



LAWRENCE BERKELEY NATIONAL LAB  
COLLEGE OF ENGINEERING AND  
ARCHITECTURE OF FRIBOURG

BACHELOR THESIS

---

Restoration of Historical Dictabelt  
Recordings

---

*Author:*  
Stefan SPACK

*Supervisor:*  
Dr. Carl HABER

*Expert:*  
Noe LUTZ

*Professors:*  
Prof. Eric FRAGNIERE  
Prof. Ottar JOHNSON

August 9, 2013

## **Abstract**

The 22 of November 1963, now almost 50 years ago, John F. Kennedy was assassinated in Dallas, Texas. During the assassination the police radio of the Dallas police department was recorded on a dictabelt. Many experts have analyzed those recordings, but none of them found a concluding answer on the question of the theory that Kennedy was shoot from two different shooters. Some of them assert that two different shooters can be heard on it, others are asserting that there are no gun shots at all recorded on the belt.

The National Archives asked the Lawrence Berkeley National Lab to restore those recordings, which are actually in a bad condition, for the 50th anniversary of the assassination.

The goal of this bachelor thesis was to improve the sound extraction quality of the PRISM software applied on dictabelts, by creating an algorithm which compensates the error of the mismatch of the 180 measuring points, by using the differences between two measurements of the same measuring point instead of the measuring itself.

The results showed a better signal to noise ratio for high frequencies compared to the old QuadAlu algorithm, but a lower signal to noise ratio for low frequencies. Good results could be obtained by mixing the sound of both algorithms together. As the noise of both algorithm is uncorrelated, if they are mixed together the noise compensate itself partially.

### **Acknowledgments**

The author would like to thank Carl Haber and Earl Cornell for their help, advices and critics during this project. Silvan Fischer and Romain Crausaz for their advices for algorithmic problems. Eric Fragniere, Ottar Johnsen and Noe Lutz for their critics and advices during the project. Carl Haber and the Lawrence Berkeley National Lab giving him the opportunity to do this interesting project.

Last but not least the College of Engineering and Architecture of Fribourg to supporting him financially during this project.

# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
<b>2</b>	<b>Theory</b>	<b>6</b>
2.1	Definitions . . . . .	6
2.1.1	Directions . . . . .	6
2.1.2	Sound . . . . .	7
2.2	Scan of the media . . . . .	9
2.2.1	2D Scan . . . . .	9
2.2.2	3D Scan . . . . .	10
2.3	Dictabelt . . . . .	11
2.3.1	The dictabelt media . . . . .	11
2.3.2	Restoring dictabelts . . . . .	13
2.4	Scan Process . . . . .	14
2.5	PRISM . . . . .	17
2.5.1	Sound extraction process . . . . .	17
2.5.2	Interface . . . . .	19
2.6	Sound Forge . . . . .	20
2.7	Statistics . . . . .	21
2.7.1	Gaussian distribution . . . . .	21
2.7.2	Histogram . . . . .	24
2.8	Spline interpolation . . . . .	25
<b>3</b>	<b>Specification of the Project</b>	<b>26</b>
3.1	Goal of the Work . . . . .	26
3.2	Functional specification . . . . .	26
3.3	Operational specification . . . . .	27
3.3.1	Improvement of the Interface . . . . .	27
3.3.2	Interpolation of corrupted parts . . . . .	27
3.3.3	Extract data with derivative method . . . . .	28
3.3.4	Correct distortion made by recording device . . . . .	28
3.3.5	Apply results to other types of media . . . . .	28
<b>4</b>	<b>Improvement of the PRISM Software</b>	<b>29</b>
4.1	Add a cursor to the soundgraph . . . . .	29
4.2	Enlarged groove picture . . . . .	30
<b>5</b>	<b>First implementation of the derivative BeltDeriv algorithm</b>	<b>31</b>



<b>6</b>	<b>Validation</b>	<b>34</b>
6.1	Hardware . . . . .	34
6.2	Software . . . . .	36
6.2.1	Sweep generator . . . . .	36
6.2.2	Recording stage . . . . .	36
6.3	Tests . . . . .	37
6.4	Results . . . . .	38
<b>7</b>	<b>Improvements in the algorithm</b>	<b>40</b>
7.1	Separate the slopes . . . . .	40
7.2	Detection and interpolation of blobs . . . . .	40
7.2.1	Detection . . . . .	40
7.2.2	Interpolation . . . . .	41
7.3	Vertical binning . . . . .	41
<b>8</b>	<b>Final BeltDeriv Algorithm</b>	<b>42</b>
8.1	Parameter and their effect on the algorithm . . . . .	42
8.2	Flowcharts . . . . .	44
8.2.1	Top level . . . . .	44
8.2.2	Calculation of the derivatives . . . . .	45
8.3	Results . . . . .	46
8.4	Get the right parameters . . . . .	47
<b>9</b>	<b>Test of BeltDeriv algorithm on Aluminum discs</b>	<b>48</b>
9.1	Aluminum discs . . . . .	48
9.2	Results . . . . .	49
<b>10</b>	<b>Conclusion</b>	<b>51</b>
10.1	Noise comparison . . . . .	51
10.2	Clicks . . . . .	52
10.3	Kennedy records . . . . .	52
10.4	Future Improvements . . . . .	53
<b>11</b>	<b>References</b>	<b>54</b>
<b>12</b>	<b>Annex</b>	<b>56</b>
12.1	Index of the DVD . . . . .	56
12.2	Labview Generator / Recorder . . . . .	57
12.2.1	Interface . . . . .	57
12.2.2	Generator . . . . .	58
12.2.3	Linear sweep block . . . . .	59
12.2.4	Logarithmic sweep block . . . . .	60
12.2.5	Switch for frequency sweep . . . . .	60
12.2.6	Sound capture . . . . .	61

# List of Figures

1.1	John F. Kennedy before the shoot . . . . .	5
2.1	Top view direction definition . . . . .	6
2.2	Profile direction definition . . . . .	7
2.3	Grooves of a dictabelt . . . . .	7
2.4	Origin and result of a blob . . . . .	8
2.5	2D-scan bench . . . . .	9
2.6	MPLS180 measurement principle . . . . .	10
2.7	Empty Dictabelt . . . . .	11
2.8	Dictabelt recorder . . . . .	12
2.9	Surface and profile of a dictabelt . . . . .	13
2.10	Measurement process to scan the media . . . . .	14
2.11	Scan of a dictabelt on the cylinder scan bench with the MPLS180 probe . .	16
2.12	Flowchart of the PRISM software . . . . .	17
2.13	Image and profile of Raw image . . . . .	17
2.14	Image and profile of Match pass (Match ALU) . . . . .	18
2.15	Tracked groove on a dictabelt . . . . .	18
2.16	Interface of PRISM . . . . .	19
2.17	Screen shot of Sound Forge . . . . .	20
2.18	Gaussian distribution . . . . .	22
2.19	Sample of the profile of the unfiltered image . . . . .	23
2.20	Sample of the same profile after the 1-D filtration . . . . .	23
2.21	Histogram of normal distributed random numbers . . . . .	24
2.22	Interpolation methods . . . . .	25
4.1	New sound graph with added sound cursor . . . . .	30
5.1	Groove movement . . . . .	32
5.2	GUI of the Belt Deriv algorithm . . . . .	33
6.1	NI Automation Rack (NI PXI-1031) . . . . .	34
6.2	Schematic of the hardware used to Record and Playback dictabelts . . . . .	35
6.3	Comparison of the first results on the dictabelt . . . . .	38
6.4	Comparison of Needle playback, QuadAlu and BeltDeriv algorithm . . . . .	39
8.1	GUI of the final BeltDeriv algorithm . . . . .	42
8.2	Toplevel of the BeltDeriv Algorithm . . . . .	44
8.3	Calculation of the derivatives . . . . .	45

8.4	Comparison of Needle playback, QuadAlu and BeltDeriv algorithm . . . . .	46
9.1	Aluminum disc of the Millman Parry collection . . . . .	48
9.2	Samples of an aluminum disc . . . . .	49
9.3	Comparison of the results on aluminum discs . . . . .	49
9.4	Comparison of QuadAlu and BeltDeriv algorithm . . . . .	50
10.1	Compare of clicks in wav-files . . . . .	52
10.2	Blob cleaning detecting fold . . . . .	53
12.1	Labview: Interface . . . . .	57
12.2	Labview: Function Generator . . . . .	58
12.3	Labview: Linear sweep . . . . .	59
12.4	Labview: Logarithmic sweep . . . . .	60
12.5	Labview: Sweep Switch . . . . .	60
12.6	Labview: Capture . . . . .	61

# Chapter 1

## Introduction

This bachelor thesis was written at the Lawrence Berkeley Lab in California, to finish my studies at the College of Engineering and Architecture of Fribourg, Switzerland in the field of electrical engineering.



Figure 1.1: John F. Kennedy before the shoot [1]

if there were more than one shooter, or just one, Lee Harvey Oswald, which is the official version. Several times in the past the dictabelt was analyzed by different experts, but there was no final conclusion about this question.[2]

Nowadays the belt is no more playable with a dictabelt recorder because the belt is cracked. The only way the original can be played is by scanning it contact-less.

Former theses were written on similar subjects at the lab under the supervision of Dr. Carl Haber.

The subject of this thesis is an attempt improve the extraction of the sound information carried on a dictabelt. The motivation of this project is to restore the dictabelt recordings, made during the assassination of John F. Kennedy, the 22th November 1963 at 12:30pm in Dallas, Texas. Since this year the assassination will mark the 50th anniversary. The National Archives asked the Lawrence Berkeley National Lab to restore those records.

The recording contains the traffic on police radio 1 of the Dallas police department, approximately between 12:29pm until 12:35pm. During this time period a motorcycle was transmitting continuously on the police radio.

This record was subject to conspiracy theories on the assassination. The question is

## Chapter 2

# Theory

### 2.1 Definitions

#### 2.1.1 Directions

As in this document directions are used, to enhance the comprehension, the directions of the images are defined as follows:

**X-direction:**

Perpendicular to the grooves, Sound of dictabelts are modulated in x-direction.

**Y-direction:**

Direction of the grooves, in sound space y-direction correspond to the time direction.

**Z-direction:**

Depth of the groove

**Line:**

All pixels with the same Y index form together a line

PRISM offers two different views. Firstly, on the top view on the media, used in the main picture and in the enlarged picture on the right side of the GUI, the directions are defined as follows:

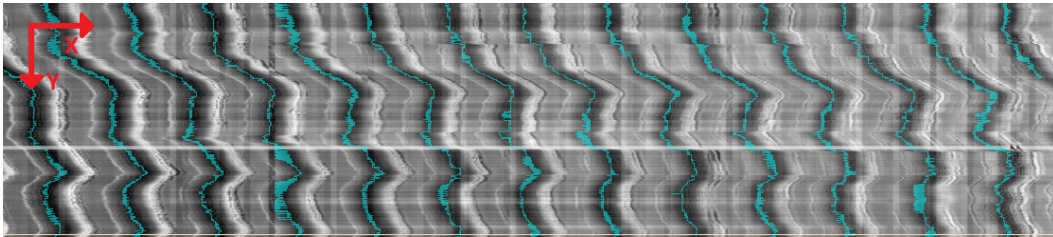


Figure 2.1: Top view direction definition

Secondly, the profile views on the bottom of the GUI are using the following definition of direction:

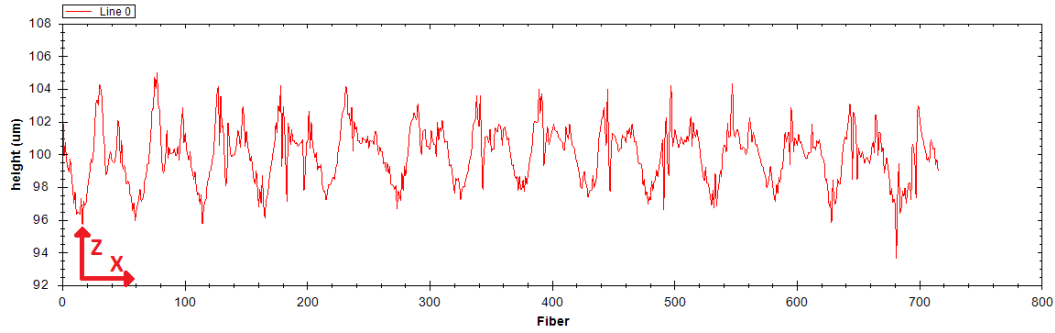


Figure 2.2: Profile direction definition

## 2.1.2 Sound

### Grooves

The groove is the continuous pit in the recording media, cut or embossed by the recording needle, and into which the playback needle is sliding. The sound is modulated in the movement of the groove. On some records the sound is modulated by variation of the depth of the groove, on others the sound is modulated by lateral movement of the groove. All the analog sound media, except the magnetic ones, are using this principle. The dictabelts are modulated laterally, contrary to their predecessors the wax-cylinders.

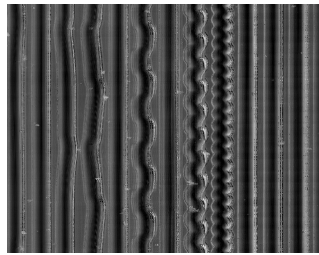


Figure 2.3: Grooves of a dictabelt

## Technology to record and playback of sound with a needle

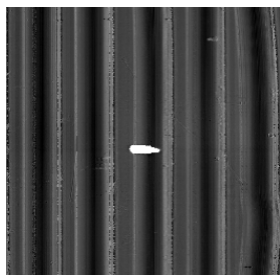
The principle of recording and the playback with a needle are the same, just the cause and the effect are changed. The principle is explained for playback:

The tip of the needle is following the groove movement. On the other side of the needle a magnet is placed. As the magnet is placed in an coil, a movement of the needle is creating an induction in the coil. This movement can be measured as an electric current, which contains the sound. As the induced current is proportional to the derivative of the movement, the sound is also proportional to the derivative of the groove movement. As the optical extraction process, developed in the IRENE project, gets a position of the groove, this information must also be derivated to extract the sound information. The most algorithms performs this derivation at the end of the process, but the developed algorithm does this at the beginning of the process.

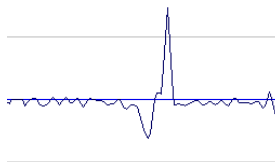
### Blobs and clicks

The recording media aren't clean in a microscopically scale. They are covered by dust particles or other imperfections on the surface, like small cracks. In the audio restoration project, which lasts now for almost ten years, these imperfections are called blobs. These blobs can be heard as clicks, which means a short peek (less than 3 ms) in the sound with a large amplitude.

If the dictabelts are played with a needle, the extracted sound is less affected by these imperfections compared with the optical extraction. Probably the needle can push the dust particles out of the groove, and small cracks or holes are not heard, because the needle is too big to follow them. But this is not the case for all kinds of sound media.



(a) Small dust particle in a groove



(b) Click in a Wavefile, pulse duration  $\approx 1ms$

Figure 2.4: Origin and result of a blob

## 2.2 Scan of the media

In the former audio restoration projects two different capture principles of the sound media were used.

### 2.2.1 2D Scan

The two dimensional Scan uses a high resolution monochromatic digital camera to scan the surface. This principle would be 20 to 80 times faster than the 3 dimensional scan, but delivers only good results if the top of the groove is flat. This is the case for gramophone discs. For the dictabelts it isn't the case, because the groove isn't cut, but embossed with a diamond into the plastic.

The captured data is a image of the slopes of the recording media. Flat areas are bright, and steep areas are dark.

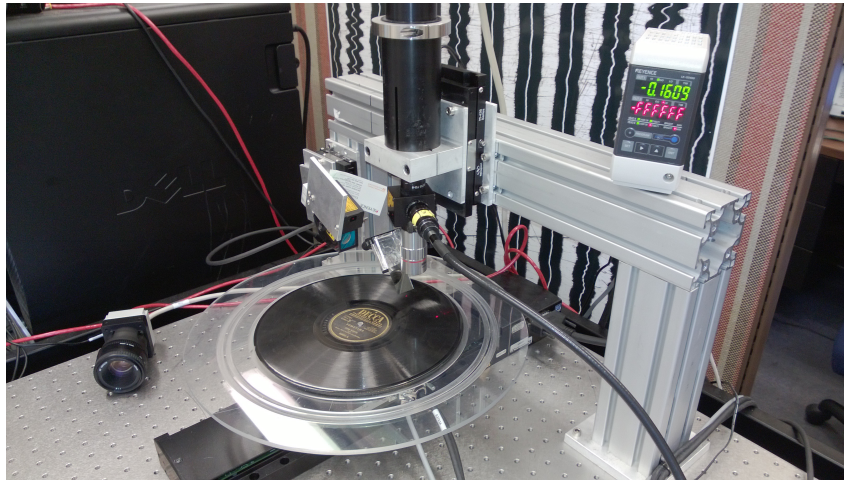


Figure 2.5: 2D-scan bench



### 2.2.2 3D Scan

This method uses a chromatic confocal line sensor, the MPLS180 manufactured by the STILSA company. It consists of 180 independent depth measurement points arranged in a line.

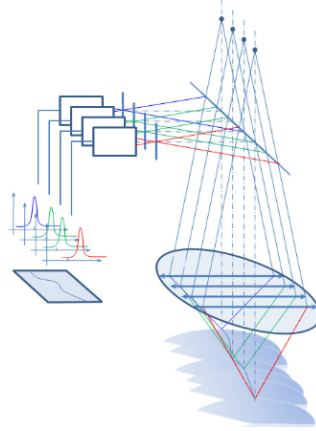


Figure 2.6: MPLS180 measurement principle

The sensor has for each measurement point a white pinhole light source which is directed to the measurement surface through a common lens with a big chromatic aberration. The chromatic aberration introduces a difference of focal length for different wavelengths of light. Normally for an optical instrument the chromatic aberration is more or less present, but not desired. The sensor uses this effect to measure the depth. Due to the chromatic aberration only one wavelength out of the white light reflected by the media is in focus. For each measurement point, the incoming frequency is sent on a CCD Image sensor through an optical grid. This grid is spitting the incoming light into its spectral components. The wavelength which is in focus creates a intensity maximum on the CCD-Sensor. This information is converted into a depth information.

Table 2.1: MTLS180 specifications

Line length	1.8mm
Number of points	180
Dist. betw. 2 pts	10 $\mu$ m
Spot size	2.8 $\mu$ m
Meas. range	0.35mm
Resolution	0.125 $\mu$ m
Accuracy	0.5 $\mu$ m

## 2.3 Dictabelt

### 2.3.1 The dictabelt media

The Dictabelt was a recording media introduced by the American Dictaphone company in 1947. This media consists of a thin plastic belt, made of vinyl looped on itself, with the following dimensions:

Table 2.2: Dictabelt dimensions

Dimension	Value [inch]	Value [mm]
Width	3.5	88.9
Thickness	0.005	0.127
Circumference	12.0	304.8

To be recorded or played the belt need mechanical support provided by the dictabelt machine or by a cylinder for the 3D-scan. The belts can be stored or mailed in a space-saving flat position, without the mechanical support. But if they were stored too long in a flat position, the folds can become permanently and a click can be heard twice per rotation if they are played.

This is the case for recordings taken during the assassination of president Kennedy in 1963, which should be restored after this Project.



Figure 2.7: Empty Dictabelt

For the dictabelts the sound is modulated into the groove vertically, in contrast with their predecessors the wax cylinders, which are modulated vertically. The surface is not only of noisier than the surface of the gramophone discs, but also not flat. As the needle is embossing the groove rather than cut, two peeks are created out of the material pushed out of the groove, by the recording needle.



Figure 2.8: Modified dictabelt recorder used during tests

There were two different kinds of dictabelts:

#### **15 minutes Belt**

These are the standard dictabelts and turns with a speed of 42 RPM.

#### **30 minute Belt**

These belts had only few success and were rarely used. To obtain the capacity of 30 minutes the belt was turning almost half as fast with 22 RPM. Unfortunately the sound quality was lower due to the reduced speed. The records of the Kennedy assassination are 30 minutes Belts

[4][5]

### 2.3.2 Restoring dictabelts

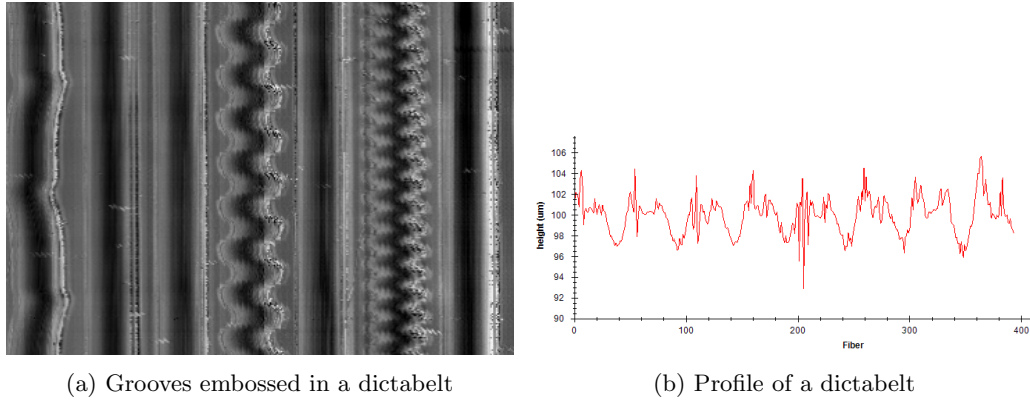


Figure 2.9: Surface and profile of a dictabelt

The dictabelt recordings are a difficult media to restore for two reasons:

The groove is embossed into the plastic media with a low depth of only several  $\mu m$ . The groove is not regularly and noisy, the shape of the groove changes continuously.

The surface is not flat, as it is the case for the gramophone discs, due to the embossing of the groove. If they would be flat they could be scanned with the 2D-Probe, which allows higher resolution and a faster scan.

Both effects can be seen in the figure 2.9.

## 2.4 Scan Process

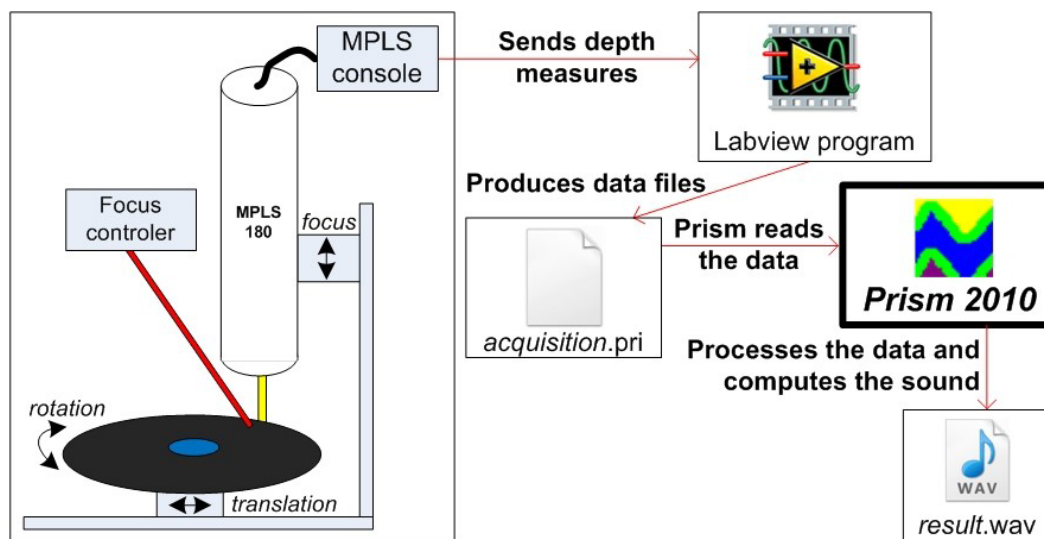


Figure 2.10: Measurement bench to scan discs [6, p. 7]

The dictabelts are scanned with the 3-D MPLS180 Line probe. To scan the dictabelt and any other media, three different high precision movement stages are required. The precision of these stages is really important, because distances in order of magnitude of  $500nm$  are measured. One rotation stage turns the media, a focus stage keeps the probe in the measuring range to the media, and the translation stage which controls the position on the media are required. There are two different scan benches present at the lab. The first one is used to scan medias like discs, where the probe is positioned in the same direction than the rotation axis. The second one is used for medias like cylinders where the probe is directed perpendicular to the rotation axis. For the dictabelts the bench for the cylinders is used.

The bench and the probe are controlled by a Labview program which creates as output the pri-file. This file contains the depth information of the scan in a floating point format. As those files are really large ( $\approx 0.5$  upto 2 GB), they are stored in a binary format to save memory space.

The probe has 180 measuring points, with a spacing of  $10\mu m$  between each. As result of this a strip of 1.8mm of the media can be scanned at the same time. After a completed rotation the translation stage moves the probe or the media by 1.8mm for the next scan. There is also an option to take multiple passes on the same position on the media. Each pass is slightly shifted by a fraction of the spacing between the fibers to reduce the distance between the data points. Usually 4-Pass is used for the dictabelts, which results in a vertical spacing between the points of  $2.5\mu m$ .

The data points often corresponds physically not to a square. The physical spacing in X-direction is given by the spacing of the fibers of the probe and the number of superposed scans:

$$\Delta_x = \frac{10\mu m}{n_{pass}} \quad (2.1)$$

The physical spacing between the points in Y-direction is given bi the circumference of the media (not constant for discs), and the number of samples per revolution. In PRISM the parameter height is containing information of samples per revolution. The spacing can be calculated with the following formula:

$$\Delta_y = \frac{2 \cdot r \cdot \pi}{samples_{rev}} \quad (2.2)$$

The sound is afterwards extracted out of the pri-file with the PRISM software.

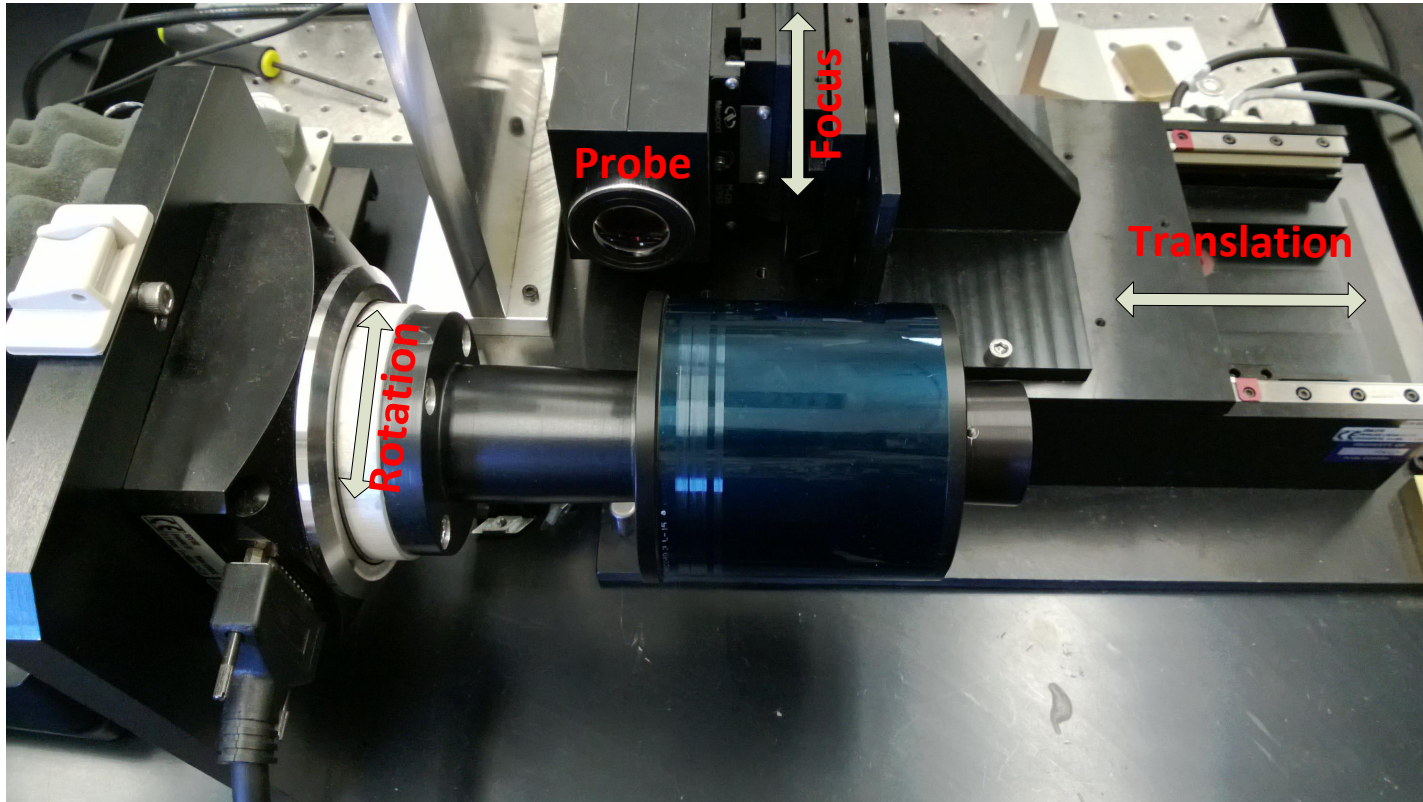


Figure 2.11: Scan of a dictabelt on the cylinder scan bench with the MPLS180 probe



## 2.5 PRISM

The PRISM software is used to extract the sound out of the pri-file created by the Labview code which scans the media.

### 2.5.1 Sound extraction process



Figure 2.12: Flowchart of the PRISM software

The Sound is extracted from the pri-file in the following steps:

#### Load Image

The Load Image methods are opening the pri-file, and converting it into the internal Raw image. Also the parameters that are stored in the header of the file are loaded into the internal structures, for example the height and the width of the scan, and the lateral offset between the passes.

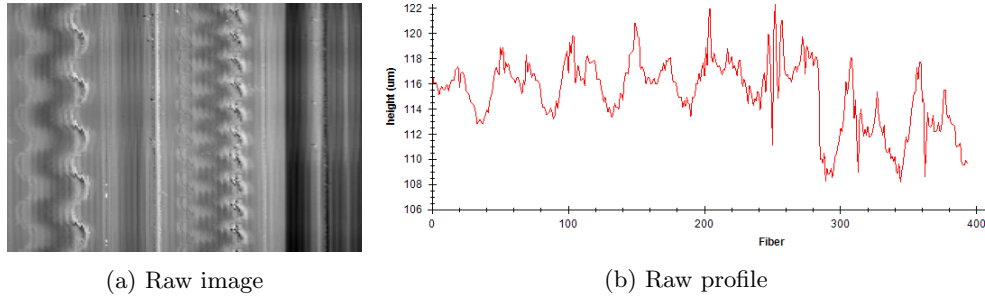


Figure 2.13: Image and profile of Raw image

#### Match pass

This optional Step corrects scan errors, mainly due to a imperfect perpendicular alignment of the probe to the surface of the media. If the alignment is imperfect, the average depth of the image is not constant, a slope over the grooves can be seen. So those methods are normalizing the average depths over the scan region. They are especially useful if the media was scanned with multiple passes, to normalize the average height of the different passes, which are merged together by the PRISM software. The algorithm does also correct outliers in the Raw image. The results of the Matchpass are stored in a separate internal image. More information about the different Matchpass algorithms can be found here[7, p.41-45].



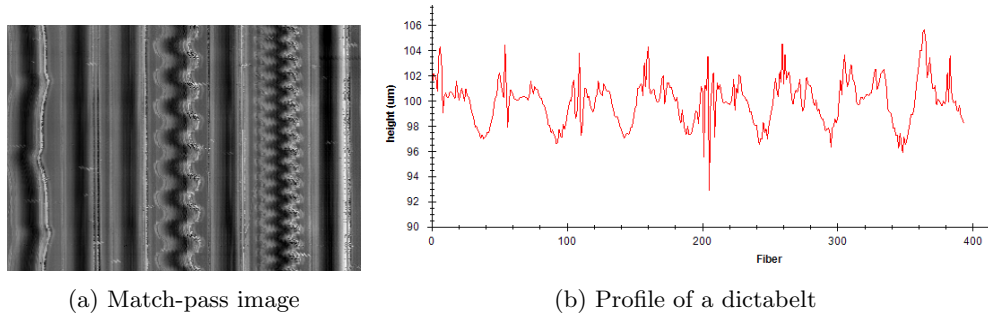


Figure 2.14: Image and profile of Match pass (Match ALU)

### Tracking

The automatic tracking algorithms are detecting the center of the grooves in the Raw image or in a Matchpass image if it does exist, and follow them. There are also a manual and an interactive mode, in which the user indicates the position of the groove. Usually they are used for media in a bad condition, on which the automatic algorithms fails. The tracking is used by the following Processing algorithms to have an approximate estimation of the position of the groove.

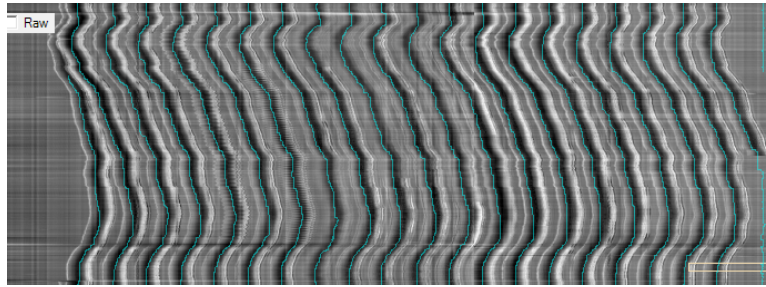


Figure 2.15: Tracked groove on a dictabelt

### Process

The most Processing algorithms are fitting the points close to the tracked groove center to a shape similar to the groove shape. The center of the groove is considered to be identically to the center of the fitted shape. The sound information is enclosed in the derivative of the groove positions. The derivative algorithm developed during this project is using an other approach to extract the sound. It is calculating the variation of the depth for each pixel in the groove for a variation in the time direction. Each of these variations for each pixel contains an approximation of the local sound at this position. With statistics this information is merged together to get the local sound.

### Create Wav

These methods are differentiating the position of the groove if a fitting Process is used. If the derivative algorithm is used this step is skipped. Afterwards the derivatives are written together with a header to the wave-file.

## 2.5.2 Interface

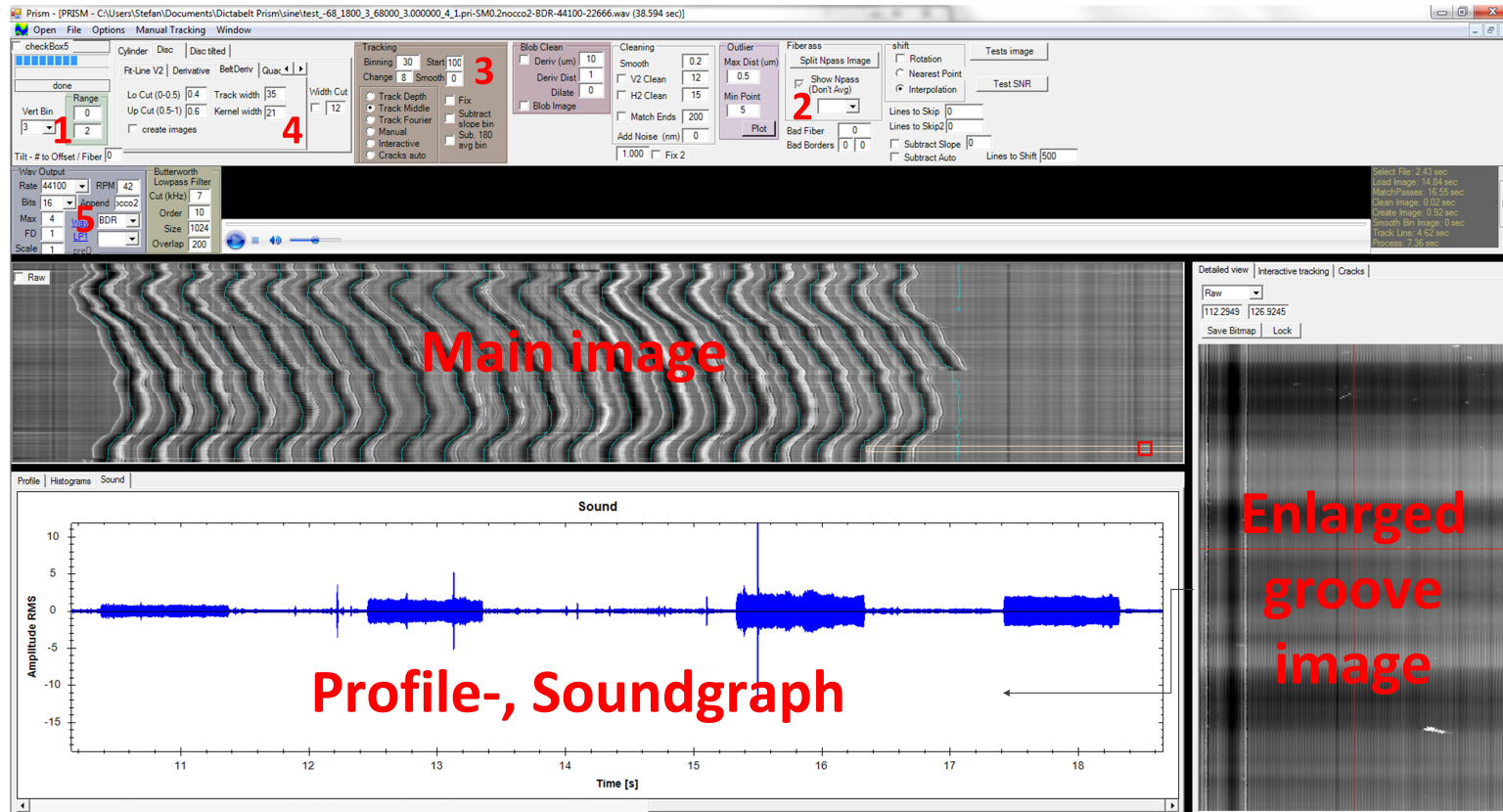


Figure 2.16: Interface of PRISM:  
 Legend: 1.LoadImage param. 2.MatchPass param.  
 3.Tracking param. 4.Process param. 5.Wave param.

## 2.6 Sound Forge

The results of the different algorithms were compared with Sound Forge Pro 10.0. This software is a professional sound post-processing tool made by Sony. As it wasn't the goal of the project to do post-processing only a few functions of this tool were used:

### Filtering

The ReaFIR Plugin which implements an FIR filter was used to cut the frequencies outside of the used spectrum. The advantage of a FIR filter is, that he has no feedback of its output. This allows to create filters with a linear phase over the frequency spectrum, which creates an equal delay for all frequencies.

The Filter can be set by shaping its frequency response by adding points to the spectrum.

### Volume

The volume was adjusted to compare the amplitudes.

### Spectrum analyzer

The built in spectrum analyzer was used to compare the different results.

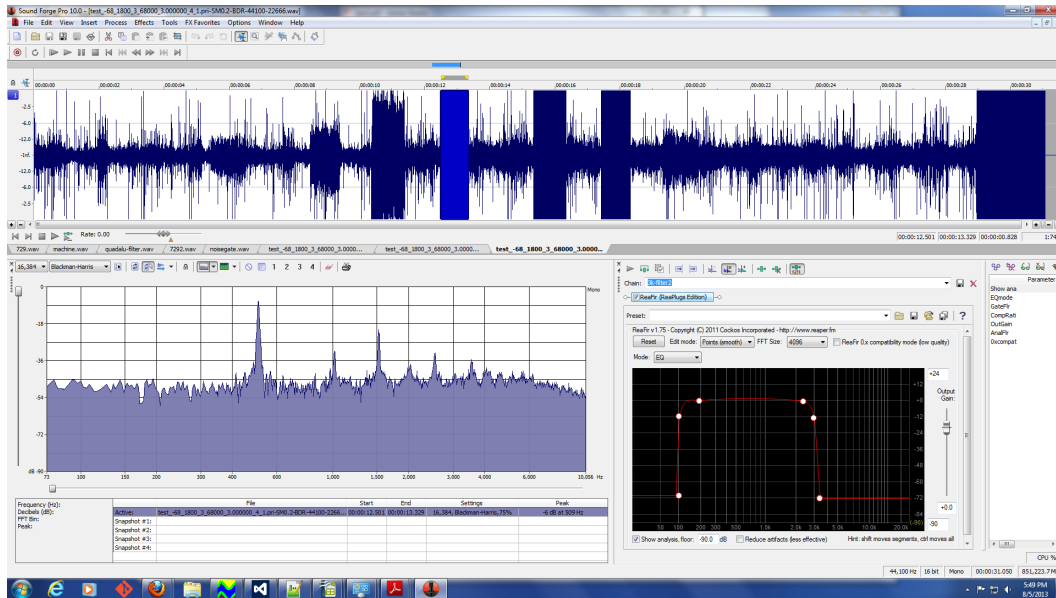


Figure 2.17: Screen shot of Sound Forge

## 2.7 Statistics

In the algorithm statistic calculations are made, the theory behind is included in this section.

### 2.7.1 Gaussian distribution

#### Definition

The gaussian distribution, also called normal distribution is an theoretical approach commonly used in statistic to describe the expected amount of errors in a set of data. To calculate the distribution a certain number of (random) samples, which are considered to be distributed normally, out of the whole set is evaluated.

The distribution is defined by:

$$P_{(x)} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (2.3)$$

The formula 2.3 contains two parameters:

- $\mu$  is a horizontal shifting constant.
- $\sigma$  is the standard deviation, the higher the standard derivation is the slower the Gaussian decreases to zero. It means that there are more difference between the samples.

[8]

The horizontal shifting constant, which correspond to the arithmetic average can be calculated straightforward out of the samples:

$$\mu = \bar{x}_{arith} = \frac{1}{n} \sum_{k=0}^{n-1} x[k] \quad (2.4)$$

The standard deviation is not that easy to calculate, usually this formula is applied:

$$\sigma = \sqrt{\frac{1}{n-1} \sum_{k=0}^{n-1} (x[k] - \bar{x}_{arith})^2} \quad (2.5)$$

The expression  $n - 1$  is due to the bessel's correction of the standard deviation [9]

The problem is if for example if the standard deviation should be recalculated on the last  $n$  samples for each measure. So for each pixel the sum under the square-root must be recalculated, because the average  $\bar{x}_{arith}$  changes. A formula in which for each measurement the oldest value of the sum is replaced by the new one, instead of recalculating improves the performance considerably. So instead of using the formula 2.5 the formula 2.6 was applied.

$$\sigma = \sqrt{\frac{1}{n-1} \left[ \left( \sum_{k=0}^{n-1} x[k]^2 \right) - \frac{1}{n} \left( \sum_{k=0}^{n-1} x[k] \right)^2 \right]} \quad (2.6)$$

The formula 2.6 allows to calculate the standard deviation by using two sliding sums, the sum of the samples and the sum of the square of the samples, thus the sums don't have to be recalculated, just to be updated. The only issue on this formula is that it is less numerical stable (see definition in chapter 2.8). The worst case would be, that the value under the square-root became negative. This is due to the rounding errors, if the sums are calculated with a limited precision (floating point). To avoid exceptions the algorithm must set negative values below the square-root to zero. [10]

### Gaussian Filter

A derived application of the Gaussian distribution is the Gaussian filter. It is a common image processing task to low-pass filter the image with a Gaussian kernel to remove high frequency content which is usually noise.

Previous experiences showed during the IRENE project, that a filtering in y-direction in the image space, which corresponds to the time direction in the sound space, is not useful. The same results can be obtained by filtering the sound after processing with less calculations. So the image is only filtered in x-direction with a one dimensional filter.

The filter kernel of a with the width of nine can be calculated as followed:

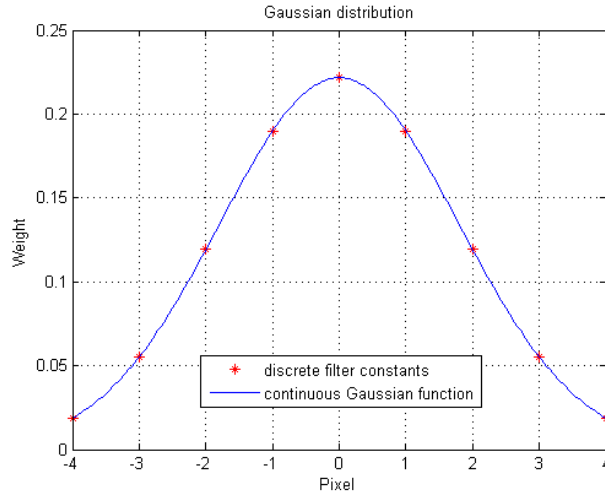


Figure 2.18: calculated with the formula 2.3,  $\sigma = 1.8$ ,  $\mu = 0$

Table 2.3: Weights of Gaussian Filter

Index	- 4	- 3	- 2	- 1	0	1	2	3	4
Weight	0.0188	0.0553	0.1196	0.1899	0.2216	0.1899	0.1196	0.0553	0.0188

In this case the pixel at the index 0 of the filtered image would take the following value:

$$NewPixel = \frac{\sum_{k=-4}^4 Pixel[k] \cdot Weight[k]}{\sum_{k=-4}^4 Weight[k]} \quad (2.7)$$

The formula 2.7 is applied for each pixel of the image. On the border of the image the non existing pixels are ignored, so the limits of both sums are truncated.

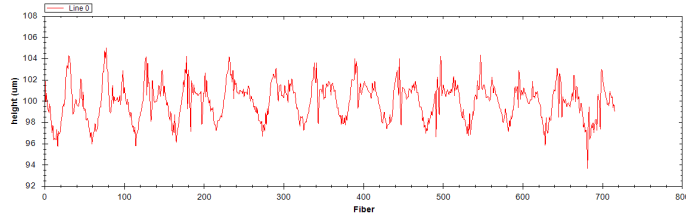


Figure 2.19: Sample of the profile of the unfiltered image

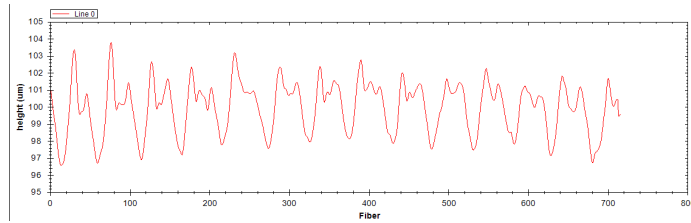


Figure 2.20: Sample of the same profile after the 1-D filtration

### 2.7.2 Histogram

A histogram is a way to visualize the distribution of a set of values. A histogram is composed of bins, which are all assigned to an interval of the distribution. Each bin contains the number of elements in this set which are located in this interval. With a histogram an approximation of the average, median, the standard deviation can be made.

In the developed algorithm often the distribution is not binned in the different intervals, only sorted. Once the values are sorted the outliers can be easily cut by throwing away a certain number of points on both ends of the distribution.

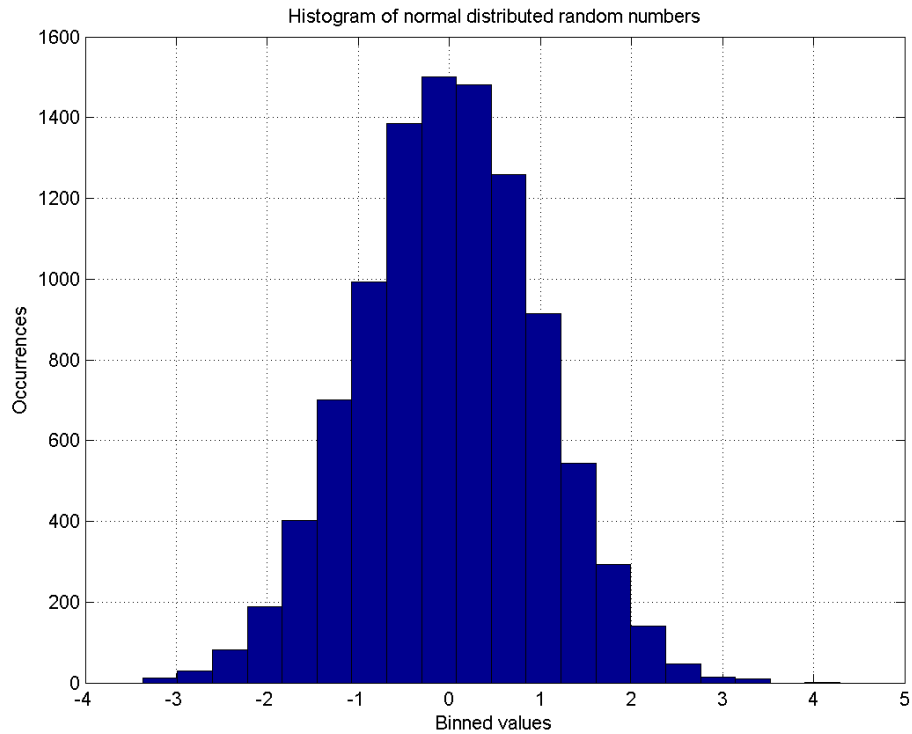


Figure 2.21: Histogram of normal distributed random numbers

## 2.8 Spline interpolation

To interpolate missing data, the spline interpolation is used. The usual polynomial interpolation calculates a polynomial with a degree of  $n - 1$ .  $n$  is the number of data points. So usually the polynomial interpolation function has a large order. This causes two problems:

### Runge's phenomenon

The Runge's phenomenon creates an oscillation which increases with the distance to the center of the interpolation. This effect can be seen in figure 2.22a.

### Numerical stability

The numerical stability is the insensitivity of a numerical algorithm on small variations on the inputs for example rounding errors. A polynomial with a high order is less stable than a polynomial with a lower order, so a small error on the input values results in a bigger error at the output for the high order polynomial.

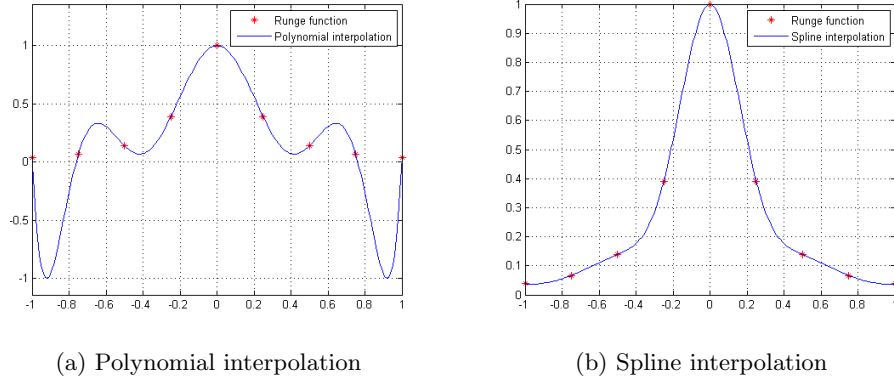


Figure 2.22: Different interpolation methods applied on the same set of points

Whereas the spline algorithm instead of calculate a high order polynomial for the whole range of data points, calculates a lower order polynomial for each interval between two points. At the position of each data point two functions are side by side. On this position they have not only the same value but also all the existing derivatives are the same. So for a cubic spine the value ,the first and the second derivative of both functions on at a data point have the same value.

It was been proved that the spline algorithm produces the interpolation with the smallest bending. This means that the interpolation creates a smooth transition between the points. Note the difference between the figures 2.22a and 2.22b [11][12]



## Chapter 3

# Specification of the Project

### 3.1 Goal of the Work

The intent of this project is to improve the quality of the restored sound of dictabelts. Up to now the sound is noisy and further analyses showed that a lot of distortions of the sound is present.

### 3.2 Functional specification

The work is separated into different Tasks:

#### **Improvement of the Interface of the Prism Software**

The problem up to now has been that the source of the noise could not be identified. The goal is to improve the user interface to facilitate the detection of the source of noise.

#### **Interpolate corrupted parts of the Media**

Due to long storages in the folded position the folds became permanent which can be heard with two clicks per rotation. The goal of the Interpolation is to detect these folds and other mechanical defects, like cracks in the belt, and interpolate the corrupted data with the samples before and after the failure.

#### **Implement a new derivative Algorithm**

The sound was extracted up to now for the dictabelts by finding the groove center with a fitting of the points orthogonal to the groove direction. This task should implement and evaluate an optimized version of the derivative algorithm. It creates sound information for each fiber with the vertical difference between two samples. The extracted sound is created by applying a weighted average on the information of the fibers into the groove.

#### **Correct distortion made by recording device**

The diploma work of Mr. Lutz and Yerly (2005) showed that the sound was distorted by the dictabelt machine. If this distortion could be described mathematically its

inverse could be applied to the extracted sound canceling its effect out as much as possible.

#### **Apply results to other types of media**

The improvements made with the four points above should be applied and evaluated on other types of media like gramophone discs or cylinders.

### **3.3 Operational specification**

#### **3.3.1 Improvement of the Interface**

The following improvements must be done:

##### **Modification on the graph who plots the sound**

Add a cursor to the graph that will show the actual position of the media-player. By clicking on the graphic the player will be set on this position. The unit of the X-Axis must be changed from samples to time (seconds).

##### **Synchronize the cursor with main picture**

The cursor must be synchronized with the position on the main picture which plots the whole recording media.

##### **Modification on the enlarged picture of the groove**

A button must be added to lock this picture on the current position of the player.

Fix the following bugs:

- The centers of the grooves aren't drawn correctly
- The crosshair doesn't point to the right position if the border of the main picture is enlarged.

#### **3.3.2 Interpolation of corrupted parts**

This task consists of the following points:

##### **Analysis of a folded dictabelt recording**

Take an old folded dictabelt and scan it, or use a scan already produced, and analyze the scan with the Prism software.

The length of corrupted data is important, along with any other useful proprieties, to detect a crack or fold and to interpolate it.

##### **Analysis of interpolation and fitting algorithms**

Get an overlook of the existing algorithms and select the most appropriate.

##### **Evaluate the algorithms**

Test the quality and the performance of the selected algorithms on Matlab or a similar tool with data extracted from Prism. Evaluate if the algorithm should be applied on the image or on the extracted sound. Choose the most appropriate algorithm.

**Implementing the algorithm**

The algorithm must be implemented in C# and integrated into the Prism software.

**Evaluate and improvement the algorithm**

Compare the results to the previous results, using the new algorithm to measure the performance and quality.

**3.3.3 Extract data with derivative method**

This task consist of the following points:

**Analysis of existing derivative algorithm**

Analyze the existing derivative algorithm and define the necessary modifications to apply it to dictabelts.

**Define structure of the method**

Define the method to calculate the sound for each fiber. Find an appropriate weighting of the samples.

**Implement method**

The method must be implemented in C# and integrated into the Prism software.

**Evaluate and improve the results**

Compare the results with the existing methods, particularly the Quad-Alu and the existing derivative Method. Improve the method if necessary.

**3.3.4 Correct distortion made by recording device**

This task consists of the following points:

**Analysis of former results**

Analyze the results obtained by Lutz and Yerly (2005). Get more information if necessary.

**Mathematical characterization**

Find a mathematical function of the AGC and the filter.

**Experimental evaluation**

Apply the inverted function to the extracted WAV file, with Matlab or a similar tool. Evaluate and compare the distortion before and after the correction.

**Implement in Prism**

Implement the obtained transfer function to Prism.

**Evaluation of the Implementation**

Evaluate and compare the distortion before and after the correction.

**3.3.5 Apply results to other types of media**

Apply the results to the other types of media, especially to the aluminum discs, which have a similar physical surface structure.

## Chapter 4

# Improvement of the PRISM Software

The first two week the PRISM software were analyzed, and improved. The goals of these modification were to improve the user interface to be able to detect the sources of noise easier.

### 4.1 Add a cursor to the soundgraph

On the bottom of the GUI the sound in function of the time is plotted. On this graph for example the clicks are showed up as short peaks with a high amplitude. But up to now this plot wasn't really helpful to find bad points, because there was not really a reference to the images.

A cursor which indicates the actual position of the sound played with a Windows Media Player (WMP) plugin inside of PRISM was added. The soundgraph window is handled by a library called ZedGraph which can create all kinds of 2D charts. [13] The ZedGraph library is quite easy to use, but also not very fast. To add the cursor was a straightforward task, also the calculating of the position by using a method of the WMP plugin, which returns the time of the current playback. But an issue was the refresh rate of the position of the cursor, which was slow ( $\approx 1\text{Hz}$ ). After an analysis of the code the problem was that even if the cursor and the sound are two different curves on the same chart, the library recalculates the whole chart if one of both has changed. And as the sound curve was containing all the samples extracted from the media, ZedGraph had to recalculate a curve with hundreds of thousands points, even if the size of the window is only several hundreds of pixel wide. To speed up the plot of the cart, the number of samples was reduced by hundred by calculation the root-mean-square average over the amplitude of hundred values and plot the averages instead of the samples. After this change the cursor was sliding fluidly while the sound was playing.

Some minor changes were also done:

#### **Set WMP position**

It is possible now to click in a position of the sound graph or the main image, and the WMP skips to this point.

#### **Y-Axis of sound graph**

The y-axis unit was changed from sample number to time in seconds.

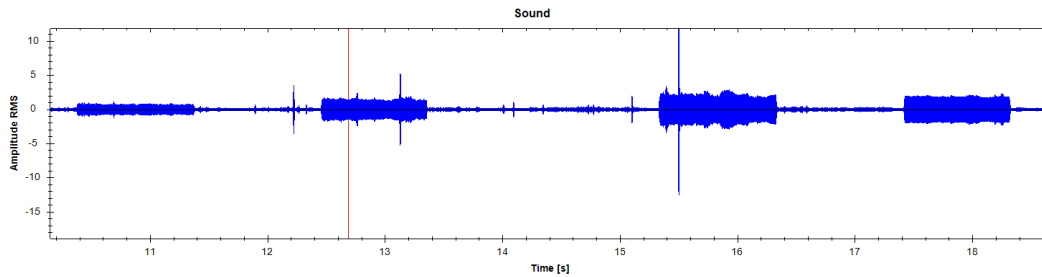


Figure 4.1: New sound graph with added sound cursor

## **4.2 Enlarged groove picture**

Two changes were made for the enlarged groove picture on the right side of the GUI:

#### **Add button to lock crosshair on actual position**

This button is located above the enlarged groove position on the right side of the GUI. If the button is clicked the crosshair can't be moved manually but set automatically to the actual position set by clicking in the main image with all the grooves or by clicking on a position in the sound graph.

#### **Fix bug of tracked lines**

On the enlarged groove picture the tracked groove center wasn't draw in the right position if the pri-file contains the data of an n-pass scan. So the drawing position wasn't corrected.

## Chapter 5

# First implementation of the derivative BeltDeriv algorithm

Most algorithms of PRISM are working on the same principle to extract the sound of the raw data.

They use a tracking algorithm to estimate the position of the groove, and try to fit the points, in the estimated groove, into a approximation of the groove shape to get a more accurate center of the groove.

For the dictablets this approach lead to a noisy output signal with many clicks. This noise is at least partially introduced by the MPLS Probe, due to some imperfection of the match between the measurement fibers. The goal of the derivative algorithm is to extract the sound information by measuring the height variation of one fiber positioned over the groove from one line to the next.

The basic idea of this approach is to compensate the mismatch between the fibers by analyzing not the effective height delivered of the fiber, but the variation of the height of the same fiber between two Points.

If the groove is moving horizontally the derivation of the height between two lines is positive on the left side of the groove, and negative on the right side of the groove. If the groove is moving in the other direction the signs are opposite. Because it is possible to extract a sound signal on each pixel located into the groove there will be a lot of redundancy.

This redundancy is used to correct noise on the sound signal. The algorithm can be configured to exploit this redundancy by passing parameters trough the GUI.

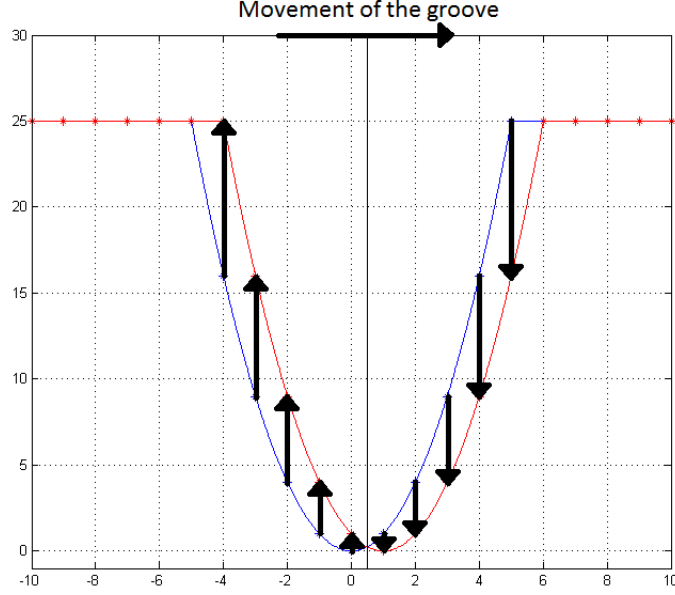


Figure 5.1: Groove movement: Influence of horizontal shift of the groove on the derivative in time direction

The Kernel width parameter gives the width of the one dimensional Gaussian Low pass filter applied on the input image to create the 1D filtered image. The functionality of this filter is explained in the chapter 2.7.1.

The sound modulated into the groove corresponds to the derivative  $\frac{dx}{dy}$ , but the derivative we are calculating with the difference between two lines of the same fiber corresponds to  $\frac{dz}{dy}$ . To obtain the sound we need to divide this derivation by the local slope of the profile of the groove ( $\frac{dz}{dx}$ ).

$$sound(y) = \frac{dx}{dy} = \frac{\frac{dz}{dy}}{\frac{dz}{dx}} \quad (5.1)$$

The formula 5.1 is applied for each pixel lying into a certain distance the tracked groove center. This distance is determined by the Track width parameter on the Beltderiv window of the GUI (see figure 5.2). The value correspond to the total number of derivatives taken in consideration to extract the sound. If this parameter is set to 21 for example the average derivative is calculated by using the ten fibers on the left, the central fiber and the ten fibers on the right of the tracked groove center. If the parameter Track width is not an odd number it will be internally incremented by one from the algorithm.

Once all the derivatives in one groove position are calculated, the sound at this position is obtained with an average of this values. It is possible to take the median of all the derivations, or by applying a weighted average on all the samples. The weight is determined by the local slope of the groove shape ( $\frac{dz}{dx}$ ). The steeper the slope is, the bigger is the derivation  $\frac{dz}{dy}$  for a given groove movement, which result usually in a better signal to noise ratio.

The outliers can be filtered out by the lower and upper cut parameter. The lower cut parameter controls the relative number of the points thrown located on the left side of the distribution and the upper cut the number of points thrown on the right side of the distribution. If the lower cut and the upper cut are both set to 0.5, the algorithm uses the median value. At the beginning a new tracking algorithm was also implemented to find the

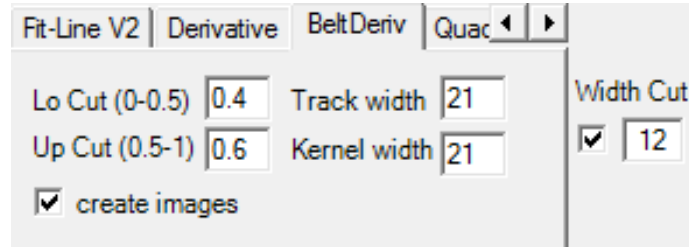


Figure 5.2: GUI of the Belt Deriv algorithm

approximate groove centers by locating the minimal values of the 1-D filtered image. But this algorithm was not as reliable as the existing ones, so in the final version the derivative algorithm uses the existing tracking algorithms.



# Chapter 6

## Validation

To validate the quality of the algorithm a dictabelt was recorded, containing sine waves with different frequencies. This belt was scanned afterwards with the 3D probe and the sound extracted with the derivative algorithm. To compare the result the belt was also played with the dictabelt recorder. Similar measurements were made by Noe Lutz and Michel Yerly back in 2005. [14]

### 6.1 Hardware

The measurements were made on an automation rack, called NI PXI-1031 made by National Instruments.

The rack contains an interface card (NI PXI-8331) to communicate with a computer. This card is connected to a PCI-Adapter (NI PXI-MXI-4) inserted into the computer over a copper cable. The rack is also equipped with a data acquisition card (NI PXI-4461) which contains two analog inputs and two analog outputs. Those inputs and one output are connected to the dictabelt recorder in two different ways, depending whether the dictabelt should be recorded or played back.

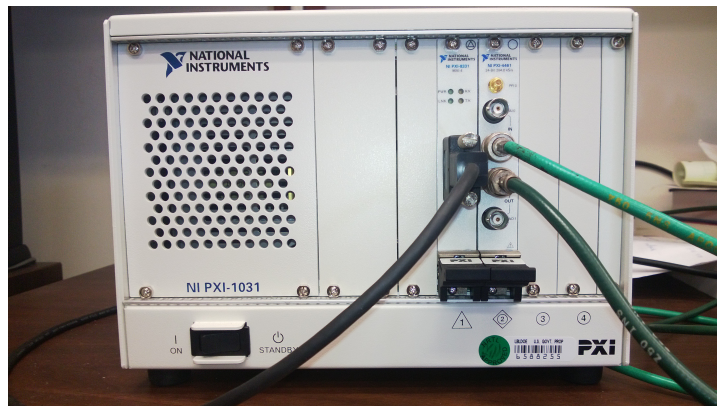


Figure 6.1: NI Automation Rack (NI PXI-1031)

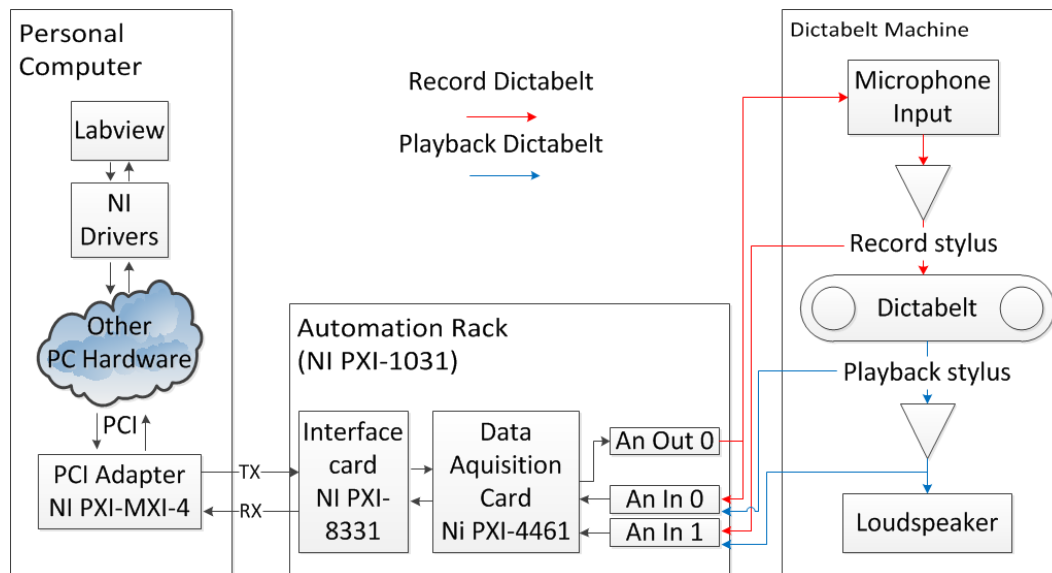


Figure 6.2: Schematic of the hardware used to Record and Playback dictabelts

## 6.2 Software

The automation rack is controlled by a Labview program that creates sine-waves on the analog output of the data acquisition card.

The Labview software is separated into two different parts, the sweep generator which controls the analog output of the data acquisition card, and the recording stage which records the voltages on the analog inputs.

The Labview program is available in the annex (see chapter 12.2).

### 6.2.1 Sweep generator

The amplitude and the frequency can be swept in linear steps for the amplitude and the frequency. For the frequency a pseudo logarithmic sweep also exists. The pseudo logarithmic sweep can be configured by choosing the decades through which the generator should sweep, and the steps in a decade. If for example the steps 1, 2, 5 are chosen, and the first through the third decade, the generator creates the following frequencies:

Table 6.1: Frequency sweep example

Step	Frequency [Hz]
1	10
2	20
3	50
4	100
5	200
6	500
7	1000
8	2000
9	5000

If silent parts between the different sine waves are desired zeros can be inserted between the steps.

### 6.2.2 Recording stage

The recording stage gathers the input values of both analog inputs and writes them into a stereo wave file.

## 6.3 Tests

The quality of the the algorithm was analysed with the following method:

The signal recorded was a series of sinusoids with a duration for each of one second. Between the sinusoids a gap with a duration of also one second was introduced. The frequency was sweeping between 10 Hz and 5kHz with a step of 1 ,2 and 5 per decade.

The test system was connected as described in figure 6.2 for the Record of dictabelts (red and black arrows only). The dictabelt recorder was recording the test signal created by the Labview program.

Once the belt was cut, it was scanned with the 3D-Probe. The from the Labview program, which controls the whole scan-stage, created a pri-file which contains now the depth information of the belt. This file is the input of the PRISM program.

The sound was extracted at this stage in three different ways.

Firstly the new derivative-algorithm which should be tested, secondly the existing QuadAlu-algorithm and finally the dictabelt recorder in order to compare the results.

The comparison was done with the Soundforge program. The following tasks were performed on the wave-files:

### **Filter**

The frequencies outside of the bandwidth of the dictabelt were cut with a FIR-filter with cut-off frequencies of 100Hz and 3kHz. With this filter the influence of the noise in the bandwidth of the sound is put in evidence.

### **Amplitude normalization**

Since the three algorithms do not produce the same volume, the volume was equalized for each of the sound-files.

### **Comparison**

The result was compared in time domain, in which the clicks, were showing up. As well as in the frequency domain. in which the noise and its distribution could be observed.

## 6.4 Results

After the first tests, the derivative algorithm produced a lot of white noise. The source of this noise is at least partially due to the selection of the points. Which are taken into account to calculate the output sound signal. Compared with the old QuadAlu algorithm the derivative algorithm has a lower signal to noise ratio for low frequency but a slightly better ratio for higher frequencies. On the figure 6.4 the spectrum of the 500 Hz sinewaves can be compared.

Although the Belt was scanned immediately after its recording, dust particles were detected on the depth image. They showed up as a large height variation. The top of the dust particles are often much higher as the groove itself. Those dust particles are blobs.

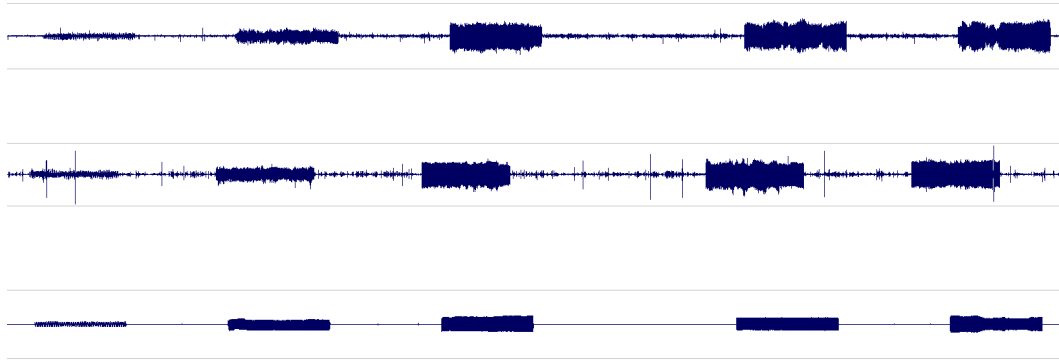


Figure 6.3: Comparison of the first results on the dictabelt for 500 Hz sinewave, created with BeltDeriv(top), QuadAlu (middle) and dictabelt machine (bottom)

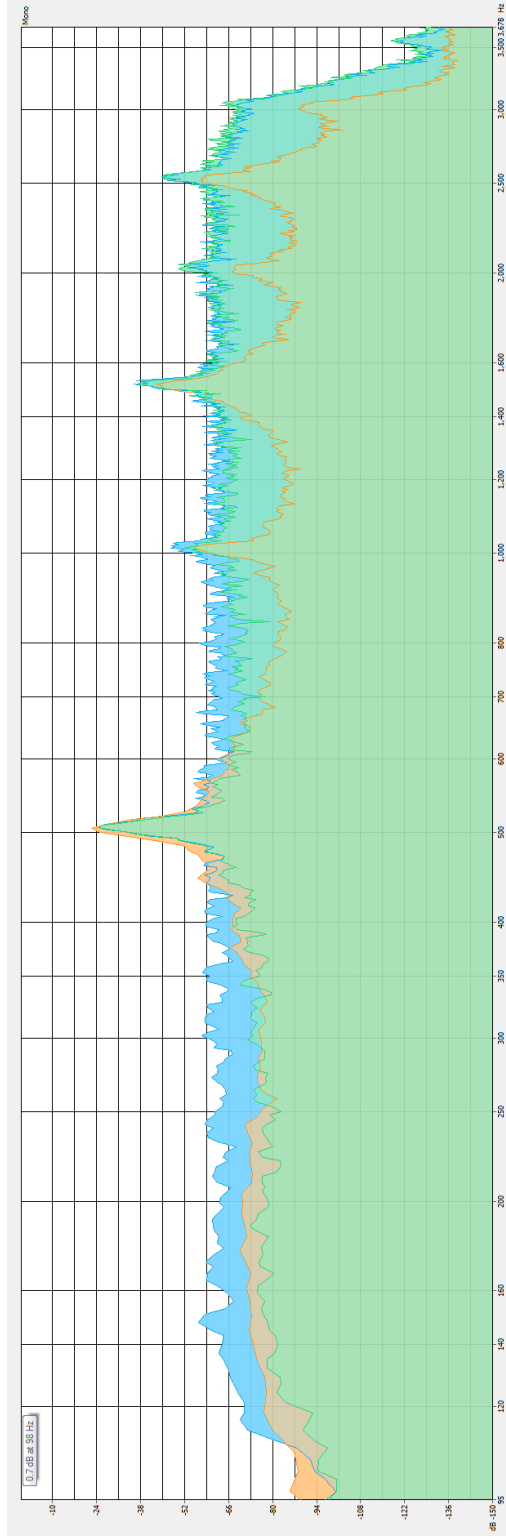


Figure 6.4: Comparison of Needle playback, QuadAlu and BeltDeriv algorithm.

■ BeltDeriv   ■ QuadAlu   ■ Dictabelt recorder

## Chapter 7

# Improvements in the algorithm

### 7.1 Separate the slopes

As the grooves are often not symmetrically one side of the groove usually got more weight than the other one. To avoid this effect an average derivative is calculated on each side of the groove. In the end, both derivatives are averaged together.

### 7.2 Detection and interpolation of blobs

As it was determined that there were a lot of blobs on the dictabelt which are creating clicks on the sound output. So in the BeltDeriv algorithm an option to detect and interpolate those blobs were implemented:

#### 7.2.1 Detection

The blobs are detected in two steps:

Firstly, in the algorithm the center of the groove is determined with the approximated groove center from the tracking algorithm as the start point. A more accurate groove center is obtained by following afterwards the slope of the groove down to the bottom of the groove. If the groove center moves too far from one line to the next the point is considered as a bad point. The threshold is not fixed, but calculated with the parameter Center Movement threshold times the standard deviation of the dynamic of the groove center from the previous points.

Secondly, the height of the groove center is compared to the average value of the previous groove centers. If the difference between the groove center and the average is larger than the parameter Center Height threshold times the standard deviation, this point is also considered as a bad point.

### 7.2.2 Interpolation

The bad points are interpolated with a spline interpolation. They aren't corrected in the image but in the sound. If a position of the groove was considered as a blob, instead of calculation a derivative and add it to the list of derivatives, just a not a number (NaN) is added. Once all the derivatives are calculated, the interpolation method is replacing the missing parts, indicated with NaNs, with the result of the spline interpolation of that section.

## 7.3 Vertical binning

The PRISM software has an option called vertical binning, if this option is chosen the pixels stored in the pri-file are reduced by the vertical binning factor. If the factor is set to two, each pixel value of one line is separately averaged with the pixel a line further to create together a new pixel. The principle of the vertical binning is a kind of oversampling to reduce the quantification error. The vertical binning factor is limited by the Nyquist-Shannon sampling theorem which says:

$$f_s \geq 2 \cdot f_{max} \quad (7.1)$$

The sampling frequency of the depth image can be calculated with the following parameters:

$$f_s = \frac{I_{height} \cdot v_{rot}}{vert_{bin}} \quad (7.2)$$

$I_{height} [Pixels]$  : number of lines, number of pixels in y-direction

$v_{rot} \left[ \frac{1}{sec} = \frac{RPM}{60sec} \right]$  : rotation speed of the belt

$vert_{bin} [Pixels]$  : vertical binning factor

The maximal binning factor can be determined with the formulas 7.1 and 7.2. The oversampling increases the scanning time, but usually the scans are anyway over-sampled for archival. This oversampling is intended for eventual better extraction algorithms in future or just to be sure that no features are lost.



## Chapter 8

# Final BeltDeriv Algorithm

This chapter talks about the BeltDeriv algorithm, and its utilization after all the improvements made on it.

### 8.1 Parameter and their effect on the algorithm

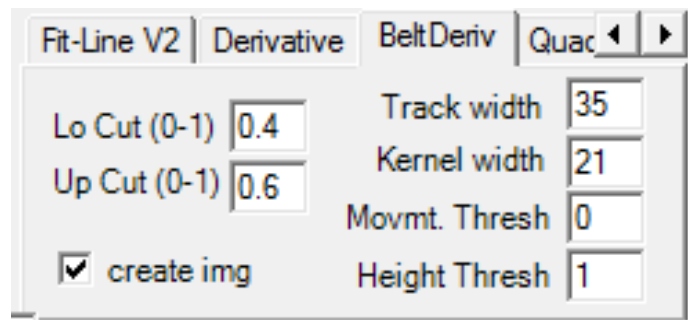


Figure 8.1: GUI of the final BeltDeriv algorithm

#### Kernel width

This parameter affects the 1D-filtered image, the larger the kernel is, there more pixels are used to calculate the filtered image. This increases the calculation time, but the filtered image gets smoother.

#### Kernel sigma

This parameter affects also the 1D-filtered image, the smaller sigma, the standart derviation is, there more weight is on the pixels in the center.

#### Track width

This parameter determines the maximal number of pixels taken in account to calculate the list of derivatives. The weighted average of the elements of this list is considered as

the sound information at this position. The algorithm takes all the pixels in account which are between the tracked groove center, and the top of the groove rim for each side, if it isn't limited by this parameter before.

#### **Lo cut and Up cut**

These parameters are the limits of derivatives taken in account to calculate the sound with the average of the derivatives in the list. The parameters take both a value between 0 and 1. The list is sorted by the value of the derivatives  $\frac{dx}{dy}$ . If Lo is set to 0.2 and Up to 0.7 the lowest 20 % and the highest 30% are excluded. The bigger the difference between Up and Low there more derivatives are used. If Lo is bigger than Up, no sound is created. If they have the same value, the closest value in the list is added in the sound array

#### **Height and Movement threshold**

These are the parameters for the blob cleaning. Movement is the sensibility of the movement of the center of the groove and Height is the sensibility on height variation of the groove center

#### **Create images**

If this check-box is checked the algorithm creates the weight image ( $\frac{dz}{dx}$ ), the value image ( $\frac{dx}{dy}$ ), and the error image (shows points which were considered as a blob. These images are help full for finding the right parameters, but increases the calculation time and the memory usage of PRISM

## 8.2 Flowcharts

### 8.2.1 Top level

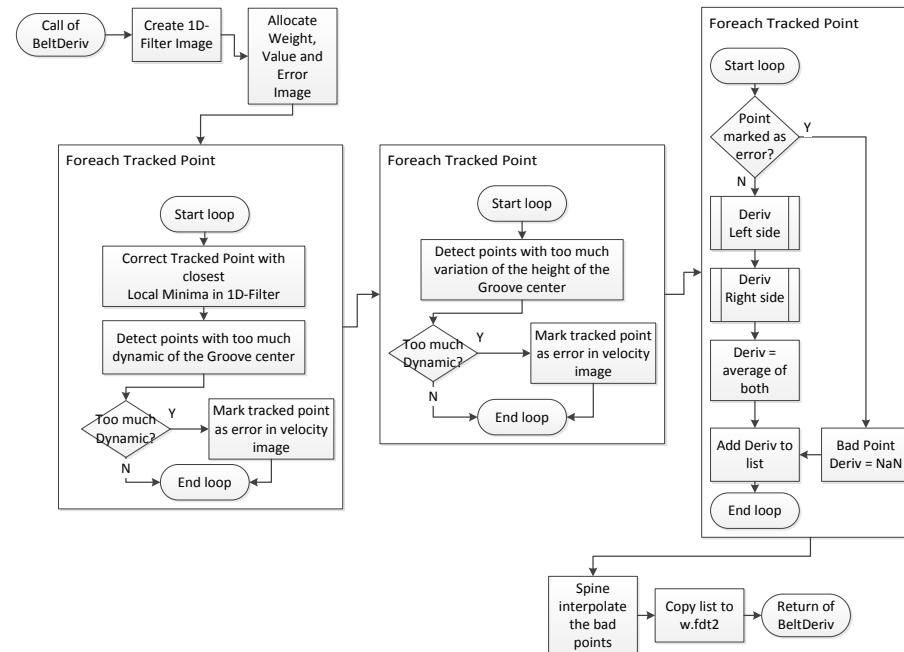


Figure 8.2: Toplevel of the BeltDeriv Algorithm

## 8.2.2 Calculation of the derivatives

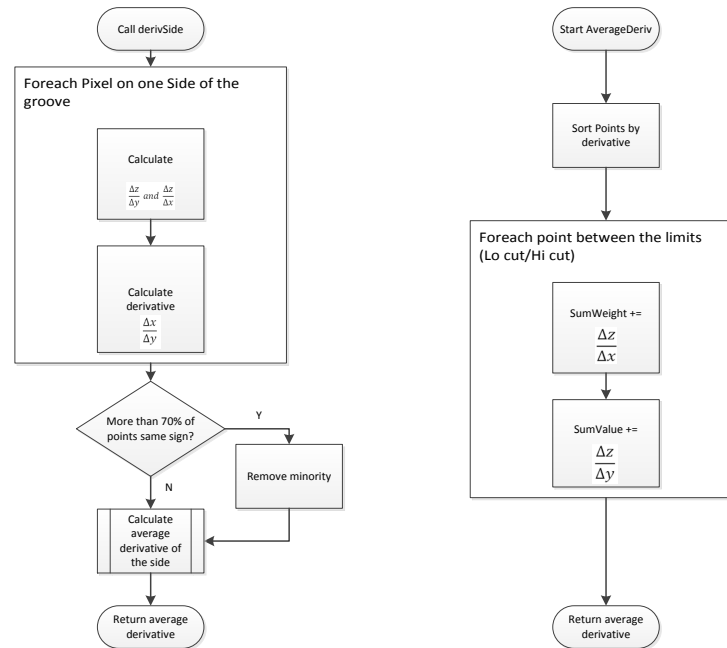


Figure 8.3: Calculation of the derivatives

### 8.3 Results

The same measure than in chapter 6.4 was applied. The difference between the first and the second algorithm aren't enormous. The final algorithm was around 3dB better.

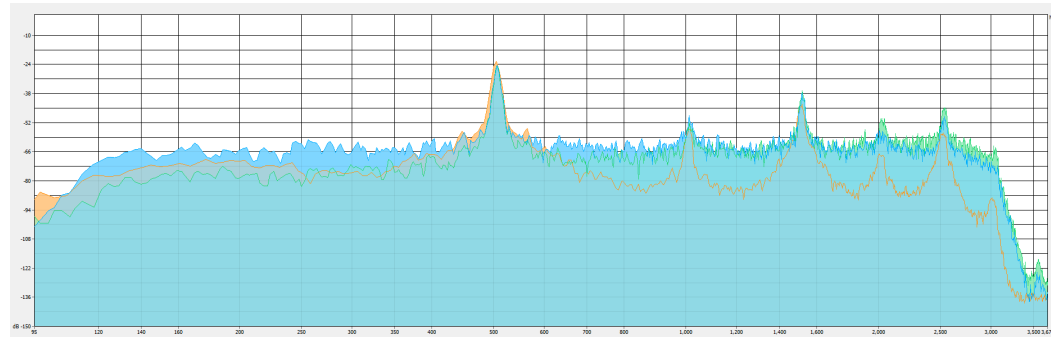


Figure 8.4: Comparison of Needle playback, QuadAlu and BeltDeriv algorithm.

■ BeltDeriv    ■ QuadAlu    ■ Dictabelt recorder

## 8.4 Get the right parameters

Even if the algorithm is robust relative to the settings of the parameter, to get an optimal result the following procedure should be applied:

### **Initial settings**

Activate MatchAlu. Check the create image box. Set both thresholds to 10 to neutralize them. Open a pri-file, and let PRISM process the file

### **Check the ImageLoad**

Modify the load parameters if necessary. (4PassFilp, No180Filp, etc.

### **Check the tracking**

Modify the tracking parameters if necessary, or perform a manual / interactive tracking.

### **Set 1D-Filter parameters**

Compare the profiles of the QuadAlu and the 1D-Filtered image, if the sigma and kernel width parameters are set to low the 1D-Filtered image has still some peaks. But if they are set too high the shape of the groove gets blurred out. Adjust the parameters, until the filtered profile became a continuous function, which preserves the depth of the groove bottoms and tops of the MatchAlu image.

### **Set the cut parameters**

This is the most difficult step. An optimum of both parameters with the lowest amount of white noise in the soundfile must be found.

### **Set the blob clean thresholds**

The user must know how a blow looks like, a sample can found in figure 2.4a on page 8. Compare the MatchAlu image with the error image, if there are too many bars which indicates a detected Blob are found on spots without blobs, the thresholds must be increased. If the blobs aren't detected the thresholds must be lowered. The short bars in the Error images are blobs detected by the height check, the long bars by the movement check.

### **Uncheck create image box**

The optimal settings should be found now. If a rerun of PRISM is necessary, for example to process a larger portion of the pri-file, consider unchecking of the create image box to increase performance of the algorithm.

## Chapter 9

# Test of BeltDeriv algorithm on Aluminum discs

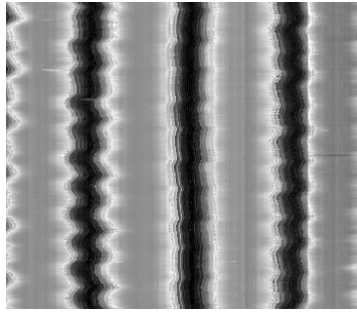
### 9.1 Aluminum discs

The aluminum discs have a similar physical structure than the dictabelts, so the algorithm was tested on this media. Up to now the sound of the dictabelts was extracted with the existing QuadAlu algorithm, which has been created for aluminum discs. The main difference for PRISM between both media is that the grooves of the alu disc is approximately 3 times deeper ( $\approx 15\mu m$  instead of  $\approx 5\mu m$ ) The tests were done on a scan made during the bachelor thesis of Alain Benninger in 2012.

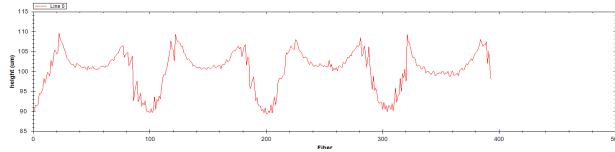
The rotation speed of those discs is 78 RPM which was a common speed back in the 1930, when they have been recorded. [7]



Figure 9.1: An aluminum disc of the Millman Parry collection [7]



(a) Grooves of an aluminum disc



(b) Profile of an aluminum disc

Figure 9.2: Samples of an aluminum disc

## 9.2 Results

The derivative algorithm was able to extract the sound out of the aluminum discs. The sound was less noisy than for the dictabelts, but the sound quality wasn't as good as the result of the QuadAlu algorithm.

The noise floor of the Derivative Algorithm is about 15 dB higher than the noise floor of the QuadAlu algorithm, which is a large difference. Apparently the QuadAlu algorithm doesn't work as good on all the discs. It should be tested if there are different results on other discs. Unfortunately due to a lack of time this wasn't possible during the project.

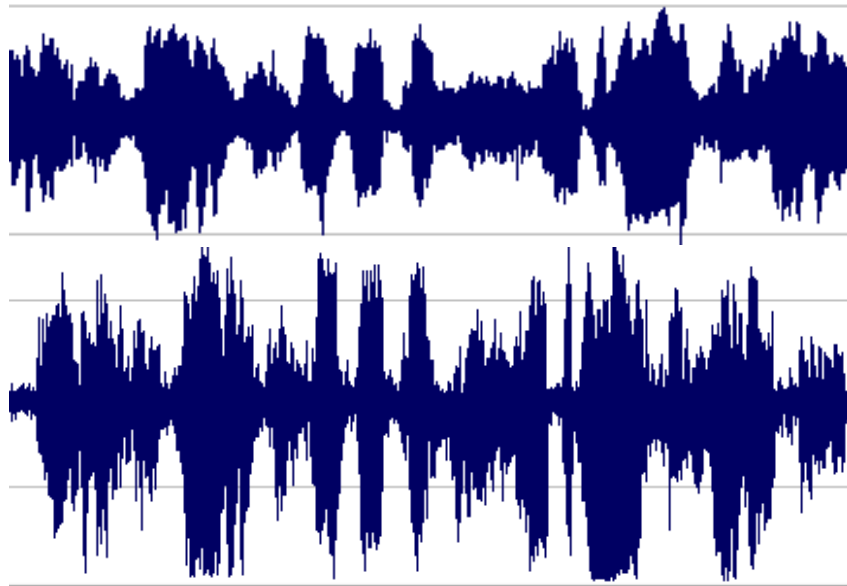


Figure 9.3: Comparison of the results on aluminum discs, created with QuadAlu(top) and BeltDeriv(bottom)



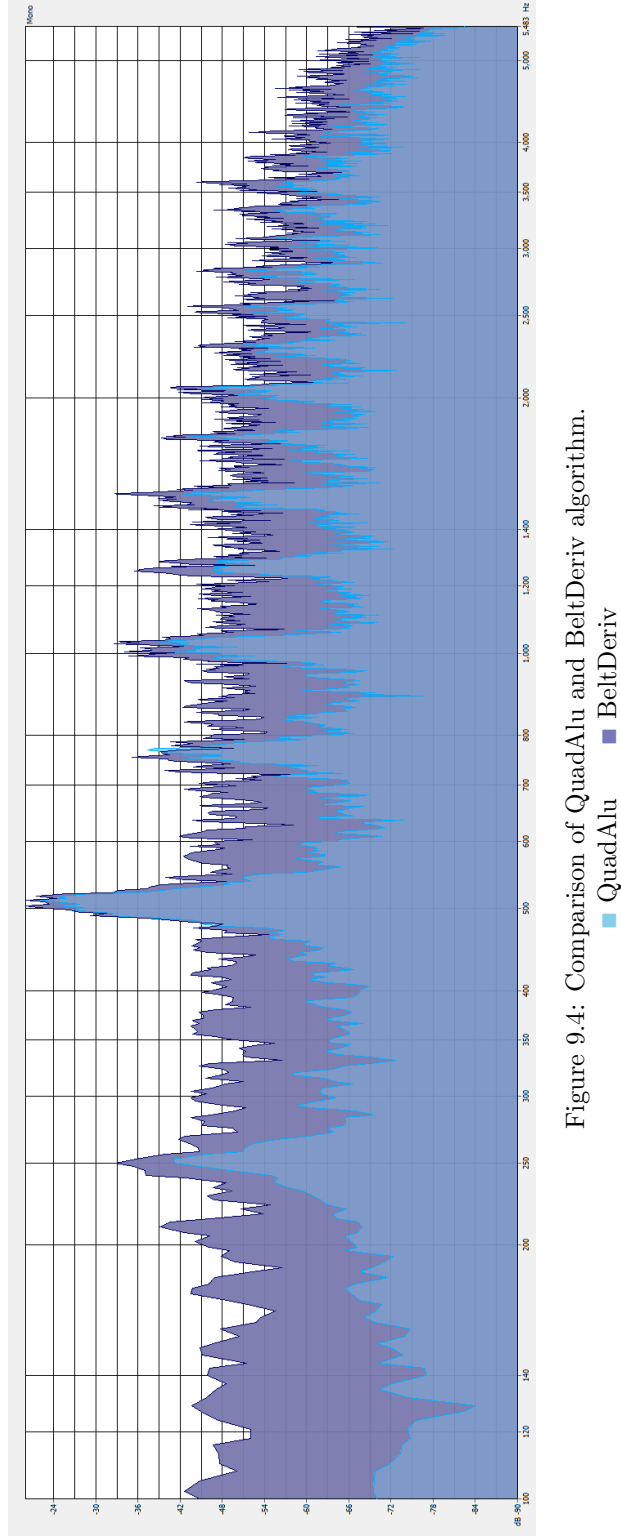


Figure 9.4: Comparison of QuadAlu and BeltDeriv algorithm.

# Chapter 10

## Conclusion

The goal of the project was to improve the signal to noise ratio of the extracted sound to be able to restore the dictabelts with the best possible quality, this result was only partially achieved.

### 10.1 Noise comparison

Amazingly while the development of the the algorithm, it was noticed that the sound could be extracted out of the raw data really fast, without correction, interpolation and the filtering of bad points, and even some bugs in the algorithm. Although this sound was quite noisy. But the further the development was progressing the harder was it to improve the quality of the sound. The noise created by the algorithm is principally white noise, which is distributed equally over the whole spectrum. The white noise is created by a lot of small errors, due to the choice of the good points, the measurement error of the MPLS180 probe and the position errors of the precision stages of the scan bench. As well as the irregular surface of the dictabelt itself. The QuadAlu algorithm in contrast produces more noise in the high frequencies than in the low frequencies. If both algorithms were compared together usually the derivative algorithm showed better results for frequencies higher than 0.5-1kHz. But the playback with a needle on the dictabelt recorder was still better. Especially it was the case for high frequencies where the signal to noise ratio was around 20 dB higher compared to the derivative algorithm. In the low frequencies the playback is comparable to the results of the QuadAlu algorithm. The errors of the QuadAlu and the derivative algorithm aren't correlated. An successful attempt in the Soundforge environment was made, to mix both soundfiles (QuadAlu and Derivative) together. The overall noise was lower than in both origin files, just at the lower and upper boundaries of the spectrum, the QuadAlu respective the derivative algorithm created less noise.

## 10.2 Clicks

The derivative algorithm creates much less clicks in the sound output, due to the detection and interpolation of the blobs compared to the QuadAlu algorithm. But the playback with the needle doesn't seem to be influenced by the most of the blobs, which are mainly dust particles on the surface of the belt. The needle pushes these particles out of its way, compared to the contactless scan of the belt with the MPLS180 probe.

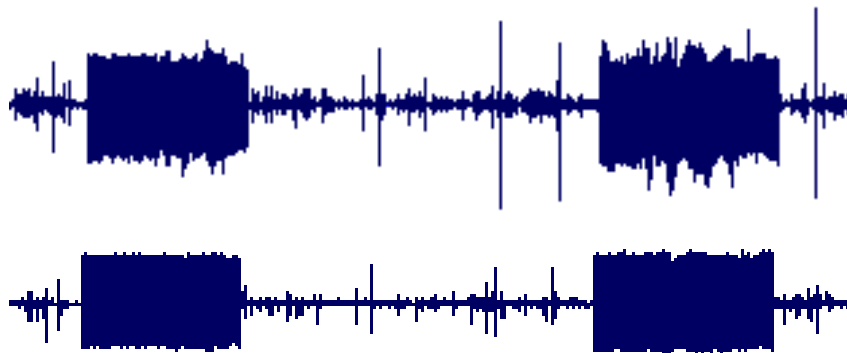


Figure 10.1: Compare of clicks in wav-files, created with QuadAlu(top) and Belt-Deriv(bottom)

## 10.3 Kennedy records

Carl and Earl went to Washington at the beginning of the project to scan a portion of the recordings made during the assassination of John F. Kennedy. The sound could be extracted out of the scans. The quality of the sound is acceptable, the voices can be understood. These scans can't be processed with the QuadAlu algorithm, because of the limited precision of the translation stage used during the scan, which result in variation of the distance between the four passes.

This autumn they will go back to Washington to scan the rest. Unfortunately it wasn't possible to add data extracted out of this files in the annex.

## 10.4 Future Improvements

Although that the project of scanning dictabelts last since almost ten years a lot of improvements can still be made.

### Improvement of the weight of the points

Up to now the outliers are rejected, a better result could maybe obtained if instead the outliers would be interpolated. This could reduce the white noise

### Scan more points

It would be helpful if the scan would take more points, using an 8Pass scan instead of a 4Pass scan, this would give twice as much redundancy to correct errors of the whole chain (Stage position error, Probe error, algorithms)

### Analysis of the needle

The playback needle seems to create a clean sound, it might be interesting to analyze how the needle touches the grooves and if it would be possible to simulate the needle in an algorithm.

### Correction of the curvature

An attempt was made to tilt the probe during the scan instead of using a 4Pass to get the same spacing between the points of a 4Pass, with one pass. The result was not satisfactory, due to variations of the height created by the curvature of the cylindrical form of the media.

### Interpolation of the folds

This point of the specification couldn't be treated due to a lack of time. But the blob detection seems to detect the folds, but not perfect. It shouldn't be too difficult to detect the two folds with a distance of the half of the height between each, by looking for a concentration of detected errors which corresponds to this criteria. The interpolation of the fold can be done with the existing spline interpolation, just by replacing the derivatives in the array with the Not A Number value.

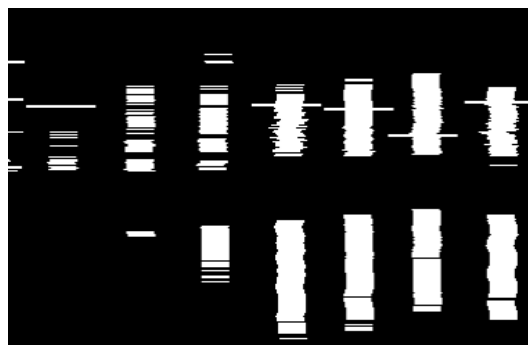


Figure 10.2: Blob cleaning detecting fold

# Chapter 11

## References

## Bibliography

- [1] Wikipedia. Assassination of john f. kennedy — wikipedia, the free encyclopedia, 2013. URL [http://en.wikipedia.org/w/index.php?title=Assassination\\_of\\_John\\_F.\\_Kennedy&oldid=567606370](http://en.wikipedia.org/w/index.php?title=Assassination_of_John_F._Kennedy&oldid=567606370). [Online; accessed 8-August-2013].
- [2] Wikipedia. Dictabelt evidence relating to the assassination of john f. kennedy — wikipedia, the free encyclopedia, 2013. URL [http://en.wikipedia.org/w/index.php?title=Dictabelt\\_evidence\\_relating\\_to\\_the\\_assassination\\_of\\_John\\_F.\\_Kennedy&oldid=543890573](http://en.wikipedia.org/w/index.php?title=Dictabelt_evidence_relating_to_the_assassination_of_John_F._Kennedy&oldid=543890573). [Online; accessed 8-August-2013].
- [3] STILSA. Mpls180 line sensors, 2008. URL [http://www.stilsa.com/catalog2/pdf/STILSA\\_MPLS.pdf](http://www.stilsa.com/catalog2/pdf/STILSA_MPLS.pdf). [Online; accessed 11-June-2013].
- [4] Wikipedia. Dictabelt — wikipedia, the free encyclopedia, 2013. URL <http://en.wikipedia.org/w/index.php?title=Dictabelt&oldid=534249939>. [Online; accessed 28-June-2013].
- [5] videointerchange.com. Dictabelt (1947), May 2012. URL [http://www.videointerchange.com/audio\\_history.htm](http://www.videointerchange.com/audio_history.htm). [Online; accessed 8-August-2013].
- [6] Adrien Nicolet. New probe for depth estimation of records: software. Diplomawork, 2010. EIA Fribourg, Lawrence Berkeley National Lab.
- [7] Alain Benninger. 2d and 3d scanning of metallic records and masters. Bachelor thesis, 2012. EIA Fribourg, Lawrence Berkeley National Lab.

- [8] Wikipedia. Normal distribution — wikipedia, the free encyclopedia, 2013. URL [http://en.wikipedia.org/w/index.php?title=Normal\\_distribution&oldid=566307994](http://en.wikipedia.org/w/index.php?title=Normal_distribution&oldid=566307994). [Online; accessed 2-August-2013].
- [9] Wikipedia. Bessel's correction — wikipedia, the free encyclopedia, 2013. URL [http://en.wikipedia.org/wiki/Bessel's\\_correction](http://en.wikipedia.org/wiki/Bessel's_correction). [Online; accessed 9-August-2013].
- [10] Wikipedia. Standardabweichung — wikipedia, die freie enzyklopedie, 2013. URL <http://de.wikipedia.org/w/index.php?title=Standardabweichung&oldid=120061524>. [Online; Stand 2. August 2013].
- [11] Wikipedia. Spline (mathematics) — wikipedia, the free encyclopedia, 2013. URL [http://en.wikipedia.org/w/index.php?title=Spline\\_\(mathematics\)&oldid=566366280](http://en.wikipedia.org/w/index.php?title=Spline_(mathematics)&oldid=566366280). [Online; accessed 1-August-2013].
- [12] Wikipedia. Spline interpolation — wikipedia, the free encyclopedia, 2013. URL [http://en.wikipedia.org/w/index.php?title=Spline\\_interpolation&oldid=558488137](http://en.wikipedia.org/w/index.php?title=Spline_interpolation&oldid=558488137). [Online; accessed 7-August-2013].
- [13] JChampion. A flexible charting library for .net, June 2007. URL <http://www.codeproject.com/Articles/5431/A-flexible-charting-library-for-NET>. [Online; accessed 3-August-2013].
- [14] Michel Yerly Noe Lutz. Studies of mecanical recording media with 3d surface profiling methods: Data collection and analysis. Diplomawork, 2005. EIA Fribourg, Lawrence Berkeley National Lab.

# Chapter 12

## Annex

### 12.1 Index of the DVD

#### **Administration**

Weekly reports

Journal

Report

Presentation at LBL

#### **Labview**

GeneratorRecorder.vi

CheckFilePath.vi

#### **Prism**

Program

Sample files

#### **These**

Report

Latex Sources

Images

References

## 12.2 Labview Generator / Recorder

### 12.2.1 Interface

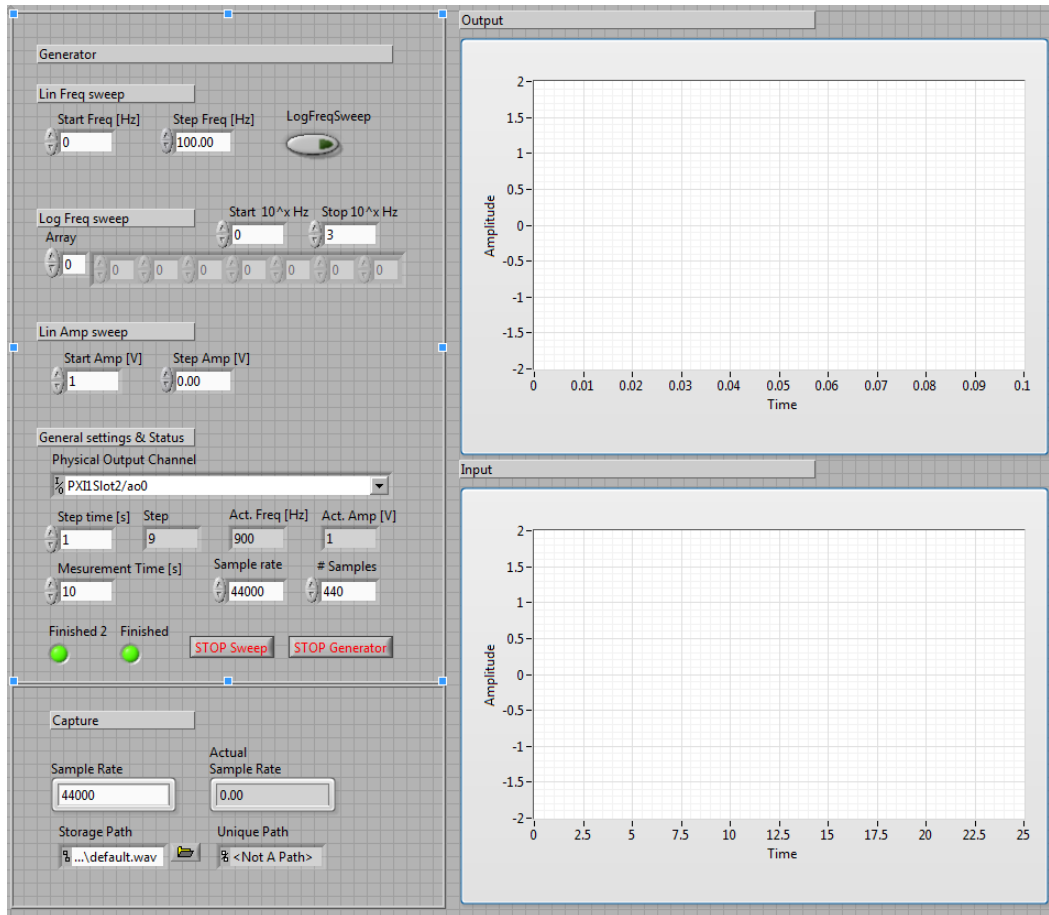


Figure 12.1: Interface of the Labview program



## 12.2.2 Generator

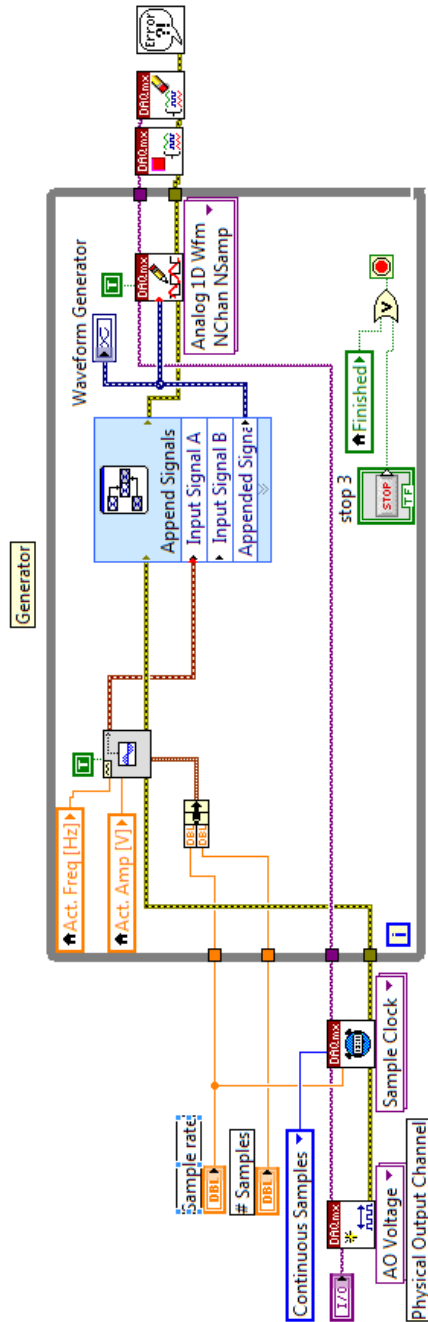


Figure 12.2: Function Generator with access to the physical analog output of NI rack

### 12.2.3 Linear sweep block

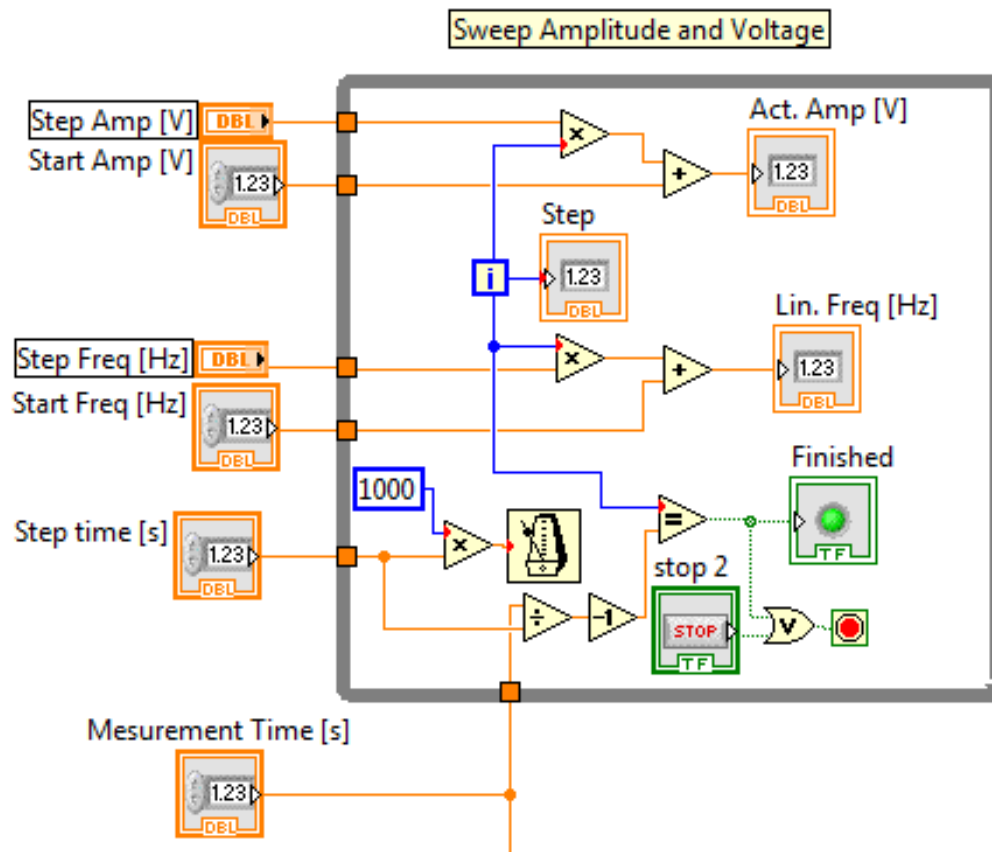


Figure 12.3: This block sweeps the variables for the current amplitude and frequency

### 12.2.4 Logarithmic sweep block

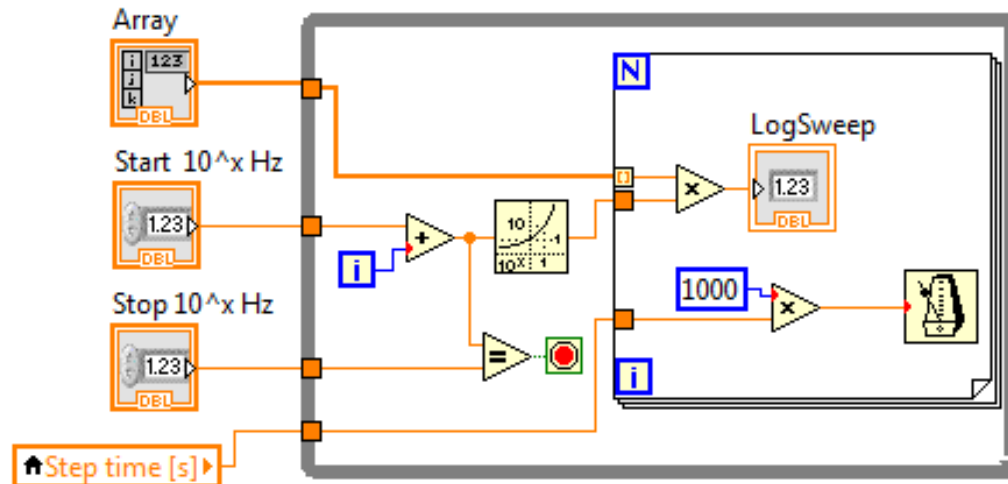


Figure 12.4: This block sweep the variable for the current frequency, if logarithmic sweep desired

### 12.2.5 Switch for frequency sweep

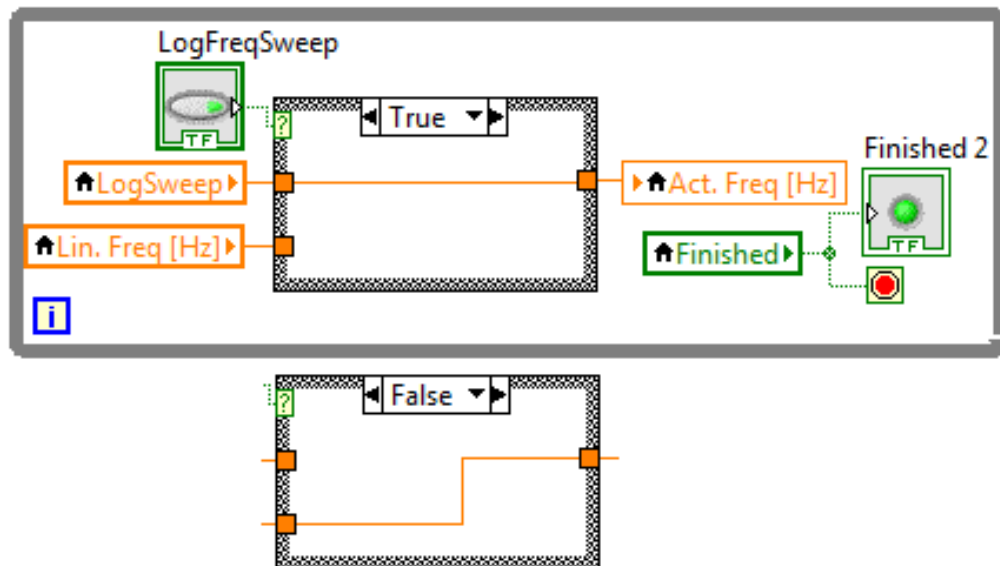


Figure 12.5: This block switches between logarithmic and linear sweep for the frequency.

## 12.2.6 Sound capture

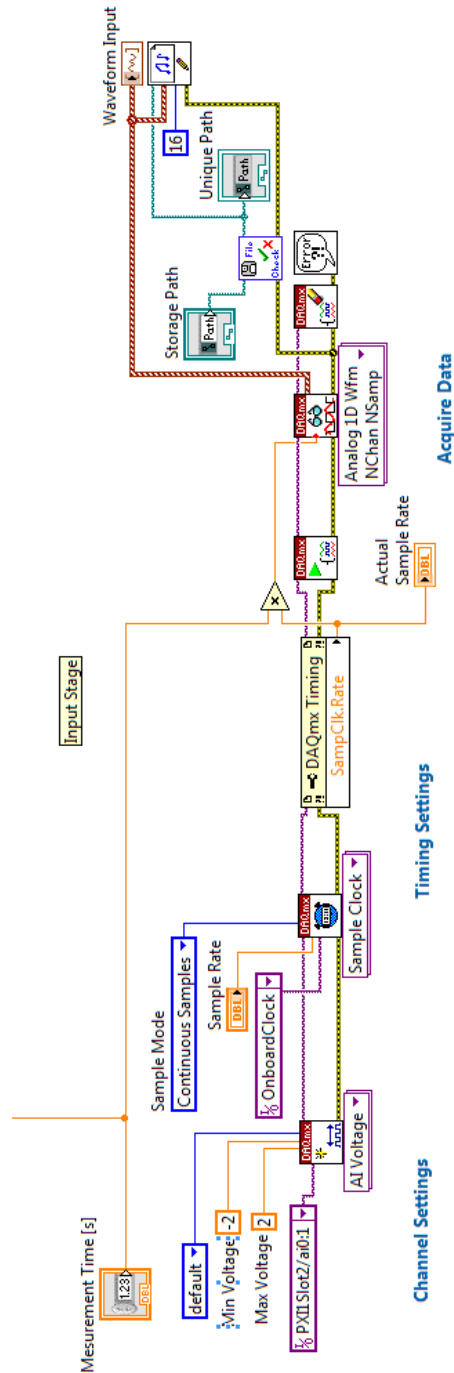


Figure 12.6: This block captures the data on both analog inputs and writes it into a wave-file.