Positioning and construction algorithms for a specific absolute positioning magnetic ruler system

Aleksandar Mojsic*

Abstract

Absolute positioning magnetic rulers are rulers which calculate the distance of the reading head based just on one reading of a magnetic signal. A new absolute positioning magnetic ruler method which is based on rulers with trapezoidal magnetic poles is considered in this paper. On a fixed position of a ruler, the reading head reads a signal on two stripes with several Hall sensors and is supposed to recover the reading position. The main aim of the paper is to create an algorithm for recovering the position for a fixed reading altitude. In order to model a ruler some angle restrictions on trapezoids are considered. We build a ruler with a simple heuristic tactic, extending it with a randomly chosen trapezoid if its angles satisfy the required angle conditions. The magnetic field of a ruler is numerically calculated with the software Radia and is used for testing of the algorithm. For an approximation of the signal function, we introduce a new piecewise approximation method which is based on the low rank approximation of a matrix. Compared to the piecewise polynomial approximation, this method significantly reduces the required memory while having the same accuracy, which is a crucial goal if the whole algorithm needs to be implemented on an FPGA. Creating this type of ruler raises the natural question of the maximum possible length. The algorithm naturally defines the stability of any ruler as a positive real number, which is crucial for its functioning. We show the correlation between the length of a constructed ruler and its stability.

Keywords: Absolute positioning, magnetic scale, magnetic ruler, magnetic field, calibration, low rank approximation, piecewise approximation, piecewise polynomial approximation

*Sedmog jula 106, 11251 Belgrade, Serbia. *E-mail address:* **mojsica@gmail.com**
Previous affiliation: Helmut Schmidt University / University of the Federal Armed Forces Hamburg, Department of Mechanical Engineering, Holstenhofweg 85, 22043 Hamburg, Germany.

1 Introduction

Magnetic rulers, also called scales, are a type of ruler which are used in many industrial applications. In the industrial market there exist two types of magnetic rulers: incremental and absolute. Some basic principles of how they work and their differences are described in [9], [10] and [1]. Just for the illustration here we describe the working principle of an incremental magnetic ruler.

Description of an incremental positioning ruler. This type of ruler is made such that rectangular ferromagnets of the same size are placed side by side, in an alternating order of south and north poles. Typical dimensions of a ruler are: the width is around 10 mm, the thickness is in the range of $1-2$ mm and the length is arbitrary and has no restrictions. The length of a single ferromagnetic rectangle is in the range of $2-5$ mm.

The reading head contains a single Hall effect sensor, which is on a constant altitude (around 1 mm) from the ruler's surface, equally away from the side edges of the ruler (figure 1). The Hall sensor reads the component of the magnetic field which is orthogonal to the surface, further referred to as the signal S. We set a Cartesian coordinate system such that the reading line of the Hall sensor is x - axis, with the origin placed at the beginning of the ruler. Then the signal S is a periodic function with a period of $2d$, where d is a length of a single pole. Moreover the signal is well defined by the restriction function $f := S\,|_{\left[a-\frac{d}{2},a+\frac{d}{2}\right]}$, where a is its zero closest to the origin.

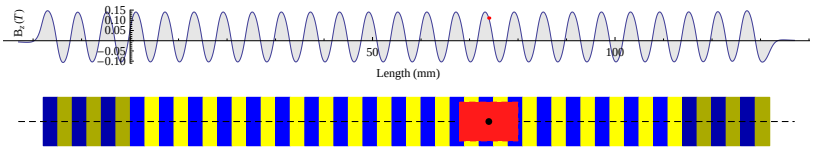

Figure 1: Illustration of the working principle of an incremental ruler. In blue color are north poles and yellow are south pole magnets. The red rectangle is the reading head containing a Hall sensor in its center (black dot). The graph shows the signal function S calculated on the dotted line, on a constant distance from the ruler surface.

To measure the distance from the beginning of the ruler to some destination point x, the head needs to start the measurement at the beginning of the ruler and then to move towards point x. A variable n counts the number of local extrema. In the final destination the head reads the signal strength $S(x)$, and a simple formula for the distance calculation is:

$$n \cdot d + (-1)^n f^{-1}(S(x)).$$

The method has a disadvantage because each time the head loses its signal, it needs to return to the beginning and restart the counting. This obstacle is overcome by the absolute positioning rulers, which can calculate the position of the head based on only one reading of the signal, no matter where it is placed.

An absolute positioning ruler. A recent absolute positioning ruler method is patented in [1], where the authors give an idea of its working principle. Magnetic pole trapezoids are placed instead of rectangles and the reading head should have more than one Hall sensor placed on two or more parallel stripes. Different inclination angle, which is defined with a border between neighboring poles, should guarantee the uniqueness of the signal reading for each position of the head (figure 2). The patent [1] does not offer any numerical description or guidance for the reconstruction of a reading position, except the idea that the inclination angle between two trapezoids should be detected.

In [10] the idea is modified and the uniqueness is suggested to be guaranteed with the uniqueness of the bases of a trapezoid .

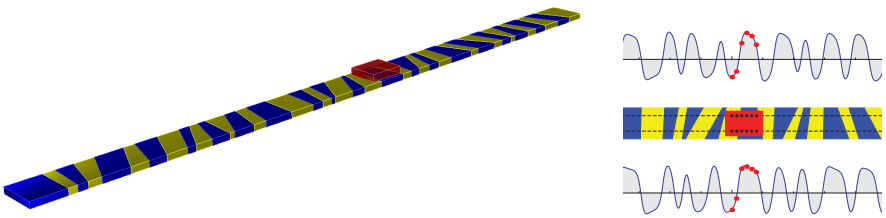

Figure 2: 3D visualization of the ruler made with trapezoids and illustration of a head reading the signal with 12 Hall sensors.

Objectives of the paper. The main motivation of this paper is to provide a robust algorithm for the calculation of the distance from the beginning of a ruler to the position of the head, based on one signal reading, which is an absolute positioning condition. The head contains an FPGA on which the algorithm should be implemented. Because of memory and processor limitation, the algorithm needs to be simple and require low memory consumption. The important task is to provide a ruler on which the algorithm can function properly, together with an answer on the question: how long such a ruler can be. Modeling magnetic rulers and calculation of their magnetic signal is also an important task.

The paper content. The paper consist of three sections. In section 2 we describe a fast method for the creation of ruler, with a simple heuristic tactic, extending it with random trapezoids if the required angle conditions on trapezoids are satisfied. We get a short ruler of 40 poles, named "*random_first_40_1.8_7.8*". That ruler together with its calculated magnetic field, we use as a benchmark data for the algorithm testings.

In section 3 we model magnetic ruler and get the signal function using the software Radia [2] with some described fixed numerical settings: the reading altitude, the direction of magnetization and the ferromagnet material. Radia is a software which is using Mathematica [8] as interface. The method used in Radia belongs to the category of boundary Integral Methods and differs strongly from the Finite Element Methods (FEM). In [5] and [6] the authors made the comparison with the FEM codes. Generally it provides a numerical solution faster than with the FEM.

In signal data modeling we include some industrial problems such as displacement of the Hall sensors and noise.

In the last section we describe the absolute positioning algorithm, which is the main result of the paper. Here we give a short description of its three subsections.

The calibration of the Hall sensors is a method introduced in subsection 4.1. Knowing that the Hall sensors measure the same signal along a reading line, we expect the same measured value, although in practice, because of several reasons the result is not the same. This industrial problem, we model in our benchmark data in subsection 3.1.3 with the Hall sensors displacement and added Gaussian noise. The first Hall sensors measurement, we take as a reference, and other Hall sensors we "correct" in a linear form, in order to minimize the least square error with respect to the reference one. In analytical chemistry, for example described in [7], the same methodology is often used for the calculation of a calibration curve.

Approximation and compression of the signal is an important issue that is solved in subsection 4.2. Since the algorithm should be implemented on an FPGA, the memory requirement is demanding. The processor needs to have memorized the entire signal function and needs to be able to reconstruct the signal function on each part of the ruler. The straight forward idea, which satisfies those conditions, is the piecewise polynomial approximation. Here we improve the idea using the low rank approximation of a matrix. The low rank approximation is a minimization problem in which the cost function measures the distance between a given matrix $M \in \mathbb{R}^{m\times n}$ and an approximating matrix $N \in \mathbb{R}^{m\times n}$, subject to a constraint that N has a reduced rank:

$$\min_{\{N\in\mathbb{R}^{m\times n}:\, rank(N)\leq r\}} \|M-N\|_F\,, \tag{1}$$

where $\|\cdot\|_F$ is the Frobenius norm. The existence of the solution is guaranteed by the Eckart–Young–Mirsky theorem in [3]. For any of the algorithms we refer to [4]. We define segments with neighboring zeros of a signal function. On every of those segments we define an equidistant grid so that all grids have the same number of elements. Let M be the matrix whose columns are sampled signal functions on those grids. Since the signal functions restricted on each corresponding segment has "similar shape", it is expected that the rank of M is a low number. That is why $r-$rank approximation matrix N is expected to be a good approximation of matrix M, even for a small number r. With any rank

decomposition we have:

$$N = R \cdot \Lambda,$$

where $R \in \mathbb{R}^{m \times p}$ and $\Lambda \in \mathbb{R}^{p \times n}$ for $p = rank\,(N)$, thus we see that the segments of a signal are approximated with a linear combination of the columns of R. We approximate those r discrete vectors with polynomials. That results in a piecewise approximation with a linear combination of r polynomials. The method saves up to 30% of memory comparing to the piecewise polynomials while having the same approximation error.

Knowing the approximated signal functions, for every position on the ruler we can assume the signal reading of the head. For a signal reading of the head, we define the fitting norm of a position as the euclidean distance of the signal reading and the assumed signal reading for that position. The absolute positioning algorithm is based on an idea to find the position with the smallest fitting norm. Because of the Hall sensors displacement and the noise of the signal that norm is not zero. The problem can occur when there is another position on the ruler with the fitting norm close to the best fitting norm. For that reason, for every position, the second best fitting norm needs to be on a stable distance from the best fitting norm. Without including the Hall sensors displacement and the noise, we closer observe the second best fitting norm function ($SBFN$) as an important characteristic of ruler. With the minimum of $SBFN$ we define the stability of a ruler. We find that the stability is reduced on longer rulers created with the heuristic random method.

2 A heuristic method for the ruler construction

The ruler needs to be created in a way that each reading of the signal guarantees the uniqueness of the position. In [9] a mixed integer programming model is described, whose solutions are the rulers constructed in a way to have unique trapezoid bases. Here we guarantee uniqueness using random numbers as pole lengths. We create a ruler using any programming language's pseudorandom generator, which has a uniform distribution on segment $[0, 1]$, here noted as $random()$. The basic idea of this algorithm is the following: We extend the ruler with another random trapezoid and we stop when the required length of the ruler is achieved. The advantage is that the ruler can be constructed as long as we want, but with growing length its stability is reduced (Section 4.4).

Remark 1. The method does not include a demand for the uniqueness of the inclination angles or the uniqueness of the bases of a trapezoid, as it is suggested in [1] and [10], respectively.

Experimental setup. A reading head holds 12 Hall sensors, 6 on each of two lines, with the distance of 1.6 mm between the Hall sensors. The ruler is 10 mm wide, the distance between the lines is 5 mm, both placed 2.5 mm away from the edges of the ruler (figure 3).

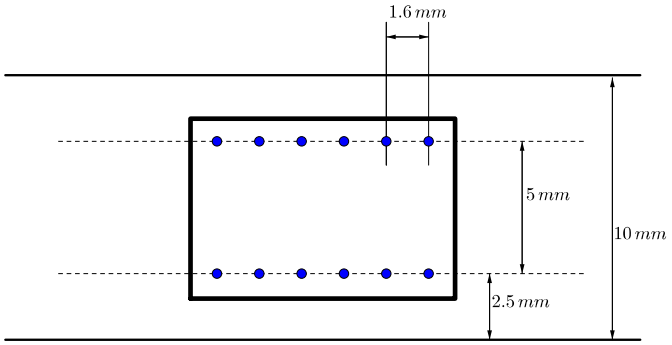

Figure 3: Reading head with 12 Hall sensors and their position relative to the ruler.

Because of industrial production limitations, there might exist some angle restrictions in the trapezoids. In particular, here we implement the limitation where each angle between lateral sides and bases needs to be in the interval $\left[60^{\circ}, 120^{\circ}\right]$ and the difference between 2 same angles in consecutive trapezoids is smaller than 23°. The bases of the trapezoid are not smaller than 1 mm. Further we consider a pole trapezoid defined by two numbers which refer to the pole lengths over the reading line. If those two numbers are a and b, the bases of the trapezoid are $\frac{1}{2}(3a-b)$ and $\frac{1}{2}(3b-a)$ (figure 4). We assume that the pole length over the reading line is bounded with numbers $\ell_{\min}$ and $\ell_{\max}$.

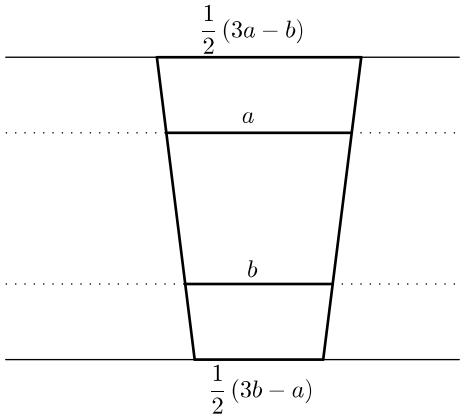

Figure 4: A pole defined with pair (a, b).

Algorithm 1 - construction

- **Step 0.** The ruler has only one north pole trapezoid defined with the pair $R = (\ell_{start}, \ell_{start})$.
- **Step 1.** If the the length of the ruler R is big enough, go to Step 4.
- **Step 2.** Get a pair of numbers,

  $$(x, y) = (\ell_{\min} + (\ell_{\max} - \ell_{\min}) \cdot random(),\ \ell_{\min} + (\ell_{\max} - \ell_{\min}) \cdot random()).$$

- **Step 3.** If the angle condition of the ruler $(R, (x, y))$ is satisfied, set $R = (R, (x, y))$ and go to Step 1, otherwise go to Step 2.
- **Step 4.** $R = (R, (\ell_{end}, y))$, where y is calculated so the last angle of the ruler has 90 degrees.

Result. Here we set $\ell_{start} = \ell_{end} = 9\,\text{mm}$. Because the distance between neighboring Hall sensors is 1.6 mm, for any case we want the pole lengths on the reading lines to be not smaller than $\ell_{\min} = 1.8\,\text{mm}$ and not bigger than $\ell_{\max} = 7.8\,\text{mm}$. The first (north) and the last (south) pole have pole lengths bigger then 8 mm, but they are the only ones so we expect it does not affect the uniqueness of the reading. For testing and demonstration purpose of the algorithm, we use a benchmark ruler made of only 40 poles, "*random_first_40_1.8_7.8*" (figure 5). The first number in the name of the benchmark ruler indicates the number of poles and the other two numbers are $\ell_{\min}$ and $\ell_{\max}$. Because the signal changes its shape at the edges of the ruler, before the beginning and after the end of the ruler we place 6 rectangular poles, each of 3 mm length.

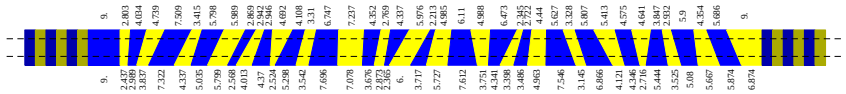

Figure 5: A ruler "*random_first_40_1.8_7.8*" with length 190.958 mm. Blue color - north poles, yellow color - south poles. The numbers next to the ruler stand for distance between poles over the two dashed lines - the reading lines of the Hall sensors.

3 Creating a signal data as a numerical calculation of magnetic field

In this section we model a ruler with the software Radia, calculate its magnetic field and create a simulated data set which is necessary for testing the main result of the paper, the algorithm for absolute positioning. Before calculating the magnetic field we explain the experimental setup used in the software Radia.

3.1 Experimental setup

3.1.1 The model of a ruler in the software Radia

To use the software, we need to define every ferromagnetic piece as a trapezoidal prism in the Cartesian coordinate system. For the length unit we use one millimeter. We place the ruler such that its upper surface belongs to the plane Oxy, the axis Ox holds the middle line of the upper surface of the ruler, and the origin is at the beginning of the ruler. The positive half line of the axis Oz has no intersections with the ruler except of the origin. The ruler is 10 mm wide and the thickness is 1.64 mm. We use the material **Steel42**, which is predefined in Radia. The direction of magnetization in the blocks are vectors $\vec{v} = (-0.27, 0, 1)$ and $-\vec{v}$ alternately, in Tesla. Since the Hall sensors read only the component of the magnetic vector field which is orthogonal to the plane of the ruler's surface, in every point where the field is calculated we consider only the third component of the calculated vector.

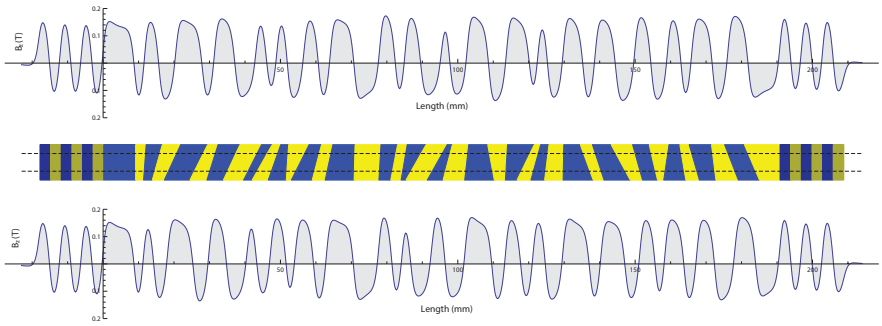

Figure 6: Signal of ruler "*random_first_40_1.8_7.8*" solved with Radia. Reading altitude from the surface of the ruler is 1.1 mm. Signal is computed on two lines: $(0, 2.5, 1.1) + t \cdot (1, 0, 0)$ and $(0, -2.5, 1.1) + t \cdot (1, 0, 0)$, $t \in [-23,\ R_{length} + 23]$.

3.1.2 The reading altitude

With different reading altitudes the signal changes its "shape" and intensity. In figure 7 it can be seen that the angle of magnetization has a stronger influence on lower reading altitudes. The influence is even stronger on bigger poles, where the signal besides zeros has two more inflection points per pole. That causes difficulties with the approximation of the signal. When the reading altitude is high, the intensity of the signal is lower but the shape is more suitable for the polynomial approximation, because its zeros are the only inflection points. The problem with the high reading altitude is that distances between two consecutive zeros of the signal tend to be smaller on the places where the poles are small. The distance of the Hall sensors is 1.6 mm and because of that reason the minimum distance between consecutive zeros of the signal should be bigger than 1.6 mm.

Having in mind the above considerations, we conclude that the best reading altitude is around 1 mm. Further in this paper we use measured data for a fixed reading altitude of 1.1 mm.

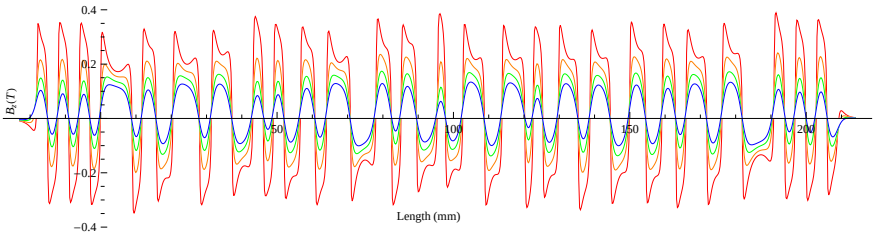

Figure 7: Signal measurement with different altitudes on one reading track of the ruler "*random_first_40_1.8_7.8*" : red - 0.1 mm, orange - 0.6 mm, green - 1.1 mm, blue - 1.6 mm.

3.1.3 The offset of positions of the Hall sensors and the noise of the signal

Positions of the Hall sensors In the industrial production the Hall sensors are not ideally placed on the anticipated positions of a head. There is always an offset in all 3 dimensions. In order to solve the industrial problem, here we want to create a data set which describes a real life experimental readings. The mentioned offset is certainly important to include. We assume that the numbers which define the disturbance of the positions of the Hall sensors $\triangle^i_{x,y,z} = (\triangle x_i, \triangle y_i, \triangle z_i)$, $i = 1, ..., 12$, have a normal distribution $\mathcal{N}(0, 0.1)$.

Noise of the signal Naturally we want to incorporate noise in a signal and closer observe its influence on the positioning algorithm. For that purpose we apply additive white Gaussian noise, with signal to noise ratio $stn = 60$, per sample, in dB.

In section 4.4 we discuss the influence of the noise and Hall sensors displacement on the stability of the algorithm.

3.2 The simulated data set

For every point in the already adjusted coordinate system we can calculate the field using the Radia software. Let function $\mathcal{S} : \mathbb{R}^3 \to \mathbb{R}$ map every point into the third component of the magnetic field vector in that point. We have that 12 functions:

$$H_i(x) = \mathcal{S}\left((x + (i-1) \cdot 1.6,\ 2.5,\ 1.1) + \triangle^i_{x,y,z}\right),\ i = 1, 2, ..., 6,$$

and

$$H_i(x) = \mathcal{S}\left((x + (i-7) \cdot 1.6,\ -2.5,\ 1.1) + \triangle^i_{x,y,z}\right),\ i = 7, 8, ..., 12.$$

for $x \in [0, R_{length} - 8]$, simulate the readings of the Hall sensors when the center of the head is in position $x + 4$. The values of functions $H_i(x)$, $i = 1, 2..., 12$, we calculate on a discrete set of points on an equidistant grid of the interval $[-9,\ R_{length} + 9]$. R_{length} stands for the length of the ruler. We choose a grid with a spacing of 0.01 mm. Further we consider that each function $H_i(x)$, $i = 1, 2, ..., 12$, is a piecewise linear approximation on the known grid and the sampled values.

4 An absolute positioning algorithm

Without losing any generality, in subsections 4.1 and 4.2 we consider functions H_i, $i = 1, 2, ..., 6$. The same conclusions and methods can be applied on functions H_i, $i = 7, 8, ..., 12$.

4.1 Calibration of the Hall sensors

Distances between adjacent Hall sensors are 1.6 mm. However in the industrial production process it is impossible to achieve absolute precision. That is why we need to take into consideration the calculation of the exact positions of the Hall sensors. For this purpose we set the following problems:

$$\min_{\delta} \int_D |H_1(x) - H_k(x - \delta)|^2 \, dx, \quad k = 2, 3, ..., 6, \tag{2}$$

where $D = (0,\ R_{length} - 8)$. Because the functions H_i, $i = 1, 2, ..., 6$ are piecewise linear, for the numerical evaluation of the integral we use the rectangle method. Although it is theoretically expected, in experimental results for the solution δ_k we do not have $\int_D |H_1(x) - H_k(x - \delta_k)|^2 \, dx = 0$.

This happens because of three main reasons:

- In the production there is an offset of the Hall sensors in all 3 dimensions. That means that the Hall sensors do not measure the signal exactly on the same distance from the ruler, and the signal closer to the ruler is stronger. At the same time because of the offset in y-direction and because the poles are trapezoids it is clear that the zeros of $H_1(x)$ and $H_k(x - \delta_k)$ are not on the same positions.

- The Hall sensors can not be calibrated ideally. Meaning that with different Hall sensors the same signal is not measured with the same values.

- Noise.

The above problems are solved by a posteriori calibration of the Hall sensors. The measured signal can be filtered and corrected which would result in an acceptable reading of the head. In figure 8 we see that the best way to calibrate the Hall sensors is linear regression.

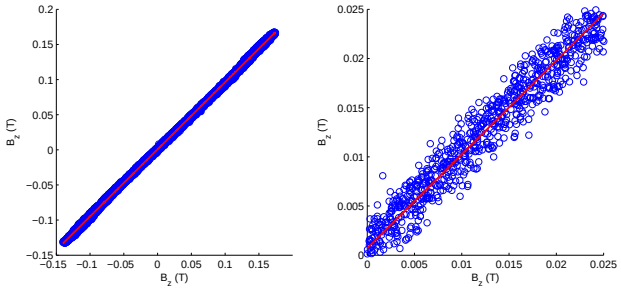

Figure 8: Left: scatter plot of $(H_1(x), H_2(x-\delta_2))$ on an equidistant grid of domain and the red is the calibration curve. Right: The same plot on segment $[0, 0.025]$.

We consider the function $H_1(x)$ to be the reference one, and the other functions $H_k(x)$, $k = 2, 3, ..., 6$, we "correct" with the following formula:

$$H_k(x) := a^k H_k(x) + b^k. \tag{3}$$

We calculate (a^k, b^k), $k = 2, ..., 6$, in order to minimize $\int_D |H_1(x) - H_k(x - \delta_k)|^2 dx$. With discretized D, we have an over-determined linear system of equations, which we solve with the least squares method.

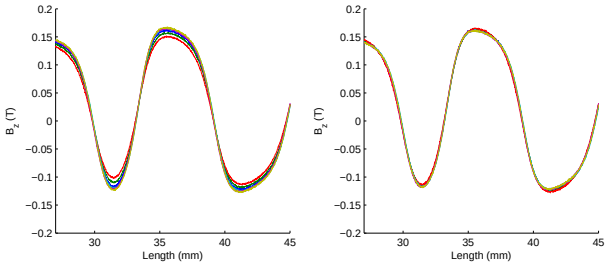

Figure 9: Functions $H_1(x)$, $H_2(x-\delta_2)$,..., $H_6(x-\delta_6)$ before and after calibration.

Pointing out that $\delta_1 = 0$, we further consider the function:

$$S(x) = \frac{1}{6}\sum_{k=1}^{6} H_k(x - \delta_k) \tag{4}$$

as the reference signal function $S : [\acute{b}, \grave{b}] \to \mathbb{R}$, where $\acute{b}$ is any number in the segment $(-1.5, 1.5)$ for which $\acute{b}$ is smaller than the closest zero of $S(x)$ nearby the origin. Equivalently $\grave{b}$ is any number in segment $(R_{length} - 1.5, R_{length} + 1.5)$ bigger than the closest zero of the $S(x)$ nearby R_{length}. With that settings of numbers $\acute{b}$ and $\grave{b}$, all zeros of $S(x)$ are in $[\acute{b}, \grave{b}]$.

4.2 Approximation of the signal

One of the main tasks is the approximation of the signal. The processor itself needs to have memorized signal functions on both lines. Since the memory capacity on FPGA is limited, the approximation must be done with a low amount of memory. At the same time the approximation must not have any errors in the zeros of the signal functions. Because of memory capacity and processor speed limitation, the signal function should be reconstructed by segments so that we can find the inverse function of the signal only in a certain part of the ruler, which exclude the fast Fourier transformation (FFT) as possibility. The above demands can be concise:

- The signal should be approximated piecewise, in segments defined with consecutive zeros of the function.
- The approximation must consume at least two numbers per segment i: "the length" l_i and "the height" h_i, where l_i is the distance between consecutive zeros, and h_i is the local extremum on the corresponding interval.

The following method gives a natural solution satisfying the above two conditions.

4.2.1 Approximation with representative functions

We see that the shape of the signal changes depending on a number l_i. The idea for the approximation is to divide the segment $[\min l_i, \max l_i]$ and create n_g length intervals:

$$[t_0, t_1), [t_1, t_2), ..., \left[t_{n_g-1}, t_{n_g}\right],$$

where $t_0 = \min l_i$, $t_{n_g} = \max l_i$ and $t_0 < t_1 < t_2 < t_{n_g-1} < t_{n_g}$.

For every length interval, we define a group of functions, whose members are the signal function restricted to the segment i if l_i belongs to the length interval. We construct a representative function for each length interval as the arithmetic mean of its group of functions (figure 10).

Having computed the representative functions, we approximate the signal in segments, from zero to zero, with a suitable representative function.

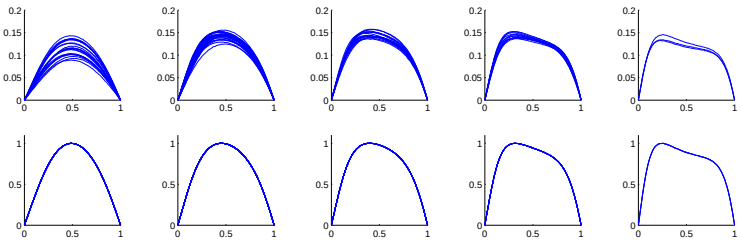

Figure 10: Above: The groups of functions for 5 length intervals. Below: 5 corresponding representative functions.

The above described method we can write in mathematical form. Let $S : [\acute{b}, \grave{b}] \to \mathbb{R}$ be the signal function defined in (4), and let numbers a_i, $i = 1, ..., n_z$, denote all its zeros such that $\acute{b} < a_1 < a_2 < \ldots < a_{n_z} < \grave{b}$. Clearly $n_z - 1$ is the number of poles of the ruler. We define $n_z - 1$ functions $s_i : [0, 1] \to \mathbb{R}$,

$$s_i(x) := |S((a_{i+1} - a_i)\, x + a_i)|\,,\ i = 1, 2, ..., n_z - 1.$$

If n_g is the number of the length interval groups, naturally we have the mapping,

$$\sigma : i \to \sigma(i) \in \{1, 2, ..., n_g\}\,,\ i = 1, ..., n_z - 1,$$

and each group defines its representative function $f_j : [0, 1] \to [0, 1]$:

$$f_j = \frac{\hat{f}_j}{\max_{[0,1]} \hat{f}_j},\ j = 1, ..., n_g,$$

where

$$\hat{f}_j = \sum_{\{k:\sigma(k)=j\}} s_k.$$

We define the numbers $l_i := a_{i+1} - a_i$, $i = 1, ..., n_z - 1$, which represent the length of segment i, and we construct the approximation function $F : [a_1, a_{n_z}] \to \mathbb{R}$:

$$F(x) = \begin{cases} h_1 \cdot f_{\sigma(1)}\left(\frac{1}{l_1}(x - a_1)\right), & x \in [a_1, a_2)\,, \\ h_2 \cdot f_{\sigma(2)}\left(\frac{1}{l_2}(x - a_2)\right), & x \in [a_2, a_3)\,, \\ \ldots \\ h_{n_z-1} \cdot f_{\sigma(n_z-1)}\left(\frac{1}{l_{n_z-1}}(x - a_{n_z-1})\right), & x \in [a_{n_z-1}, a_{n_z}]\,, \end{cases}$$

where the numbers h_i, $i = 1, 2, ..., n_z - 1$, are calculated as the solution of the following minimization problem

$$\min_{h_1, h_2, ..., h_{n_z-1}} \int_{a_0}^{a_{n_z}} |F(x) - S(x)|^2\, dx.$$

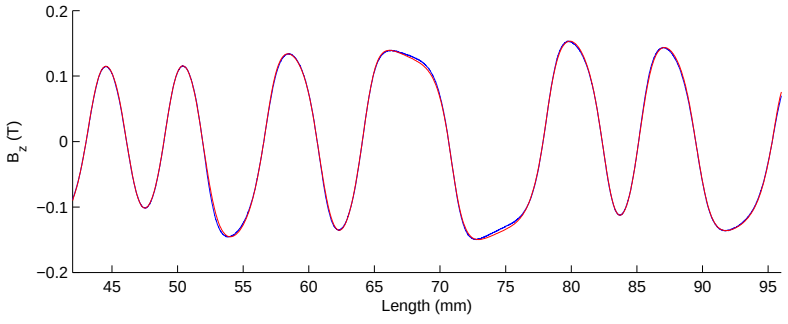

Figure 11: A segment of the signal of ruler "*random_first_40_1.8_7.8*" (blue) together with its approximation (red). The approximation function is calculated with the representative functions from figure 10.

An approximation with these representative functions consumes only 2 numbers per segment plus the memory needed for storage of the representative functions. Because of its simplicity and the low memory consumption, this method is worth considering (figure 11).

4.2.2 Piecewise polynomial approximation

In case we can afford more than two numbers per segment, we can improve the approximation by using a piecewise polynomial approximation. We approximate the functions s_i directly with a polynomial of the form:

$$s_i(x) \approx p_i(x) = x(x-1)\left(c_1^i + c_2^i x + ... + c_r^i x^{r-1}\right),\ i = 1, .., n_z - 1, \tag{5}$$

Sampling s_i, $i = 1, 2, ..., n_z - 1$, on an equidistant grid $I = \left[0, \frac{1}{d}, \frac{2}{d}, ..., \frac{d-1}{d}, 1\right]$, we get vectors $s_i(I) = \left[0, s_i\left(\frac{1}{d}\right), s_i\left(\frac{2}{d}\right), ...\ , s_i\left(\frac{d-1}{d}\right), 0\right]$, $i = 1, 2, ..., n_z - 1$, where d is some large natural number. Since $\min l_i \in (1.5\,\text{mm}, 2\,\text{mm})$ and samplings of the signal function are taken on an equidistant grid with spacing 0.01 mm, we deduce that the number d should be smaller than 150, in this paper we use $d = 100$.

We closer observe matrix M, where $s_i(I)$ are placed as columns:

$$M = \left(\begin{bmatrix} \\ s_1(I) \\ \\ \end{bmatrix} \begin{bmatrix} \\ s_2(I) \\ \\ \end{bmatrix}, ..., \begin{bmatrix} \\ s_{n_z-1}(I) \\ \\ \end{bmatrix}\right). \tag{6}$$

Let us define a matrix:

$$D^r = \left(\begin{bmatrix} \\ (I-1)\,I \\ \\ \end{bmatrix} \begin{bmatrix} \\ (I-1)\,I^2 \\ \\ \end{bmatrix}, ..., \begin{bmatrix} \\ (I-1)\,I^r \\ \\ \end{bmatrix}\right), \tag{7}$$

having in mind the notation where $(I-1)\,I^n$ is a vector column with elements $\left(\frac{i}{d} - 1\right)\left(\frac{i}{d}\right)^n$, $i = 0, ..., d$.

If we place vectors $p_i(I)$, $i = 1, 2, ..., n_z - 1$, as columns, we get a matrix:

$$D^r \cdot C,$$

where the rows of matrix C are the coefficients $c_1^i, c_2^i, ..., c_r^i$, $i = 1, .., n_z - 1$, in (5).

To sum up, with sampling on I, we transform (5) into the overdetermined linear system:

$$D^r \cdot C = M, \tag{8}$$

which we solve with the least squares method.

It is clear that $s_i(I)$ is a discrete approximation of the function s_i, so all one needs to achieve for an approximation of the signal function S is to store the matrix M and numbers l_i, $i = 1, ..., n_z - 1$. With this method the approximation error of the matrix M is

$$\|D^r \cdot C - M\|_F .$$

At the end we have the approximation function $F : [a_1, a_{n_z}] \to \mathbb{R}$:

$$F(x) = \begin{cases} p_1\left(\frac{1}{l_1}(x - a_1)\right), & x \in [a_1, a_2), \\ p_2\left(\frac{1}{l_2}(x - a_2)\right), & x \in [a_2, a_3), \\ \dots \\ p_{n_z-1}\left(\frac{1}{l_{n_z-1}}(x - a_{n_z-1})\right), & x \in [a_{n_z-1}, a_{n_z}]. \end{cases}$$

Problem 2. (5) can be written in the form:

$$s_i(x) \approx c_1^i x(x-1) + c_2^i x^2(x-1) + ... + c_r^i x^r(x-1) \tag{9}$$

and it is clear that s_i is approximated with a linear combination of r polynomials:

$$x(x-1),\ x^2(x-1), ...,\ x^r(x-1). \tag{10}$$

One important question raises: Can we replace those r polynomials in (10) with some other r polynomials, in order to improve the approximation? The memory consumption for the signal approximation would remain r coefficients per segment plus the polynomial coefficients, so the answer to this question is important.

4.2.3 Low rank approximation

We closer observe a matrix M, defined in (6). Matrix M can be approximated with a matrix N of lower rank $r = rank(N)$. For a matrix N of rank r, we find the rank factorization:

$$M \approx N = R \cdot \Lambda \tag{11}$$

were $R \in \mathbb{R}^{(d+1)\times r}$ and $\Lambda \in \mathbb{R}^{r\times(n_z-1)}$. Clearly the rank factorization is not unique because we have

$$N = R \cdot \Lambda = (R \cdot P) \cdot \left(P^{-1} \cdot \Lambda\right), \tag{12}$$

for any $P \in \mathbb{R}^{r\times r}$, which has an inverse. With (11) we approximate each vector $s_i(I)$ with the columns of the matrix N. Because of (12) each column of the matrix N is equal to the linear combination of r vectors, which are the columns of the matrix $R \cdot P$. If we approximate them with polynomials, we have an answer for problem 2. So equivalently as in (8), with the solution of the overdetermined linear system:

$$D^t \cdot Z_P = R \cdot P$$

we approximate the columns of matrix $R \cdot P$ with t polynomials whose coefficient are the columns of matrix Z_P.

It is easy to see that no matter what is the choice of the matrix P,

$$\left\| D^t \cdot Z_P \cdot \left(P^{-1} \cdot \Lambda \right) - M \right\|_F ,$$

is the same number.

Generally, problem 2 can be written in form:

$$\min_{Z \in \mathbb{R}^{t \times r}, \Lambda \in \mathbb{R}^{r \times (n_z - 1)}} \Phi \left(Z, \Lambda \right), \tag{13}$$

where $\Phi \left(Z, \Lambda \right) := \left\| D^t \cdot Z \cdot \Lambda - M \right\|_F$ and $D^t \in \mathbb{R}^{d \times t}$ is defined in (7).

So far we do not have a solution for problem (13). Knowing that (11) is an optimal rank factorization, with $(C, \Lambda) = \left(C_P, P^{-1} \cdot \Lambda \right)$ we assume we are "close" to the solution of (13). Table 1 shows us that we do not have an optimal solution, because $\Phi \left(Z, \Lambda \right)$ is smaller for $r = 7$ than for $r = 8$.

Remark 3. Comparing (11) and our approximation with representative functions, we see that each function s_i is approximated with a linear combination of r discrete functions instead of approximation with one function, which represents the length group $\sigma \left(i \right)$.

4.2.4 Comparison of the approximation methods

Table 1 shows the advantage of new method comparing to the piecewise polynomial approximation.

	r =3	r =4	r =5	r =6	r =7	r =8
$\left\| D^r \cdot C - M \right\|$	0.14599	0.11963	0.057186	0.038722	0.011395	0.0078793
$\left\| D^t \cdot Z \cdot \Lambda - M \right\|$	0.06699	0.02307	0.012136	0.005985	0.0054289	0.0054334

Table 1: The error of the approximations of a matrix M, with polynomial approximation and with a new method for $t = 10$. The matrix M is calculated on the signal of ruler "*random_first_40_1.8_7.8*".

4.3 The positioning algorithm

Programming of the processor

The positioning algorithm we propose in this section solves the following inverse problem:

Problem 4. At a fixed but unknown position, the head reads 12 values of the signal, 6 on each line. One should determine the position $\bar{x}$ of the head.

Before we formulate the algorithm, we need to calculate the data which are specific for each different ruler with head and its Hall sensors. Further the index variable ℓ defines the belonging to the track that holds the Hall sensors

H_i, $i = 1, ..., 6$, or H_i, $i = 7, ..., 12$, with the corresponding values 1 or 2. For both tracks we solve the problem 2 and get δ_i^ℓ, $i = 1, 2, ..., 6$, appointing that $\delta_1^\ell = 0$. As described in subsection 4.1, we get the coefficients $\left(a^k, b^k\right)$, $k = 2, ..., 6$. Equivalently we do the same for the other track, when H_7 is the reference Hall sensor, and we get coefficients $\left(a^k, b^k\right)$, $k = 8, ..., 12$. As in (4), we calculate the signal functions S_ℓ, $\ell = 1, 2$. Using any of approximation methods presented in subsection 4.2 we approximate S_1 and S_2 with F_1 and F_2, respectively.

To sum up, the following data needs to be stored in the memory of the processor: δ_k^ℓ, $\left(a^{k+(\ell-1)6}, b^{k+(\ell-1)6}\right)$, $\ell = 1, 2$, $k = 2, ..., 6$, and the data for approximation functions F_1 and F_2.

Algorithm 2 - positioning

1. **The input.** In one reading from the Hall sensors the processor gets 12 values, 6 in both lines: $\left(v_1^1, ..., v_6^1\right)$ and $\left(v_1^2, ..., v_6^2\right)$.

2. **Filtering the input.** Besides v_1^1 and v_1^2, other 10 values are filtered with formula (3), using coefficients calculated in programming part:

$$v_k^\ell := a^{k+(\ell-1)6} \left(v_k^\ell\right)^i + b^{k+(\ell-1)6}, \quad k = 2, ..., 6, \ \ell = 1, 2.$$

3. **The reference point.** Find $\bar{i}$ and $\bar{\ell}$ such that

$$\left|v_{\bar{i}}^{\bar{\ell}}\right| = \min\left\{\left|v_i^\ell\right|, \ \ell = 1, 2., \ i = 1, ..., 6\right\}.$$

4. **Calculating the fitting norm.** The set $F_{\bar{\ell}}^{-1}\left(v_{\bar{i}}^{\bar{\ell}}\right)$ gives possible positions of the Hall sensors $\bar{i}$. For every $x \in F_{\bar{\ell}}^{-1}\left(v_{\bar{i}}^{\bar{\ell}}\right)$ calculate the fitting norm

$$\mathcal{N}(x) = \left(\sum_{k \in \mathbb{Z}: \, \bar{i}+k \in \{1,...,6\}} \sum_{\ell=1}^{2} \left(F_\ell\left(x + \left(\delta_{\bar{i}+k}^\ell - \delta_{\bar{i}}^\ell\right)\right) - v_{\bar{i}+k}^\ell\right)^2\right)^{\frac{1}{2}}$$

5. **The output.** The position with the smallest fitting norm is the position of the Hall sensor $\bar{i}$ on the line $\bar{\ell}$,

$$x_r : \mathcal{N}(x_r) = \min\left\{\mathcal{N}(x) \ : \ x \in F_{\bar{\ell}}^{-1}\left(v_{\bar{i}}^{\bar{\ell}}\right)\right\}. \tag{14}$$

Finally we have:

$$\bar{x} = x_r - \delta_{\bar{i}}^{\bar{\ell}}.$$

In summary, the algorithm searches for the position with the smallest fitting norm (figure 12).

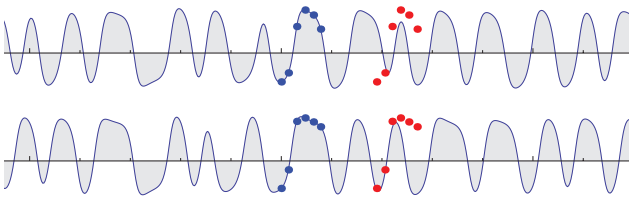

Figure 12: The illustration of the fitting method. Red dots are at the wrong and blue dots on the correct position.

Remark 5. The choice of the reference point in step 3, stabilizes the algorithm because of the noise influence on the set $F_{\bar{\ell}}^{-1}\left(v_{\bar{i}}^{\bar{\ell}}\right)$. For smaller $\left|v_{\bar{i}}^{\bar{\ell}}\right|$, the noise influence is reduced. A formal explanation can be deduced as follows. For a point $x_0 \in (a_1, a_{n_z})$, for which $y_0 = F_{\bar{\ell}}(x_0)$, let the segment $I \subset [a_1, a_{n_z}]$ be such that $x_0 \in I$ and the restriction function $F_{\bar{\ell}} \mid_I$ has its inverse function $G := F_{\bar{\ell}} \mid_I^{-1}$. We have an estimate:

$$F_{\bar{\ell}} \mid_I^{-1} (y_0 + \delta) - F_{\bar{\ell}} \mid_I^{-1} (y_0) = G^{'}(y_0)\,\delta + \frac{1}{2} G^{''}(y_0)\,\delta^2 + \frac{1}{3!} G^{'''}(y_0)\,\delta^3 + \cdots$$

Using any finite difference approximation of the derivative, we get:

$$\begin{gathered} G^{'}(0) = \min_{y \in I} G^{'}(y) \\ G^{''}(0) = \min_{y \in I} G^{''}(y) \\ \vdots \end{gathered}$$

The goal is to make $\left|F_{\bar{\ell}} \mid_I^{-1} (y_0 + \delta) - F_{\bar{\ell}} \mid_I^{-1} (y_0)\right|$ the smallest, and that is clearly achieved for the smallest $|y_0|$.

Remark 6. It is very important that the positioning algorithm can be implemented in order to dynamically decompress the signal functions on certain segments. The values $F_{\bar{\ell}}^{-1}\left(v_{\bar{i}}^{\bar{\ell}}\right)$ do not need to be calculated for the whole ruler domain, it is enough to do it segment by segment, in order to calculate the suitable fitting norm. Only in the case when the fitting norm is smaller than the smallest, the algorithm remembers the position and the norm, and continues to search for the best fitting norm and its position on the ruler. This justifies why we use piecewise methods of approximation.

Remark 7. The positioning algorithm can function properly on rulers even if some inclination angles or some bases of trapezoids are repeated (remark 1). This result is achieved because the positioning algorithm is not based on the angle detection or the detection of trapezoids.

4.3.1 Speed improvement - Preconditioning

On a certain position of the head, for a reading $V = \left(v_1^1, ..., v_6^1, v_1^2, ..., v_6^2\right)$, we have

$$sign\left(V\right) = \left(1, 1, 1, -1, -1, -1, \ 1, 1, 1, -1, -1\right).$$

We see that the reading head belongs to an odd pole number i, for which it must be satisfied $l_i^1 > 3.2\,\text{mm}$ and $l_i^2 > 4.8\,\text{mm}$, next to the pole $i+1$ for which $l_{i+1}^1 > 4.8\,\text{mm}$ and $l_{i+1}^2 > 1.6\,\text{mm}$.

This gives us a motivation to make a classification of the possible positions, based just on the sign of the input V. That would mean that we do not need to calculate the fitting norm for the whole set $F_{\bar{\ell}}^{-1}\left(v_{\bar{i}}^{\bar{\ell}}\right)$ in step 4 of the algorithm.

For an $V = \left(v_1^1, ..., v_6^1, v_1^2, ..., v_6^2\right)$, let the number n be the number of sign groups of $\left(v_1^1, ..., v_6^1\right)$. More precisely:

$$n = \text{card}\left\{i : \ sign(v_i^1) \neq sign(v_{i+1}^1),\ i = 1, 2, ..., 5\right\} + 1.$$

In the above case $n = 2$. For each sign group, we have numbers $d_1^1, ...d_n^1$ which represent the cardinality of the group:

$$\begin{gathered} d_1^1 = \min\left\{i : \ sign(v_i^1) \neq sign(v_1^1)\right\} - 1, \\ d_2^1 = \min\left\{i : \ sign(v_i^1) \neq sign(v_{d_1^1+1}^1)\right\} - 1, \\ \vdots \\ d_n^1 = \min\left\{i : \ sign(v_i^1) \neq sign(v_{d_{n-1}^1+1}^1)\right\} - 1, \end{gathered}$$

equivalently, for $\left(v_1^2, ..., v_6^2\right)$, we calculate numbers m and d_1^2, d_2^2, ..., d_m^2. Clearly we have $\sum_{i=1}^{n} d_i^1 = 6$ and $\sum_{i=1}^{m} d_i^2 = 6$. Out of the calculated natural numbers, using them as digits we create a number, and we have a mapping $f : \mathbb{R}^{12} \to \mathbb{Z}$,

$$f\left(V\right) = \begin{cases} sign\left(v_1^1\right) \cdot d_1^1 d_2^1 ... d_n^1 d_1^2 d_2^2 ... d_m^2, & sign\left(v_1^2\right) = 1, \\ sign\left(v_1^1\right) \cdot d_1^1 d_2^1 ... d_n^1\, 0\, d_1^2 d_2^2 ... d_m^2, & \text{otherwise.} \end{cases} \tag{15}$$

That number we call "the characteristic number". In above case we have $f\left(V\right) = 3342$. For example for $\text{sign}\left(V\right) = \left(-, -, +, +, -, -, -, +, +, +, -, -\right)$, we get $f\left(V\right) = -2220132$.

Moving the head from the beginning to the end of the ruler we can create an array of k characteristic numbers $\left(c_i\right)_{i=1}^{k}$. The numbers in the array can repeat. For every characteristic number c_i we can assign the number $\sigma\left(i\right)$ of a pole where the head position belongs in the moment of reading. We have well defined set $\mathcal{P} = \left\{\left(c_i, \sigma\left(i\right)\right),\ i = 1, ..., k\right\}$. For every $e \in \left\{c_i,\ i = 1, ..., k\right\}$, we define a non-empty set:

$$s\left(e\right) = \left\{\sigma\left(i\right) : \ c_i = e,\ \left(c_i, \sigma\left(i\right)\right) \in \mathcal{P}\right\}.$$

$s\left(e\right)$ is the set of possible pole positions of the head for a characteristic number e. It can be calculated really fast because the array $\left(c_i\right)_{i=1}^{k}$ can be sorted.

To sum up the method: For every input V, we have a set of possible positions $s\left(f\left(V\right)\right)$, which suggests the pole numbers that potentially hold the head position.

Example. This method is applied on ruler "*random_first_40_1.8_7.8*". We have an array of 296 characteristic numbers. Each characteristic number suggests its possible pole positions. The maximum of suggested position is 5, and the average number of suggested position is 1.5777 (table 2).

This method on average removes a large number of unnecessary inverse function calculations and fitting norm checkings, so it significantly increases the algorithm speed.

pole num.	characteristic num.
1:	66
	651
	5151
	4251
	4242
	42411
	411411
	321411
	...
2:	-2311122
	-2310222
	-22202222
	-22202132
	...

characteristic num.	positions
-131101221:	16
-122101221:	16
-32101221:	8
-31201221:	8
-22201311:	10
-22201221:	8
...	
...	
21210321:	25
22110231:	9
22110321:	9, 25
22111221:	25
23101221:	25
32101221:	33

Table 2: Left: the beginning of the array of characteristic numbers $(c_i)_{i=1}^k$ of ruler "*random_first_40_1.8_7.8*". Right: the beginning and the end of the sorted array $(c_i)_{i=1}^k$ together with its sugested positions.

4.3.2 Precision improvement

In (14) we search for the best position on a discrete set of suggested positions. Because of the noise influence discussed in remark 5, the global minimum exists in some neighborhood of solution x_r. Once x_r is calculated, we solve the problem:

$$N\left(\bar{x}_r\right) = \min\left\{\mathcal{N}\left(x\right) \,:\, x \in \left[x_r - \epsilon, x_r + \epsilon\right]\right\}, \tag{16}$$

and finally $\bar{x}_r - \delta_{\bar{i}}^{\bar{\ell}}$ is the position on the ruler with the best fitting norm on the entire domain.

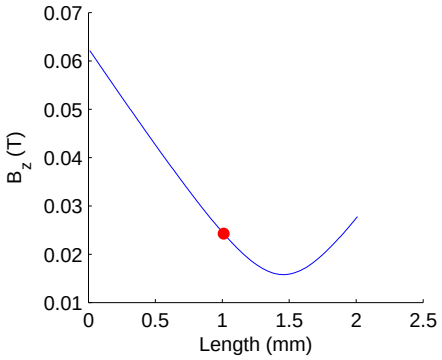

Figure 13: $\mathcal{N}(x)$ on a segment $[x_r - \epsilon, x_r + \epsilon]$, for $\epsilon = 1\,\text{mm}$.

In figure 13 we see a plot of $\mathcal{N}(x)$ in a neighborhood of x_r, so the solution of (16) we can easily achieve numerically with a simple bisection method.

We apply the algorithm on the readings from the beginning of the ruler to its end, with a grid spacing of 0.01 mm. For every position we calculate $N(x_r)$, $N(\bar{x}_r)$ and the error of positioning before and after "precision improvement". As a result, in table 3 we see significant improvement of precision.

	mean error	maximal error
without precision improvement	0.017544	0.117692
with precision improvement	0.008141	0.058876

Table 3: Mean and maximal positioning error in millimeters.

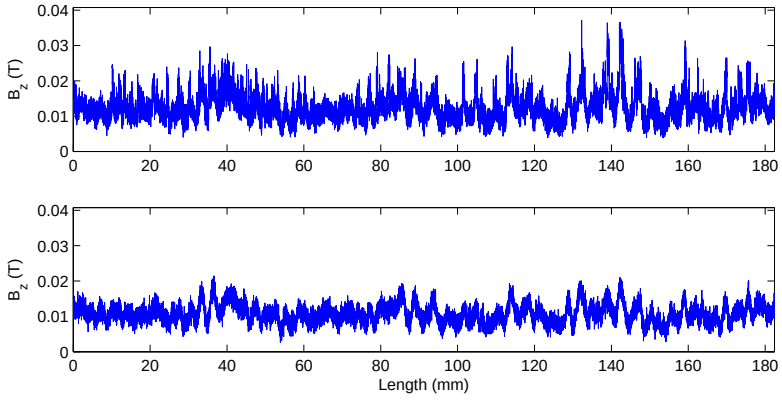

Figure 14: The best fitting norm before and after "precision improvement".

Remark 8. In figure 14 we can see that the best fitting norm is not 0. Equiva-

lently, table 3 show us the error existence. The explanation is the displacement of the Hall sensors together with the added noise.

4.4 Stability of the ruler

One of the paper's objectives is to find out how long a ruler can be. The construction method in section 2 has no length limitation, but it is expected that with the length the uniqueness of the signal readings would be less guaranteed. Since the best fitting norm leads the positioning algorithm to the solution position, the problem can appear in the case when the second best fitting norm is close to the best fitting norm.

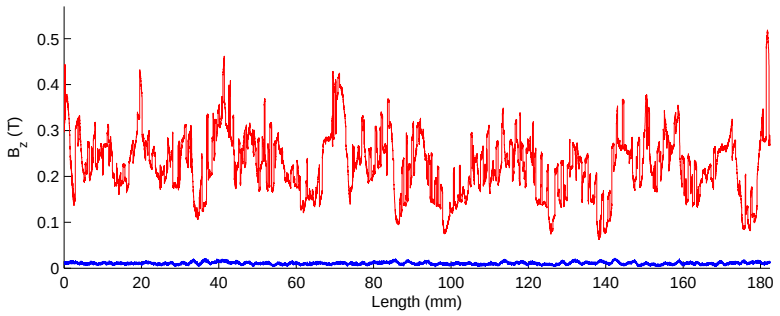

Figure 15: The best (blue graph) and the second best (red graph) fitting norm of the ruler "*random_first_40_1.8_7.8*".

With the experimental setup described in subsection 3.1.3, the best fitting norm (BFN) can reach the value 0.021. The positioning algorithm will work properly if the second best fitting norm ($SBFN$) is above that value (figure 15).

When we construct a ruler, we can not predict the noise and the Hall cells misplacement on the head. Each time we test the functioning of the algorithm will give a slight oscillation of BFN and $SBFN$. However, we need to find a way to classify all the rulers. For the following definition we consider the simulated data set without included misplacement of the Hall cells and added Gaussian noise.

Definition 9. For a simulated data set of a ruler's signal, without included misplacement of the Hall cells and added Gaussian noise, the minimum of $SBFN$ function we call the stability of the ruler.

Ruler "*random_first_40_1.8_7.8*" seems to be stable enough, but it is a short one. Table 4 and figure 16 show what happens with the stability of some longer rulers build with the method from section 2.

Ruler:	length (mm):	$\min SBFN$:
random_1800_1.8_5	6193.273	0.0177
random_1200_1.8_5	4122.665	0.0273
random_600_1.8_5	2054.129	0.0278
random_140_1.8_5	454.706	0.0525

Table 4: The length and the stability of four rulers created with the method from section 2.

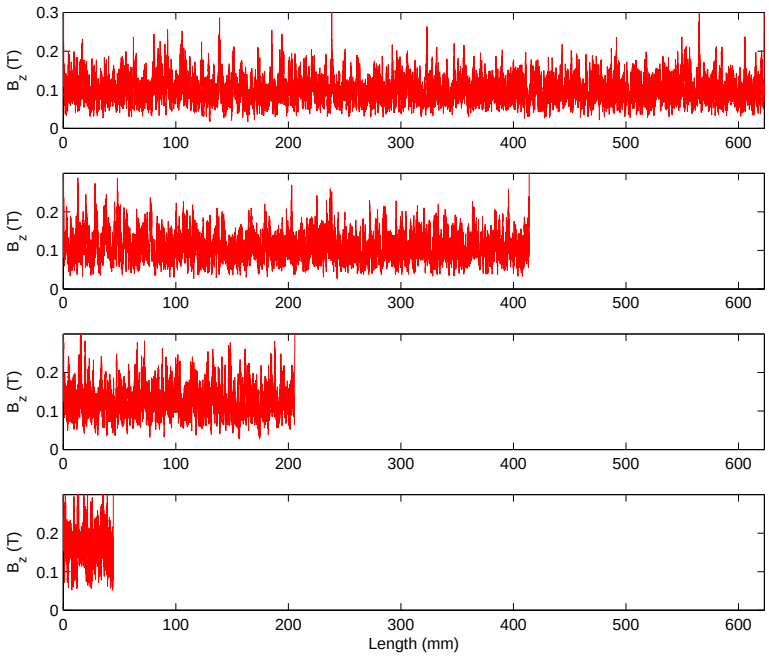

Figure 16: $SBFN$ of the rulers from table 4.

Conclusion

The construction algorithm presented in this paper is able to define a ruler with arbitrary length.

The positioning algorithm successfully solves the positioning problem while consuming a low amount of memory, since the signal is piecewise reconstructed.

The positioning algorithm can work properly on rulers even when an inclination angle is repeated, or when the bases of trapezoids or several consecutive bases of trapezoids are repeated on some different positions of a ruler.

The introduced piecewise approximation method with a linear polynomial combination compared to the piecewise polynomial approximation, gives better results with the same memory consumption. The approximation method can be improved since the problem (13) remains open.

In the construction algorithm one could check the stability of a ruler before extension. That would include the desired stability for a ruler. However, the speed of calculation would take too much time. On a PC with an intel i5 processor with 2.8GHz and 8GB of RAM the calculation of the signal of a three meters long ruler with grid spacing of 0.1 mm takes around one hour using Radia. Having in mind the calculation time, the implementation of this idea seems impossible on a home PC.

The working principle of trapezoid rulers presented in this paper is valid for a different number of Hall sensors on the reading head. Different layouts of Hall sensors, for example placed on more than two reading tracks, can also provide the necessary level of the reading uniqueness of every position. The question that naturally raises is: what is the best Hall sensor layout for a fixed number of Hall sensors? This problem remains open.

For any layout of Hall sensors, placing one additional Hall sensor would increase the difference between the second best fitting norm and the best fitting norm in a point.

Without the noise and the misplacement of the Hall sensors, theoretically for every position the best fitting norm would be zero. We conclude that the potential maximum length of the ruler in industrial production directly depends on factors that influence the best fitting norm function.

Acknowledgment The author would like to thank to Eric Yuan from Helmut Schmidt University for numerous talks and suggestions concerning heuristic algorithms, and to Dr.-Ing. Ildiko Ficza from Faculty of Mechanical Engineering Brno University of Technology, for many suggestions concerning the structure of the paper. The author would also like to thank to Dr. Jörn Hoyer, Dr.-Ing. Torsten Becker and Angelo Cuccato from Bogen Electronic GmbH for introduction with all industrial problems that were modeled and treated in the paper, and to Thomas Lange, Marcus Rose and Torsten Materne from Technical University of Berlin, chair for Electronics and Medical Signal Processing, for suggesting the approximation with piecewise polynomials at the beginning of the topic research. Many special thanks of appreciation to Dr. Armin Fügenschuh from Helmut Schmidt University for giving the suggestion to solve the calibration of the Hall sensors with the regression analysis, for giving many grammar correction and pointing out the parts that needs more detailed explanation, which improved the context of the paper. Approximately one third of the research leading to these results has received funding from the Central Innovation Program SME supported by the German Federal Ministry for Economic Affairs and Energy, under project agreement No. ZIM - KF 3213701 WM3 - „WinPos“.

References

[1] J. Hoyer and T. Becker (2013) Measuring device and a method for measuring the position of bodies, European Patent EP2846126B1.

[2] Software Radia Version 4.29 (2010) European Synchrotron Radiation Facility, Grenoble, France.

[3] Eckart G, Young G (1936) The approximation of one matrix by another of lower rank. Psychometrika 1, Pages 211–218.

[4] I. Markovsky, Low-Rank Approximation: Algorithms, Implementation, Applications, Springer, 2012.

[5] P. Elleaume, O. Chubar, J. Chavanne, "Computing 3D Magnetic Field from Insertion Devices", proc. of the PAC97 Conference May 1997, p.3509-3511.

[6] O. Chubar, P. Elleaume, J. Chavanne, "A 3D Magnetostatics Computer Code for Insertion devices", SRI97 Conference August 1997, J. Synchrotron Rad. (1998). 5, 481-484

[7] Skoog, D. A., Holler, F. J., and Crouch, S. R. (2007) Principles of instrumental analysis, 6th Ed., Belmont, CA, Thomson.

[8] Wolfram Research, Inc., Mathematica, Version 10.2 (2015), Champaign, IL.

[9] A. Fügenschuh, M. Fügenschuh, M. Ludszuweit, A. Mojsic and J. Sokół, Mathematical Model for Absolute Magnetic Measuring Systems in Industrial Applications, 4th International Conference on Mathematical Modeling in Physical Sciences (IC-MSquare2015), 5-8 June 2015, Mykonos, Greece, Journal of Physics: Conference Series 633(1), pp. 12080-12083, D. Vlachos and E. C. Vagenas (Eds.), 2015.

[10] M. Fügenschuh, A. Fügenschuh, M. Ludszuweit, A. Mojsic, J. Sokoł, Mathematical Optimization of a Magnetic Ruler Layout With Rotated Pole Boundaries. To appear in: Proceedings of the International Conference on Operations Research OR2015, Vienna, Sep 1-4, 2015.