of the input signal can be detected.

The MFM signal is built of positive and negative pulses with more or less known properties. These pulses are the objects to be recognized and appear as black and white dots on a gray background. The geometry of the pulses is rotational invariant.

Different methods from the book Image Based Measurement Systems by Van der Heijden[16] are applied on the simulation of the MFM signal and described in the next sections.

5.1 Spot Detection

In the signal generated in the simulation, the dots are clearly visible when displayed as an image. From an image processing view this looks like a spot detection problem. A model for the description of the spot field is given in equation 5.1.

\[ f(x, y) = C_0 + \sum_i C_i h(x - x_i, y - y_i) + n(x, y) \]  

(5.1)
Chapter 5. Image Processing Techniques

In this equation $C_0$ is the background and $C_i h$ are the amplitudes of the spots on the respective $(x_i, y_i)$ positions. The goal is to detect the positions of the spots in the $f(x, y)$ image, in which the $C_i$ value is the value of this bit, either $-1$ or $1$.

It is a very obvious way to detect these pulses by comparing the image with a template. The template is a description of the image to be detected. A cross correlation can be calculated between the input image and the template. This cross correlation information can be used in a detector where values on specific positions will exceed a threshold, defining the position of the spot.

In terms of image processing the process for the spot detection now exists of two parts: Feature extraction and classification.

5.2 Feature extraction

Goal of the algorithms is finding the dots in the image. These dots have certain properties, called features. By extracting these features from the MFM signal, and so reducing the disturbances in the signal, the relevant information is left over for detection.

5.2.1 Point spread function

The point spread function (PSF) of the pulses is described in chapter 3 as the pulse description of the simulation signal. This same function can be used in the image processing detection mechanisms for comparison with the input signal. The resolution of the PSF is depending on the resolution of the dots in the MFM signal. An example of the PSF and the associated matrix is given in figure 5.1.

5.2.2 Euclidean Distance

By comparing the MFM signal with a template, a distance can be calculated between these two signals. The position of the template on the signal with the smallest distance is the expected position for a peak. This comparison can be done with the euclidean distance between the image and the template. The calculation for the distance is given in equation 5.2.

$$d^2(\xi, \eta) = \int \int_{x,y} (f(x,y) - h(x - \xi, y - \eta))^2 \quad d(\xi, \eta) = \min\{d(\xi, \eta)\} \quad (5.2)$$

Which can be written as

$$d^2(\xi, \eta) = \int \int_{x,y} (f(x + \xi, y + \eta) - h(x, y))^2 \quad (5.3)$$

The discrete case is given is equation 5.4, where $d^2(\xi, \eta)$ corresponds to
Chapter 5. Image Processing Techniques

(a) Full pulse description  (b) Pulse part of interest

(c) Associated matrix
\[
\begin{bmatrix}
-0.06 & 0.11 & 0.21 & 0.11 & -0.06 \\
0.11 & 0.50 & 0.72 & 0.50 & 0.11 \\
0.21 & 0.72 & 1.00 & 0.72 & 0.21 \\
0.11 & 0.50 & 0.72 & 0.50 & 0.11 \\
-0.06 & 0.11 & 0.21 & 0.11 & -0.06 \\
\end{bmatrix}
\]

Figure 5.1: Example of PSF description

\(d_{n,m} = d(n\Delta, m\Delta)\). In equation 5.2 and 5.4 \(f\) is the MFM signal and \(h\) is the template image, the description of the pulse in the absence of noise. The constants \(K\) and \(L\) should be chosen such that the template image covers the whole object of interest, in this specific case the size of a single dot in the MFM signal.

\[
d_{n,m}^2 = \sum_{k=-K}^{K} \sum_{l=-L}^{L} (f_{n+k,m+l} - h_{k,l})^2 \tag{5.4}
\]

With a template image as close as possible to the written pulse and an input signal with no 2D-ISI the distance will be minimal and approach zero. The 2D-ISI and medium noise will influence the euclidean distance and jitter will change the location of the minimum. An example of the euclidean distance of an input signal to a template is given in figure 5.2. A noisy input signal gives an enhanced output which can be used in the classification step.

Calculating the minimal distance is an intensive operation in case of a MFM signal with a high sample rate. For every position the distance from the template to the input signal is calculated. With low resolution it is hard to distinguish the minimal distance in the presence of noise. With a limited number of samples the noise has relatively larger influence and the pulse description is less distinctive.
Chapter 5. Image Processing Techniques

5.2.3 Convolution

Convolution of the signal with the mathematical description of a pulse is an equivalent of matched filtering. In this detector the signal is convoluted with the PSF.

The operation is a weighted sum of signal levels from a region of the signal. The weight factors are the samples of the pulse description. The matrix of weight factors is called the convolution mask $h$, which is in fact a template image. With a convolution mask that matches the pulses coming from the MFM signal, the weighted sum is maximized on the location where the pulses exist in the MFM signal. After this convolution the pulses will be enhanced and noise will be suppressed. This method is also called linear image filtering: an input signal is lowpass filtered with the description of the expected objects in this input signal.

The convolution can be mathematically described as equation 5.5. The detector starts with the signal $f_{n,m}$ and this signal is convoluted with a convolution mask, $h_{n,m}$. Both are matrices and with the inner product the values for the convoluted signal $g_{n,m}$ results.

$$g_{n,m} = h^t f = \sum_{k=-K}^{K} \sum_{l=-L}^{L} h_{k,l} f_{n-k,m-l} = h_{n,m} * f_{n,m} \quad (5.5)$$

Convolution masks

For optimal feature extraction a suitable convolution mask has to be chosen. The pulse description of the MFM signal can be used as a convolution mask. Due to the 2D-ISI on the MFM signal not all properties of this description
Chapter 5. Image Processing Techniques

will be visible, for example the undershoot of the pulse, which is overlapping with the neighbouring pulses. An adapted convolution mask can give similar results with a reduced number of calculations.

A gaussian pulse can be used as a convolution mask. In equation 5.6 the description for a gaussian pulse is given.

\[ h_{x,y} = A e^{-\left(\frac{(x-x_0)^2}{2\sigma_x^2} + \frac{(y-y_0)^2}{2\sigma_y^2}\right)} \]  \hspace{1cm} (5.6)

Where \( A \) is the amplitude of the pulse. \( x_0 \) and \( y_0 \) the centers of the pulse and \( \sigma_x \) and \( \sigma_y \) the \( x \) and \( y \) spreads.

This pulse as convolution mask can easily be adapted from a circular to elliptical form by using different values for \( \sigma_x \) and \( \sigma_y \). Examples of convolution masks created by a gaussian pulse are given in figure 5.3. In figure 5.3(c) also a rotated mask can be found. In case of the typical MFM signal example given in section 3.4.4 this mask can be used before rotation and transformation of the signal is applied.

![Figure 5.3: Examples of convolution masks](image)

Applying convolution masks

In figure 5.4 the output signals can be found after applying various convolution masks. The MFM signal is given in figure 5.4(a). After convolution with an 8x8 mask of the pulse description of section 3.3.2 the output is given in figure 5.4(b). Figure 5.4(c) shows the output after convolution with a gaussian description for the convolution mask as a 5x5 matrix.

The figure with the Gaussian mask indicates that not an exact description of the written pulse is necessary for this feature extraction. A smaller mask can already enhance the signal and save computation power.
Chapter 5. Image Processing Techniques

In figure 5.5 the result of applying convolution on a noisy MFM signal with a bit pattern of 5x5 ones can be found. The output after convolution is given in figure 5.5(b). The black dots that occur as a result of the 2D-ISI still exist and may cause problems in the classification.

(a) Simulated MFM Output (b) Convolution with pulse description (c) Convolution with gauss pulse

Figure 5.4: Applying different convolution masks

(a) Noisy MFM signal of 5x5 ones (b) Convolution output of 5x5 ones

Figure 5.5: Applying convolution on noisy 5x5 ones

Image border effects

When a convolution is applied, there will occur irregularities on the borders. An example can be seen in figure 5.6. In figure 5.6(b) a disturbance of the bits on all borders is visible.

It is a common problem with convolution, which can be solved by padding the border with the same values as the background, usually zero’s. In the simulations the signal is created with a wide border preventing the occurrence of such effects in the convolution. After the convolution the padding can be removed before further processing. In figure 5.6(c) a padding with zero’s
has been applied and pulses on the borders are not disturbed by the border effect anymore.

![Simulated MFM Output](image1)
![Convolution without padding with zero's](image2)
![Convolution with padding](image3)

(a) Simulated MFM Output (b) Convolution without padding with zero's

Figure 5.6: Border effects with convolution

### 5.2.4 Laplacian of the Gaussian

On the signal after convolution the Laplace operator can be applied. The Laplacian is the second order derivative, given in equation 5.7.

\[
\Delta f = \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}
\]  

(5.7)

The result of this operator on a signal after convolution is a strong response on the positions for the black and white dots on the MFM signal. In figure 5.7(b) the result of this operator applied on figure 5.7(a) can be found.

![Convoluteted MFM signal](image4)
![After application of Laplacian](image5)

(a) Convoluteted MFM signal (b) After application of Laplacian

Figure 5.7: Signal before and after laplacian.
5.3 Classification

After the feature extraction methods the dots in the signal are enhanced. The next step is to localize the dots in the signal and decide whether or not they are a written pulse or a result of ISI and ITI.

5.3.1 Non-local maximum suppression

Besides a way of detecting pulses, there is also the problem of the localization of the pulses. In an ideal situation the detector should be able to localize the pulses, making the use of a clock unnecessary. A non-local maximum suppression algorithm is a method for solving this problem. As the name says, it is a suppression of the non-local maxima. At the output of this algorithm the local maxima are still available, representing the pulses of the signal. With this algorithm the influence of jitter is also minimized.

In the MFM signal, matrix $f$, there are positive and negative peaks. To make all dots positive the absolute value of the signal is taken, resulting in $f_{abs}$. Now for all samples in the signal a comparison can be made with the neighbouring values. When a sample is larger than the samples in its direct surrounding and above the threshold value, the position is classified as a peak. The output matrix $f_{out}$ now consists of values $-1$ or $1$ on peak positions and $0$ elsewhere. The algorithm of this method is given in listing 5.1.

```matlab
f_abs = abs(f);
f_out = zeros(size(f));
threshold = THD;
for i = 1:rowSamples
    for j = 1:colSamples
        if ((f_abs(i,j) >= f_abs(i-1,j)) && ...
            (f_abs(i,j) >= f_abs(i+1,j)) && ...
            (f_abs(i,j) >= f_abs(i,j-1)) && ...
            (f_abs(i,j) >= f_abs(i,j+1)) && ...
            (f_abs(i,j) >= f_abs(i+1,j+1)) && ...
            (f_abs(i,j) >= f_abs(i-1,j-1)) && ...
            (f_abs(i,j) >= f_abs(i+1,j-1)) && ...
            (f_abs(i,j) >= f_abs(i-1,j+1)) )
            if( f_abs(i,j) > threshold ))
                f_out(i,j) = sign(f(i,j));
            end
        else
            f_out(i,j) = 0;
        end
    end
end
```

Listing 5.1: Local Maximum Suppression algorithm in Matlab
The number of samples in the surrounding which is taken into account can be varied. Depending on the size, the sample rate and the period of the dots on the signal.

The result of applying this algorithm to the signal after convolution is given in figure 5.8. In 5.8(a) the convolution output is given, which results in figure 5.8(b) after the algorithm. An input signal with noise can be found in figure 5.9. In that case more local maxima appear, which results in extra positive and negative samples.

In figure 5.8(b) also the effects of 2D-ISI are visible. In the output extra local maxima appear. The 2D-ISI introduces extra peaks. For example the extra white dot between the black dots in the upper left 3x3 bits. To prevent the algorithm from classifying these effects the first step is to change the threshold. When the 2D-ISI is such that the interference peak will be higher than a written pulse, this won’t work. This can be seen in figure 5.10 where in input signal with 5x5 ones is used. The output after non-local maximum suppression in figure 5.10(b) shows extra detected pulses due to the 2D-ISI. A solution would be to require a minimal distance between the position of detected peaks.

![Figure 5.8: Application of non-local maximum suppression algorithm.](image)

### 5.4 Image processing on patterned media

The feature extraction and classification methods described above can be applied on a simulation of the MFM signal. With suitable PSF descriptions and threshold values a comparison between the performance of these methods and the regular detection methods described can be made in the presence of noise and jitter. In the next chapter the results of these simulations are given and discussed.
Figure 5.9: Application of non-local maximum suppression algorithm on noisy signal.

Figure 5.10: Application of non-local maximum suppression algorithm on signal of 5x5 ones.
Results and Discussion

This chapter gives the results of the different detectors with different influence from jitter and medium noise. The simulations are done with the pulse description given in chapter 3 with an input matrix of 25x25 bits. The signal is simulated with varying sample rates and pulse periods. On the borders the pulses are not influenced from all sides, as described in section 3.4.5. By removing two bits on all sides a number of databits of 23x23 bits is used in each simulation signal. This signal is generated many times with a pseudorandom bitpattern.

In conventional storage media a raw bit error rate of $10^{-4}$ is the maximum for a detector to be useful in detection. For the simulations in this work this is also the guideline. If in a series of simulations no errors occur this doesn’t mean the bit error rate is zero. It just means that not enough bits are simulated to detect the errors. However, also detecting only a few errors with a big number of simulations is no measurement for a low BER. Before the BER is statistically significant at least 100 errors have to be detected. This means for a statistically significant BER of $10^{-4}$ at least one million bits have to be simulated.

6.1 Patterned media simulations

For the simulations a bit pattern is generated with individual pulses of 15x15 samples, where the pulse includes the overshoot. These pulses are placed with a pulse period of 7 to 9 samples from peak to peak. In the case of 7 samples a peak will be exactly on the overshoot peak of a neighbouring pulse. The 2D-ISI is maximized in that case. In figure 6.1 the examples of this
signal can be found. A 5x5 bit pattern is generated for different distances and SNR values.

![Signal examples for different distances and SNR values. (a) - (c) - distance 7 samples, (d)-(f) - distance 9 samples.](image)

By using a larger distance between the pulses the influence of the 2D-ISI on the signal can be compared. The result of these simulations with influence of jitter and medium noise is given below.

### 6.1.1 Jitter Influence

Jitter can be described as noise on the position of the pulses. Especially the detectors where the sample value on the expected points is used could suffer from this noise. The jitter will also change the 2D-ISI, some pulses will be closer together than others. In figure 6.2 the bit error rates (BER) for the different detectors can be found.

In figure 6.2(a) it can be seen that the threshold detector (THD) has the highest BER. The convolution improves the detection of the pulses and performs the best. Up to a jitter of 0.6 the BER of $10^{-4}$ is still feasible.

When a signal is simulated with pulses closer to each other, the performance changes. With pulses placed close to each other, the convolution will also have more errors. The convolution is not able to filter out the pulses. The result of a simulation with a peak distance of 8 samples can be seen in figure...
In figure 6.3 the results of the detectors in the presence of more noise, a lower SNR, is given. As expected the convolution will perform better than the other detectors. The noise influence is filtered out and the detector is still able to detect the bits successfully, while the threshold and DFE will have much more errors.

6.1.2 Medium Noise Influence

In figure 6.4 the bit error rates for the different detectors can be found for an increasing signal to noise ratio (SNR). On low SNR the performance of all detectors will be worse, but it is clear that the threshold detector has the
same behaviour as can be seen in the jitter plots. The performance won’t get better above a certain SNR, due to the worst-case patterns in the signal.

6.1.3 Pulse distance

With a variation in the pulse period the influence of 2D-ISI can be examined. In figure 6.5 the BER for a varying period can be found. Each pulse is simulated with 15x15 samples and the peaks of the pulses are placed on a distance of 7 to 11 samples. The BER is given for a SNR of 0 and 4 dB. The pulse which is given in section 3.3.2 is used in the simulations.

In the situation of a distance of 7 samples, the peak of a pulse is exactly on the overshoot peak of a neighbouring pulse. With this maximal 2D-ISI it is expected that detectors suffer from this. As could also be seen in figure 6.4(b) the DFE will perform the best on a small pulse distance, on a higher SNR. In that case the convolution cannot distinguish the pulses.

6.1.4 Lower sample rate

In the previous results the sample rate was 15x15 samples per individual pulse. Also a lower sample rate is simulated. With a sample rate of 7x7 samples per pulse and a pulse distance of 4 samples a signal is generated with 3x3 samples per bit. With a pulse distance of 5 samples a signal with 4x4 samples per bit is generated. On both signals also the triple detector is applied.

In figure 6.6 a BER-SNR plot can be found. In figure 6.6(a) it can be seen that the triple detector is able to reach a BER below $10^{-4}$, by higher SNR values. For the limited number of simulations no errors occurred for the
other detectors. Only for SNR = 0 dB a BER around $10^{-3}$ can be seen.

The result of a simulation with a sample rate of 3x3 samples per bit can be seen in figure 6.6(b). For both the triple and the convolution detector the BER won’t get better above a SNR around 20 dB.

In case of jitter the bit error rates against the signal to noise ratio can be found in figure 6.7. As expected more errors occur and the triple detector is not able to distinguish pulses from each other. In section 6.2 some analysis is done on the detection of the triple detector.
Figure 6.7(b) points out that the DFE is able to reach the best BER in case of jitter.

Figure 6.7: Detector performance for different samples per bit with jitter.

6.1.5 Simple coding

In the paper of Groenland [11] a simple coding scheme is introduced. In this coding scheme the signal is prevented from the occurrence of worst-case patterns where ISI of neighbouring pulses will cancel out one pulse. In a 3x3 bit pattern always one bit has value 1 and one bit has value 0.

In figure 6.8 the uncoded signal with the threshold, DFE and convolution detector is compared with the coded signal of the threshold detector of the paper. For this specific coding method the signal simulation of the paper of Groenland [11] has been used.

Due to a slightly different description of the pulses in the simulation of the paper there is a performance difference in comparison to figure 6.6. All detectors in figure 6.8 are applied on the pulse description from the paper. It can be seen that a coded signal on the threshold detector improves detection, but still the DFE and convolution in an uncoded situation perform better.

6.2 Triple detector

Special attention is given to the triple detector. The performance of the detector is disappointing, so some analysis is done on the signal to get more information about the situation where detection is unsuccessful.

In figure 6.9 a simulated signal with no noise and no jitter is given. There is a distance of 3 samples between the pulses. As expected all pulses are detected
Figure 6.8: Jitter influence on threshold, DFE and convolution detection with coded input signal for THD.

right, which can be seen from the circles around the detected samples in figure 6.9(a).

Figure 6.9: Triple detector on 5x5 bits pattern with 3x3 samples per bit.

In figure 6.10 the signal of a bit pattern with all ones can be seen. The ISI and ITI is very large and the peaks between the bit pulses are detected as maximal values. This is the wrong value and a lot of errors occur. In simulations of a larger signal with random bit values also patterns will occur with 3x3, or even more, bits of the same value.

Figure 6.11 shows a signal with a lot of jitter. Although this jitter is extreme, the picture shows when detection errors occur. In both the first row and first column only 4 pulses are detected, instead of 5. This missed pulse results in
Chapter 6. Results and Discussion

(a) Input signal  
(b) Detected samples

Figure 6.10: Triple detector on 5x5 bits pattern with all ones.

An error burst, because the next detected pulses are shifted one position.

(a) Input signal  
(b) Detected samples

Figure 6.11: Triple detector on 5x5 bits pattern with high jitter.

In figure 6.12 a signal with more pulses is generated. The detected pulses are nicely ordered on a raster, but on the seventh row an error occurs. All other rows have 4 samples, and this row only 3. Now all next pulses on that row are shifted one position. The detected pulses are right, but on the wrong position and this results in a difference between the input and output bitstream.

From this example it is clear that the method of the triple detector is very sensitive to burst errors.
Chapter 6. Results and Discussion

Figure 6.12: Triple detector on 15x15 bits pattern.

(a) Input signal  (b) Detected samples
Conclusions and Recommendations

7.1 Conclusions

Different detectors have been investigated on random patterns with jitter and medium noise. The error rates of the detectors have been compared resulting in the following conclusions.

7.1.1 Detector performance

For the situation with jitter on the received signal, the decision feedback equalization and convolution detectors perform better than the simple threshold detector. The threshold detector always will have errors due to the worst-case patterns, where a pulse is surrounded by pulses of the same sign. This presence of those patterns is of less influence with the other detectors. In the DFE the influences from surrounding pulses are removed and with the convolution the signal is filtered, which enhances the pulses and makes an influenced pulse better visible for detection.

The performance of the convolution detector is the worst in case of pulses close to each other. In that case the pulses can’t be distinguished and detection fails.

7.1.2 Detector choice

The choice of a more intelligent detector will result in a lower error rate on the detection of the pulses especially in the case of jitter and low medium
noise. Because the pulse description of the signal is known it is possible to use this information in the detection. The more computation power needed for detection also has to be taken into account before a detector can be labeled as better. Its performance improvement has to be higher than the factor of extra computation power needed for the detection.

In this work it turned out that the DFE detection performs best with a small pulse period. In the case of a bigger pulse period the convolution can handle the most jitter and noise. An error rate of $10^{-2}$ with the threshold detector is reduced to $10^{-6}$ with the convolution detector, taking about a 100 times more computation power.

7.1.3 Coding

For the simple threshold also used in the paper [11] the use of a relative simple coding scheme already lowers the error rate. For the detectors with DFE and Convolution this difference is much smaller. The use of a coding scheme than lowers the bit-density which is not compensated by a reduction of errors.

The use of another coding scheme which is designed to prevent the worst-case patterns for the DFE and convolution detectors might improve the detection. In this work no further research has been done to this worst-case patterns.

7.1.4 Computation Power

While the image processing techniques perform better than the regular methods, this also has its cost. The DFE and convolution need more computations for the detection of a pulse. In the MatLab simulations it turned out that these simulations take up to a 100 times more computation power, resulting in an error rate which is, even in case of jitter, several orders of magnitude lower than with just threshold detection. This increase of computation power is compensated by the improvement in error detection.

7.2 Recommendations

7.2.1 Worst-case patterns

In the detection with the simple threshold it turned out that there are a relative big number of errors in the case of no jitter. From the work done within the TST-SMI group it turned out this is due to worst-case patterns in which a pulse is cancelled by the influence from its surrounding. For the other detectors the error rate is very low in the case of no jitter, but it is unknown what the exact reason is for these errors. This might be related to worst-case patterns and can be researched further.
7.2.2 Computation Power
A more intelligent detector also needs more computation power. In this research the difference between computation power has only taken into account by comparing the time needed for the calculations in MatLab. There can be done more research on this comparison in order to conclude if the computations for a detector are worth the improvement on the error rate.

7.2.3 Non-linear signal patterns
The detected signal coming from the MFM is not directly usable in the detector due to non linearities of the readout, like a rotation and perspective, which can be seen when shown as an image. With image processing tools this can be compensated before any detection is applied. It should be studied if it is possible to convert the signal in such a way that the linear detection methods which are used in this work can be used directly.

In a paper of Tang et. al. [15] a technique is described for write synchronization in a bit patterned medium. The synchronization is performed on a disk drive and a spin stand comparable to writing to a disk. Although this is not the same as bit patterned media, the technique could be useful.

7.2.4 Image Processing
As expected the convolution detector has a performance which is better than a simple threshold method, but the image processing field has more techniques and between these methods also variations occur. In this work only the convolution has been done with the description of one signal pulse, whether it might also be possible to use other filter descriptions with for example more than one pulse included, which might improve the detection of the signal.

7.2.5 Bits on signal border
In the simulations it turned out that the pulses on the border of the field should not be used in the calculation of the error rate. The pulses on the border are not disturbed by influence from all directions. In the simulations it is important to remove this surrounding and only use the databits.

7.2.6 Hexagonal Pattern
A change in the ordering of the bits in the pattern gives another description of the 2D-ISI. With a hexagonal pattern the distance to all direct neighbouring pulses will be equal, so also the 2D-ISI from all pulses will be equal. This will influence the detection rates and might result in a higher bit density by placing the pulses closer to each other.
Chapter 7. Conclusions and Recommendations
Appendix A

Simulator and detector code

In this appendix the code for the signal simulation and the detectors is given.

A.1 Signal Simulation

A.1.1 Pulse generator

```matlab
function pulse = pulse2d(pulselength, amplitude, pw50, origins, noise)

% Create a lorentzpulse

% Input:
% pulselength (resolution),
% amplitudes [a1 a2 a3],
% pw50 [p1 p2 p3],
% origins [o1 o2 o3]
% Output:
% two dimensional matrix with pulse of size pulselength
% (or plot if no output is asked on function call)

% Set variables if not given as input
if nargin < 2, amplitude = [-1.1 2.45 -1.1]; end
if nargin < 3, pw50 = [1.9 1.9 1.9]; end
if nargin < 4, origins = [-0.7 0 0.7]; end
if nargin < 5, noise = 0; end % Noise is given as the variance
if nargin < 1, pulselength = 61; end

% make empty matrix
[X,Y] = meshgrid(-floor(pulselength/2):1:floor(pulselength/2));
R = sqrt(X.^2 + Y.^2);
lorentz = zeros(size(R));

% Combine 3 lorentzpulses
```
% with their amplitudes, origins and pulsewidths
for i = 1:3
    lorentz = lorentz + (amplitude(i)./(1+((R./(pulselength*0.1))-... 
    origins(i))./(pw50(i)/2)).^2));
end

% plot or give output
if nargout < 1
    figure;
    mesh(X,Y,lorentz);
else
    pulse = lorentz;
end

Listing A.1: Pulse generator

A.1.2 Pattern generator

function [useField input] = squarefield(rows, cols, ...
    resolution, distance, noise, jitter, input)
% Function to create a square field
% build from combined pulses

%% Set variables
amplitude = [-1.1 2.45 -1.1];
pw50 = [1.9 1.9 1.9];
origins = [-0.7 0 0.7];
index = 1;

% Use standard pulse
xyPulse = pulse2d(resolution, amplitude, pw50, origins, noise);
field = zeros((rows)*resolution,(cols)*resolution);
if nargin < 7
    input = round(rand(rows,cols)).*2-1;
else
    if size(input) ~= [rows, cols]
        disp('Cols and Rows do not correspond to size(input)');
    disp(' ');
    return
end
if size(input) ~= [rows, cols]
    disp('Cols and Rows do not correspond to size(input)');
    disp(' ');
    return
end
for i = 1:rows
    for j = 1:cols
        % Get a random jitter distortion for the created signal.
        jitterX = 0; jitterY = 0;
        if i>1
            jitterX = round(jitter.*randn(1));
        end
        if j>1
            jitterY = round(jitter.*randn(1));
        end
        % calculate start- and endpoint for each pulse,
        % depending on the distance between pulses


% and jitter influence
startPointX = (i -1)*resolution+1-((i-1)*distance)+jitterX;
endPointX = i*resolution-((i-1)*distance)+jitterX;
startPointY = (j-1)*resolution+1-((j-1)*distance)+jitterY;
endPointY = j*resolution-((j-1)*distance)+jitterY;
% generate noise
ruis = noise.*randn(endPointX-startPointX+1,endPointY-startPointY+1);
% Add pulse and noise to signal (on calculated positions
field(startPointX:endPointX,startPointY:endPointY) = ...
field(startPointX:endPointX,startPointY:endPointY) + ...
input(i,j)*xyPulse + ruis;
end
useField = zeros(j*resolution-((j-1)*distance),i*resolution-((i-1)*distance));
useField(1:endPointY, 1:endPointX) = field(1:endPointY, 1:endPointX);

% Plot an image of the generated field
% only if not output matrix is asked
if nargout == 0
    colormap(gray)
    imagesc(useField);
end

function [output samples] = threshold2d(input, ... 
rows , cols , resolution , distance)
% Function to detect signals in a 2D field.
% Pulses expected in the middle of 'resolution'
% with 'distance' between pulses
output = zeros(rows,cols);
peakX = round(resolution/2);
peakY = round(resolution/2);
samples = zeros(1, rows*cols);
k = 1;
threshold = 0;
for i = 1:rows
    for j = 1:cols
        if input(peakX,peakY) > threshold || ... 
           input(peakX,peakY) < -1*threshold
            % bit output based on sign of input signal
            output(i,j) = sign(input(peakX,peakY));
        end
        samples(k) = input(peakX,peakY);
        % calculate new peak position (Y-direction)
        peakY = peakY + resolution - distance;
        k = k+1;
    end
end

Listing A.2: Pattern Generator

A.2 Detectors

A.2.1 Threshold detector

function [output samples] = threshold2d(input, ... 
rows , cols , resolution , distance)
% Function to detect signals in a 2D field.
% Pulses expected in the middle of 'resolution'
% with 'distance' between pulses
output = zeros(rows,cols);
peakX = round(resolution/2);
peakY = round(resolution/2);
samples = zeros(1, rows*cols);
k = 1;
threshold = 0;
for i = 1:rows
    for j = 1:cols
        if input(peakX,peakY) > threshold || ... 
           input(peakX,peakY) < -1*threshold
            % bit output based on sign of input signal
            output(i,j) = sign(input(peakX,peakY));
        end
        samples(k) = input(peakX,peakY);
        % calculate new peak position (Y-direction)
        peakY = peakY + resolution - distance;
        k = k+1;
    end
end

Listing A.2: Pattern Generator
% calculate new peak position (X-direction)
peakX = peakX + resolution - distance;
peakY = round(resolution/2);
end

Listing A.3: Threshold detector

A.2.2 Peak detector

function [ peakOutput peakSamples ] = peak2d( mfmOutput , ... 
    rows , cols , resolution , distance)
    % Peak detector for detection of the original signal
    % of the MFM
    % Inputs: mfmOutput, rows, cols, resolution, distance
    % Output: detected bits (size(rows,cols)), sample values
    % Set sizes output values
    peakSamples = zeros(1, rows*cols);
    peakOutput = zeros(rows*cols);
    % Peak detector:
    % The peak detector given in Wang Chapter 11.
    % For this detector a differentiator is used to detect
    % zero crossings. The signal is also put into a
    % rectifier and a threshold detector. When a zero
    % crossing is detected and the threshold is reached,
    % a peak is detected.
    % Differentiator
    differentiatedOutput = diff(mfmOutput);
    % Rectifier
    % first rectify the signal,
    threshold = 0.4;
    signalLength = min(size(mfmOutput));
    rect = zeros(signalLength);
    for i = 3: signalLength
        for j = 3: signalLength
            if mfmOutput(i,j) > threshold && ...
                mfmOutput(i,j-1) > threshold && ...
                mfmOutput(i,j-2) > threshold && ...
                mfmOutput(i-1,j) > threshold && ...
                mfmOutput(i-1,j-1) > threshold && ...
                mfmOutput(i-1,j-2) > threshold && ...
                mfmOutput(i-2,j) > threshold && ...
                mfmOutput(i-2,j-1) > threshold && ...
                mfmOutput(i-2,j-2) > threshold

                % Signal above threshold on 3x3 samples
                rect(i-2:i,j-2:j) = 1;
            elseif mfmOutput(i,j) < -1*threshold && ...
                mfmOutput(i,j-1) < -1*threshold && ...
                mfmOutput(i,j-2) < -1*threshold && ...

                rect(i-2:i,j-2:j) = 1;
        end
    end

% calculate new peak position (X-direction)
peakX = peakX + resolution - distance;
peakY = round(resolution/2);
end
Chapter A. Simulator and detector code

44 \texttt{mfmOutput(i-1,j) <- 1*threshold \&\& ...}
45 \texttt{mfmOutput(i-1,j-1) <- 1*threshold \&\& ...}
46 \texttt{mfmOutput(i-1,j-2) <- 1*threshold \&\& ...}
47 \texttt{mfmOutput(i-2,j) <- 1*threshold \&\& ...}
48 \texttt{mfmOutput(i-2,j-1) <- 1*threshold \&\& ...}
49 \texttt{mfmOutput(i-2,j-2) <- 1*threshold}

50 \% signals below negative threshold on 3x3 samples
51 \texttt{rect(i-2,i,j-2:j) = -1;}
52 \texttt{end}
53 \texttt{end}
54 \texttt{end}
55
56 \% Start detection
57 \texttt{firstPeak = 0;}
58 \texttt{peakOutput2 = zeros(rows,cols);}
59 \texttt{for i = 1:rows}
60 \texttt{for j = 1:cols}

61 \% define detection window
62 \texttt{peakPointX = (i-1)*(resolution-distance)+ ...}
63 \texttt{round(resolution/2);}
64 \texttt{peakPointY = (j-1)*(resolution-distance)+ ...}
65 \texttt{round(resolution/2);}
66 \texttt{halfWindow = round((resolution-distance)/2);}
67 \texttt{currentMFMWindow = mfmOutput(peakPointX-halfWindow:peakPointX+ ...}
68 \texttt{halfWindow,peakPointY-halfWindow:peakPointY+halfWindow);}
69 \texttt{currentWindow = differentiatedOutput(peakPointX-halfWindow: ...}
70 \texttt{peakPointX+halfWindow,peakPointY-halfWindow:peakPointY+halfWindow);}
71 \texttt{currentRect = rect(peakPointX-halfWindow:peakPointX+halfWindow, ...}
72 \texttt{peakPointY-halfWindow:peakPointY+halfWindow);}

73 \% Start comparison on detection window
74 \texttt{[windowX windowY] = size(currentWindow);}
75 \texttt{for k = 5:windowX-1}
76 \texttt{for l = 5:windowY-1}
77 \texttt{if (currentWindow(k-3,l) >= 0 \&\& currentWindow(k,l) <= 0 \&\& ...}
78 \texttt{all(all(currentRect(k-2:k,l-2:l)) \&\& firstPeak == 0)}
79 \texttt{peakOutput2(i,j) = 1;}
80 \texttt{rectAndDiff(k-2:k,l-2:l) = ones(3,3); \% For displaying steps}
81 \texttt{firstPeak = 1;}
82 \texttt{elseif (currentWindow(k-3,l) <= 0 \&\& currentWindow(k,l) >= 0 \&\& ...}
83 \texttt{all(all(currentRect(k-2:k,l-2:l)) \&\& firstPeak == 0)}
84 \texttt{peakOutput2(i,j) = -1;}
85 \texttt{firstPeak = 1;}
86 \texttt{end}
87 \texttt{end}
88 \texttt{end}
89 \texttt{firstPeak = 0;}
90 \texttt{end}
91 \texttt{end}
92 \texttt{peakOutput = peakOutput2;}
93 \texttt{end}

Listing A.4: Peak detector
function [dfeOutput samples] = dfe2d(input, rows, cols, ...
    resolution, distance)
% Function to detect signals in a 2D field using the Decision Feedback Equalization
% Inputs: input signal, number of rows, number of columns,
% resolution, distance
% Output: output bits, sample values
% Input signal is including empty surrounding, pulses are
% removed on expected peak points (resulting in jitter influence)

% Create output matrix
dfeOutput = zeros(rows,cols);

% Create ideal pulse, used for removal from signal
amplitude = [-1.1 2.45 -1.1];
pw50 = [1.9 1.9 1.9];
origins = [-0.7 0 0.7];
xyPulse = pulse2d(resolution, amplitude, pw50, origins, 0);

peakX = round(resolution/2); % start value for x direction
peakY = round(resolution/2); % start value for y direction
factor = 1; % removal factor

% set samples matrix (for sample histograms)
samples = zeros(1, rows*cols);
k = 1; % samples Counter

for i = 1: rows
    for j = 1: cols
        % bit value is sign of value on expected peak point
        dfeOutput(i,j) = sign(input(peakX,peakY));
samples(k) = input(peakX,peakY);
k = k+1; % increase sample counter
        peakY = peakY + resolution - distance; % calculate new peak value
    end
    peakX = peakX + resolution - distance; % Calculate new peak value
end

for i = 1:rows
    for j = 1:cols
        % bit value is sign of value on expected peak point
        dfeOutput(i,j) = sign(input(peakX,peakY));
samples(k) = input(peakX,peakY);
k = k+1; % increase sample counter
        peakY = peakY + resolution - distance; % calculate new peak value
    end
    peakX = peakX + resolution - distance; % Calculate new peak value
end

peakX = peakX + resolution - distance; % Reset y direction

66
Listing A.5: DFE detector

A.2.4 Triple detector

```matlab
function [trioOutput samples detectedField] = trio2d(input, rows, cols, resolution, distance) % Function to detect signals in a 2D field. % Using 3x3 samples per bit

% Calculate parameters
trioOutput = zeros(rows,cols); mfmLength = size(input); startPointHeight = round(distance/2); startPointWidth = round(distance/2); fieldToBeSampled = input(startPointHeight:(mfmLength(1)-startPointHeight), ... startPointWidth:(mfmLength(2)-startPointWidth)); startHeight = numel(fieldToBeSampled(1,:)); startWidth = numel(fieldToBeSampled(:,1)); expectedHeight = 3*rows; expectedWidth = 3*cols;

% Resample input signal to 3x3 samples per bit
trioField = resample(resample(fieldToBeSampled,expectedHeight, ... startHeight,0),expectedWidth,startWidth,0)'; sizeTrioField = size(trioField); trioField2 = zeros(sizeTrioField(1)+1,sizeTrioField(2)+1); trioField = trioField2; smallDetectedField = zeros(size(trioField)); detectedField = zeros(size(input));

[maxX maxY] = size(fieldToBeSampled);

% Calculate and set initial detection parameters
endPoint = numel(trioField(:,1))-startPointHeight-1; endPointDistance = 2; startX = 1; startY = 1; endX = startX + endPointDistance; endY = startY + endPointDistance; samples = zeros(1,rows*cols); k = 1;

[maxX maxY] = size(trioField);

trioOutputIndex = zeros(rows,cols,2); % Define a index matrix

% loop through signal
for i = 1:rows
    for j = 1:cols
        % Set new sample values, based on index of previous bit
        if i > 1 % only on column > 1
```
Chapter A. Simulator and detector code

if trioOutputIndex(i-1,j,2) > 0
    startY = trioOutputIndex(i-1,j,2)+2;
else
    startY = startY;
end
% startY should not change if previous detection has failed

end
else
    startY = 1;
    endY = startY + endPointDistance;
end

% Set new sample values, based on index of previous bit
if j > 1 % only on row > 1
    if trioOutputIndex(i,j-1,1) > 0
        startX = trioOutputIndex(i,j-1,1)+2;
    else
        startX = startX;
    end
    endX = endX + endPointDistance;
else
    startX = 1;
    endX = startX + endPointDistance;
end
endY = startY + endPointDistance;
clear values;

% Detect the absolute maximum in the current 3x3 sample signal
if (startX < maxX && endX <= maxX && startY < maxY && endY <= maxY)
    [values indices] = max(abs(trioField(startX:endX,startY:endY)));
    [maxValue maxIndex] = max(values);
    % Save the index values for each detected maximum
    trioOutputIndex(i,j,:) = ...
        [startX+indices(maxIndex)-1 startY+maxIndex-1];
    % rows and cols changed due to resampling, j,i inverse
    % Output for detector is based on sign of the signal
    trioOutput(j,i) = ...
        sign(trioField(startX+indices(maxIndex)-1,startY+maxIndex-1));
smallDetectedField(startX+indices(maxIndex)-1,startY+maxIndex-1) = 1;
samples(k) = trioField(startX+indices(maxIndex)-1,startY+maxIndex-1);
end
k = k+1;
end
startX = 1;
endX = startX + endPointDistance;

Listing A.6: Triple detector
A.3 Image processing

A.3.1 Convolution

```matlab
function [convOutput convSamples] = ...
    convDetection2d(mfmOutput, rows, cols, colRes, rowRes, distance)
% convolution detector for detection of the original signal of
% the MFM using the most simple detector after convolution
% mfmOutput is convoluted and samples are taken at the
% expected peak points, sign of sample is the bit value
% Inputs: mfmOutput, distance, resolution, length
% Output: Original bitstream

%% set variables:
convOutput = zeros(rows, cols);
convSamples = zeros(1, rows*cols);
k=1;
% Define field borders
rasterSize = colRes - distance;
pulseHeigth = round((1/6)*rowRes);
pulseWidth = round((1/6)*colRes);
scanFieldWidthEnd = round((1/3)*colRes)-pulseWidth;
scanFieldHeightEnd = round((1/3)*rowRes)-pulseHeigth;
partPointX = (colRes - pulseWidth )/2;
partPointY = (rowRes - pulseHeigth)/2;

%% Create comparison pulse
amplitude = [-1.1 2.45 -1.1];
pw50 = [ 1.9 1.9 1.9];
noise = 0;
firstPeak = 0;
pulse = pulse2d(colRes, amplitude, pw50, origins, noise);
pulsePart = ...
    pulse(partPointX:partPointX+pulseWidth,partPointY:partPointY+pulseHeigth);

%% Convolution
% Whole signal is convoluted
convolutionOutput = conv2(mfmOutput, pulsePart(2:10,2:10),'full');

%% Apply detector
offset = 4;
for i = 1:cols
    for j = 1:rows
        sampleX = round(colRes/2)+offset+(i-1)*(colRes-distance);
        sampleY = round(rowRes/2)+offset+(j-1)*(rowRes-distance);
        convOutput(j,i) = sign(convolutionOutput(sampleY,sampleX));
    end
end
end
```
A.3.2 Non Local Maximum Detection

```matlab
f_abs = abs(f);
f_out = zeros(size(f));
threshold = THD;
for i = 1:rowSamples
    for j = 1:colSamples
        if (((f_abs(i,j) >= f_abs(i-1,j)) && ...
            (f_abs(i,j) >= f_abs(i+1,j)) && ...
            (f_abs(i,j) >= f_abs(i,j-1)) && ...
            (f_abs(i,j) >= f_abs(i,j+1))) && ...
            (f_abs(i,j) >= f_abs(i-1,j-1)) && ...
            (f_abs(i,j) >= f_abs(i+1,j+1)) && ...
            (f_abs(i,j) >= f_abs(i-1,j+1)))
            if( f_abs(i,j) > threshold ))
            f_out(i,j) = sign(f(i,j));
        end
        else
            f_out(i,j) = 0;
        end
    end
end
```

Listing A.8: Local Maximum Suppression algorithm in Matlab
Bibliography


[10] H. Song, V. Kumar, *Low-Density Parity-Check codes for Partial Response Channels*, 2004


