Robust spatial approximation of laser scanner point clouds by means of free-form curve approaches in deformation analysis

J. Bureick, H. Alkhatib, I. Neumann
Geodetic Institute,
Leibniz Universität Hannover, Nienburger Str. 1, 30167 Hannover, Germany

Abstract.

In many geodetic engineering applications it is necessary to solve the problem of describing a measured data point cloud, measured, e.g. by laser scanner, by means of free-form curves or surfaces, e.g., with B-Splines as basis functions. The state of the art approaches to determine B-Splines yields results which are seriously manipulated by the occurrence of data gaps and outliers.

Optimal and robust B-Spline fitting depend, however, on optimal selection of the knot vector. Hence we combine in our approach Monte-Carlo methods and the location and curvature of the measured data in order to determine the knot vector of the B-Spline in such a way that no oscillating effects at the edges of data gaps occur. We introduce an optimized approach based on computed weights by means of resampling techniques. In order to minimize the effect of outliers, we apply robust M-estimators for the estimation of control points.

The above mentioned approach will be applied to a multi-sensor system based on kinematic terrestrial laserscanning in the field of rail track inspection.

Keywords. Deformation, free-form curve, B-Splines, knot adjustment, robust parameter estimation, Monte-Carlo resampling techniques

1 Introduction

In several geodetic applications deformations and deflections to a target-state are derived from point clouds, captured, e.g. by laser scanner. In order to determine deformations or deflections, the spatial object has to be modelled. Especially complex objects need to be approximated by free-form curves and surfaces, such as B-Splines, in a sophisticated manner.

Unfortunately, the measurements of the deformed object may contain data gaps and outliers. The state of the art approaches to determine B-Splines yields results which are seriously manipulated by the occurrence of data gaps and outliers. Missing data lead to oscillating effects at the edges of the data gap. Outliers could have an unlimited effect on the results, if the unknown parameters (the control points) are estimated by means of the least-squares methods. Furthermore, the outliers have to be distinguished from "real" deformations and wear marks.

B-Spline fitting usually consists of 3 main steps. First step is the parameterization of the measured data. The second step is the knot adjustment, which yields the knot vector **U**. The third step is the determination of the control points by means of a linear Gauss-Markov-Model (GMM) with the previously determined parameterization and knot vector as input parameters.

The parameterization of the measured data can be achieved using the mentioned methods of Piegl and Tiller (1997): equally spaced, chord length and centripetal. Lai and Lu (1996) introduced an approach to estimate location parameters of the measured points which leads to a non-linear least squares fit.

Knot adjustment for data fitting with B-Splines includes two main tasks. On the one hand the number of knots has to be determined. On the other hand the locations of the knots have to be adjusted.

The former task, a model selection problem, can be solved by applying an information criterion (Akaike or Bayesian, cf. Gálvez et al. (2015)) or the usage of a significance test (cf. Liu and Wang (2004)).

The latter task, an optimization problem, was tackled by plenty of researchers with a vast variety of approaches. Since the first works in the field of Splines in the 1960s and 1970s the optimal choice of the knot locations became important.

However, two problems make the optimal choice of the knot locations difficult. First of all there is no analytic expression for the optimal knot locations and secondly there exist many local optima of the least-squares function (cf. Gálvez et al. (2015); Jupp (1978); Rice (1969)).

Nevertheless, there are approaches to estimate the optimal knot locations. Schmitt and Neuner (2015) try to estimate the knot locations and the position of the control points at the same time. In order to solve the resulting highly non-linear system, they introduce adequate initial values and constraints.

The approaches to align the knot vector to the measured points are well known in many research papers. Piegl and Tiller (1997) align it to the location parameter of the measured points. Park and Lee (2007) align it to the curvature of the measured points.

With the rising capability of information technology artificial intelligence techniques obtain good results in an adequate amount of time. Some approaches use neural and functional networks, respectively. Other approaches use metaheuristic techniques like genetic algorithms (cf. Sarfraz and Raza (2001); Yoshimoto et al. (1999)), artificial immune systems (cf. Gálvez et al. (2015); Ülker and Arslan (2009)) or estimation of distribution algorithms (cf. Zhao et al. (2011)).

As far as we know, the artificial immune system of Gálvez et al. (2015) is the approach which yields the best results in knot adjustment until now, especially for complex data with gaps, discontinuities and cusps. Nevertheless this approach is still time-consuming and CPU-intensive and it is not unusual that the final solution converges into a local optimum instead of the global optimum.

The third and final step in B-Spline fitting is the estimation of the position of the control points. Piegl and Tiller (1997) and Koch (2009) estimate the control points as parameters by means of a linear GMM. The observation vector is formed of the measured data. The design matrix consists of the basis functions. The parameters were determined by minimizing the residual sum of squares. As far as we know there is no work, which describes the usage of robust estimators, like Huber-, Hampel or L1-estimator, instead of the least-square-estimator to determine the position of control points of a B-Spline. Because of the characteristics of laser scan data, like the fast but uncontrolled acquisition of millions of data points, we have to consider the probability for a significant amount of outliers, which seriously affects the results. That is the reason for introducing robust estimators into the estimation of the control points.

The paper is organized as follows: In Section 2 the mathematical essentials for B-Spline curves and the estimation of the position of the control points are briefly described. Section 3 points out the basic properties of robust parameter estimation. Section 4 contains a detailed description of our proposed approach of knot adjustment. In Section 5 the different results of the proposed approach and the robust estimation are presented and compared among each other and with an existing algorithm. This paper finishes with a short conclusion and an outlook in Section 6.

2 B-Spline curves

The functional relation of a B-Spline curve is defined by (cf. Piegl and Tiller (1997)):

$$\mathbf{C}(\bar{u})=\sum_{i=0}^{n} N_{i,p}(\bar{u})\,\mathbf{x}_i. \tag{1}$$

The curve point $\mathbf{C}(\bar{u})=\left|x(\bar{u}),y(\bar{u})\right|^T$ is calculated by a linear combination of the p-th-degree B-Spline basis functions $N_{i,p}(\bar{u})$ with $i\in\{0,...,n\}$ and the control points $\mathbf{x}_i=\left|x_i,y_i\right|^T$, where n+1 is the number of control points. The p-th-degree basis functions can be calculated by a recursion formula (cf. Cox (1972); de Boor (1972))

$$N_{i,0}(\bar{u})=\begin{cases} 1\,if\;u_i\le\bar{u}<u_{i+1} \\ 0\;otherwise \end{cases} \tag{2}$$

$$N_{i,p}(\bar{u})=\frac{\bar{u}-u_i}{u_{i+p}-u_i}N_{i,p-1}(\bar{u})+\frac{u_{i+p+1}-\bar{u}}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1}(\bar{u})$$

where

$$\mathbf{U}=\left|u_0,\ldots,u_m\right| \quad \text{with} \quad u_i\le u_{i+1}, i\in\{0,\ldots,m-1\} \tag{3}$$

The knot vector **U** is a nondecreasing sequence of real numbers. The real numbers u_i are called knots. The first p+1 knot of **U** usually consist of zeros. The last p+1 knots usually consist of ones. m+1 is the size of the knot vector and can be calculated by:

$$m=n+p+1 \tag{4}$$

The parameterization $\bar{u}$ of the (measured) data points is also called location parameter. They are stored in the vector $\mathbf{U}_l=[\bar{u}_1,...\bar{u}_r]$, with r the number of the data points. For example Piegl and Tiller (1997) mention 3 methods (equally spaced, chord length, centripetal) to parameterize the observations. In the proposed approach we used the chord length, which sums up the Euclidean distance between the sorted observations, to parameterize the measured data. Since the parameterization method is not in the focus of this paper, the widely used method chord length was chosen. Nevertheless our approach allows to use the other methods.

In order to fit a B-Spline to measured data **l** the knot vector **U** and the vector with parameterized data points **U**$_l$ are determined previously. In a linear GMM the parameter **x**, the positions of the control points, are estimated. The design matrix **A** is constructed by the basis functions:

$$\mathbf{A} = \begin{bmatrix} N_{0,p}(\bar{u}_1) & \cdots & N_{n,p}(\bar{u}_1) \\ \vdots & & \vdots \\ N_{0,p}(\bar{u}_r) & \cdots & N_{n,p}(\bar{u}_r) \end{bmatrix} \quad (5)$$

3 Robust parameter estimation

Robust estimators include the attribute, that their influence function Ψ is limited. That means that the influence of data with large residuals (possible outliers) on the parameter estimation is limited. The M-estimators, like Huber-, Hampel- or L1-estimator, can be distinguished in their influence function. For example, the influence function of the Huber-estimator weights the residuals of the measured data in the following way (see Eq. 6 and Figure 1):

$$\psi(\varepsilon_i) = \begin{cases} \varepsilon_i & for\,|\varepsilon_i| < c \\ sign(\varepsilon_i)\cdot c & for\,|\varepsilon_i| \geq c \end{cases} \quad (6)$$

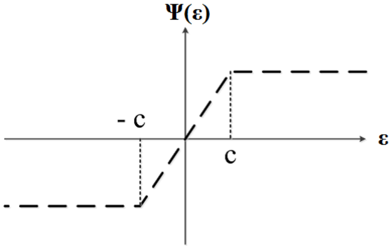

Figure 1: Influence function for Huber-estimator

For the residuals ε_i , which are smaller than the so called tuning constant c, the influence function equals the influence function of the least-squares estimator. The influence of residuals, which are larger than c, is limited to the value of c. For more information see, e.g. Hartung et al. (2009) and Wicki (1998).

By applying these estimators, a non-linear equation system has to be solved. This can be achieved by using the iterative reweighted least squares algorithm illustrated in Figure 2 (cf. Huber (1981), p.179 ff.). After an initial determination of the parameters **x**, the residuals **v** and the variance factor σ (calculated by median absolute deviation (mad) of **v**) with equal weights, the algorithm enters a while-loop which ends when the sum of absolute differences in **v** of 2 consecutive iterations is smaller than a certain threshold near 0. Until then in each iteration new weights p$_i$ are calculated depending on **v** and σ of the previous iteration and depending on the influence function Ψ. The weights p$_i$ are the main diagonal elements of the weight matrix **P**. **x**, **v** and σ are estimated iteratively by means of the updated weights.

```
Data: designmatrix $\mathbf{A}$;
measured data $\mathbf{l}$;
Result: parameter $\mathbf{x}$;
$\mathbf{x}^{(0)} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{l}$;
$\mathbf{v}^{(0)} = \mathbf{A}\mathbf{x}^{(0)} - \mathbf{l}$;
$\sigma^{(0)} = 1,4785 \cdot mad(\mathbf{v}^{(0)})$;
while $|\mathbf{v}^{(k+1)} - \mathbf{v}(k)| \not\approx 0$ do
    $p_i^{(k+1)} = \frac{\Psi(\frac{v_i^{(k)}}{\sigma^{(k)}})}{\frac{v_i^{(k)}}{\sigma^{(k)}}} \rightarrow \mathbf{P}^{(k+1)} = diag([p_1^{(k+1)}, ..., p_n^{(k+1)}])$;
    $\mathbf{x}^{(k+1)} = (\mathbf{A}^T\mathbf{P}^{(k+1)}\mathbf{A})^{-1}\mathbf{A}^{(T)}\mathbf{P}^{k+1}\mathbf{l}$;
    $\mathbf{v}^{(k+1)} = \mathbf{A}\mathbf{x}^{(k+1)} - \mathbf{l}$;
    $\sigma^{(k+1)} = 1,4785 \cdot mad(\mathbf{v}^{(k)})$;
end
```

Figure 2: Algorithm iterative reweighted least squares

4 Methodology of knot adjustment

Our methodology is illustrated in Figure 3. Before the algorithm starts, the number of control points *n* and the degree *p* of the basis function have to be chosen. As already mentioned in Section 1 this is a model selection problem which can be solved afterwards by applying an information criterion or a significance test to different solutions with a diverging number of control points or degree of basis function. This is not part of this work.

Also the maximal number of iterations *itermax* has to be chosen. At the moment *itermax* (in this case: 20,000) serves as stop-criterion of our algorithm.

Our algorithm offers 3 possible methods (“location”, “curvature”, “ranking”) to calculate the probability **R**. All methods are described in Section 4.1. The probability calculating method *pcm* stores the selected method and has to be chosen previously. In case of selection of the method “ranking” the number of iterations with equally weighted **R** *iterchance* has to be chosen (see Section 4.1). At the beginning the measured data **l** has to be parameterized (using chord length). The resulting location parameter are stored in **U**$_l$.

In case of choosing “location” as *pcm*, **R** has to be calculated depending on **U**$_l$. In case of choosing "curvature" as *pcm*, the curvature **cur** of the measured data has be determined and subsequently **R** has to be obtained depending on **cur** and $\mathbf{U}_l$.

```
Data: measured data (sorted) l;
# control points n;
degree basis function p;
# iterations itermax;
choice of probability calculating method pcm;
# iterations with equally weighted R iterchance
Result: optimal position of internal knots in the
        knot vector $\mathbf{U}_{best}$
initialization;
determine the location parameter $\mathbf{U}_l$ of l;
if pcm = "location" then
    calculate probability R depending on $\mathbf{U}_l$
else
    if pcm = "curvature" then
        calculate the curvature cur of l;
        calculate R depending on cur and $\mathbf{U}_l$;
    end
end
for iter = 1 : itermax do
    if pcm = "ranking" then
        if iter $\leq$ iterchance then
            calculate R equally weighted
        else
            calculate R depending on $\mathbf{U}_{ranking}$
        end
    end
    Choose the n-p internal knots $\mathbf{U}_{internal}$ randomly
    depending on R;
    Arrange $\mathbf{U}_{internal}$ to the knot vector $\mathbf{U}_{actual}$;
    Estimate control points with a GMM depending
    on $\mathbf{U}_l$ and $\mathbf{U}_{actual}$;
    Calculate the residual sum of squares $\Omega_{actual}$ ;
    if $\Omega_{actual} < \Omega_{ranking}$ then
        $\Omega_{ranking} = \Omega_{actual}$;
        $\mathbf{U}_{ranking} = \mathbf{U}_{actual}$;
    end
end
Choose $\mathbf{U}_{ranking}$ with smallest $\Omega_{ranking}$ as $\mathbf{U}_{best}$, $\Omega_{best}$;
for iter = p+1:n do
    for $U_{best}(1, iter)$ = 0:0.001:1 do
        Arrange the complete knot vector $\mathbf{U}_{actual}$;
        Estimate control points with a GMM
        depending on $\mathbf{U}_l$ and $\mathbf{U}_{actual}$;
        Calculate the residual sum of squares $\Omega_{actual}$;
        if $\Omega_{actual} < \Omega_{best}$ then
            $\Omega_{best} = \Omega_{actual}$;
            $\mathbf{U}_{best} = \mathbf{U}_{actual}$
        end
    end
end
```

Figure 3: Methodology of knot adjustment

At the beginning of the following for-loop there is, in case of choosing "ranking" as *pcm*, an inquiry which checks if the actual number of iteration is lower or equal *iterchance*. In this case **R** is calculated equally weighted. If the actual number of iteration is larger than *iterchance* **R** is calculated depending on $\mathbf{U}_{ranking}$.

In the following step, all *n-p* internal knots $\mathbf{U}_{\mathrm{internal}}$ are chosen randomly, but depending on **R**. $\mathbf{U}_{internal}$, together with the multiple start- and end-knots, has to be arranged to the complete knot vector $\mathbf{U}_{actual}$ in a non-decreasing way. For the choice of the knots, see Section 4.2.

In the next step the control points are estimated in a GMM by using $\mathbf{U}_l$ and $\mathbf{U}_{actual}$. It is possible to use a least squares estimator as well as a robust estimator. For this solution the residual sum of squares Ω_{actual} is calculated. Ω_{actual} has to be compared with the Ω stored in the ranking $\Omega_{ranking}$. When Ω_{actual} is smaller than one or more $\Omega_{ranking}$, $\mathbf{U}_{\mathrm{actual}}$ and Ω_{actual} are stored in the ranking and the result with the highest Ω in the ranking will be deleted. These steps are repeated until *itermax* is reached.

When *itermax* is reached the knot vector $\mathbf{U}_{best}$ with the smallest Ω_{best} is chosen out of $\mathbf{U}_{ranking}$ and $\Omega_{ranking}$ and each internal knot of $\mathbf{U}_{best}$ is sequentially modified and stored as $\mathbf{U}_{best}$ and Ω_{best} when the resulting Ω_{actual} is smaller than Ω_{best}. After modifying each internal knot $\mathbf{U}_{best}$ and Ω_{best} are obtained and the algorithm ends.

4.1 Calculation of the probability

As already mentioned in Section 4 and depicted in Figure 3 we introduce 3 methods to calculate the probability **R**.

The first method "location" calculates the probability out of the parameterized location parameter $\mathbf{U}_{\mathrm{l}}$ of the measured data **l**.

First of all, the possible span of the internal knots (in this case the span ranged from 0 to 1, because of the parameterization of the measured data) is divided in many (in this case: 1000) parts. For each part where the mean distance to the next 2 location parameters exceeds a certain threshold (in this case: 0.02) the probability of this part is set to 0. In the other case the probability is set to 1. As a consequence the internal knots can only be chosen in areas where measured data is nearby. That means that internal knot spans are extreme unlikely to be located in data gaps, which has, due to possible singularities in the design matrix, negative effects on the appearance of the B-Spline.

The second method "curvature" calculates the probability depending on curvature values of the measured data points. The calculation is similar to the calculation of the method "location" with the difference that the probability of the parts lying under the threshold is calculated as the mean curvature of the 5 closest measured data points.

The third method “ranking” calculates the probability depending on a ranking list $\mathbf{U}_{ranking}$ of the knot vectors with the smallest sum of squares $\boldsymbol{\Omega}$. For the first iterations (in this case: *iterchance* = 3000) the whole knot span is weighted equally. Otherwise the unwished chance that the algorithm converges to a local optimum increases. The knot vectors with the smallest $\boldsymbol{\Omega}$ are stored in $\mathbf{U}_{ranking}$ (in this case $\mathbf{U}_{ranking}$ consist of the top 20 knot vectors). For an iteration number larger *iterchance* the probability is calculated depending on $\mathbf{U}_{ranking}$. For each part, where an internal knot of $\mathbf{U}_{ranking}$ is placed, the probability for that part is increased inversely proportional to $\boldsymbol{\Omega}$. As a consequence the probability for choosing parts increases, where a good solutions was achieved. In order to solve the problem that the algorithm converges to a local optimum, some knot vectors were still chosen with an equally weighted probability. Using the method “ranking”, our algorithm transforms into an evolutionary strategy.

4.2 Choice of the knots

In order to determine knots out of the calculated probabilities **R** a resampling step, established for particle filter, is introduced (cf. Simon (2006), pp. 466f.). In step 1 *n-p* random numbers are generated uniformly distributed on [0,1]. In step 2 the probabilities of the 1000 parts are accumulated and stored for each part (see Figure 4).

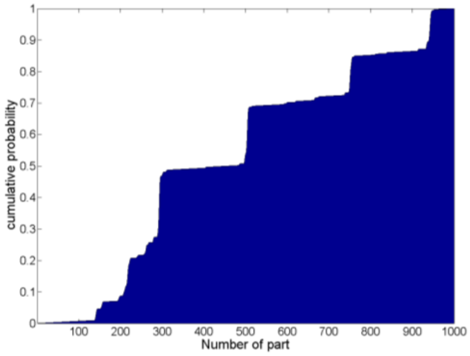

Figure 4: Cumulative probability in method “ranking” (after last iteration)

Finally, the part where the accumulated probability is greater than the randomly chosen number is chosen. That means that parts with low probabilities are unlikely to be chosen as the new internal knot.

5 Results

5.1 Knot adjustment

In order to verify the capability of our algorithm it is applied to several test functions. Yoshimoto et al. (1999) and Gálvez et al. (2015) used the following functions (Eq. 7-9) which represent complex data with discontinuities and cusps.

The first function represents a step function (see Figure 5):

$$\varphi_1(\omega) = \frac{90}{1+e^{-100(\omega-0.4)}} \tag{7}$$

The second function contains a discontinuity (comparable to a data gap, see Figure 6):

$$\varphi_2(\omega) = \begin{cases} \dfrac{1}{0.01+(\omega-0.3)^2} & for\ \omega < 0.6 \\ \dfrac{1}{0.015+(\omega-0.65)^2} & for\ \omega \geq 0.6 \end{cases} \tag{8}$$

The third function contains a cusp (see Figure 7):

$$\varphi_3(\omega) = \frac{100}{e^{|10\omega-5|}} + \frac{(10\omega-5)^5}{500} \tag{9}$$

For each test function 201 data points are generated using the Uniform distribution within the interval *U*~[0,1]. All data points are perturbed by an additive random noise that follows the normal distribution *N*~[0,1] (cf. Gálvez et al. (2015), p. 96f.).

In the following Tables 1 to 3 $\boldsymbol{\Omega}_{\text{best}}$ of the different probability calculating methods of the proposed approach are compared to $\boldsymbol{\Omega}_{\text{best}}$ of the implemented clonal selection algorithm (csa) of Gálvez et al. (2015). κ is the number of internal knots and can be calculated according to Eq. 10.

$$\kappa = n - p \tag{10}$$

For reason of comparability, $\boldsymbol{\Omega}_{\text{best}}$ is calculated as average of 30 runs, without regarding the 5 best and worst runs (cf. Gálvez et al. (2015), p.98). The mentioned standard deviation σ is calculated out of all 30 runs.

Table 1. $\boldsymbol{\Omega}_{\text{best}}$ (±σ) for function $\varphi_1(\omega)$ from κ=1 to κ=7

κ	location	curvature	ranking	csa
1	**2896.99** **(±0.00)**	**2896.99** **(±0.00)**	**2896.99** **(±0.00)**	**2896.99** **(±0.00)**
2	651.46	651.19	**606.02**	**606.02**

	(±0.78)	(±0.90)	**(±0.00)**	**(±0.00)**
3	272.25	272.39	271.53	**271.09**
	(±0.89)	(±0.76)	**(±0.01)**	(±1.88)
4	229.81	229.71	228.19	**225.84**
	(±1.67)	(±1.76)	**(±0.07)**	(±12.24)
5	217.61	216.45	**215.74**	219.24
	(±1.68)	**(±1.06)**	(±1.22)	(±6.51)
6	172.60	171.29	170.08	**169.69**
	(±2.79)	(±1.19)	**(±0.40)**	(±0.73)
7	166.86	165.60	**163.51**	166.03
	(±1.79)	(±1.54)	**(±0.50)**	(±2.95)

Table 1 to 3 show that the method “ranking” always yields a smaller mean value for Ω_{best} than the methods “location” and “curvature”. In comparison to the csa the method “ranking” provides in the majority of the cases slightly smaller mean values for Ω_{best}. In the vast majority of the cases the standard deviation σ of the method “ranking” is significantly smaller than σ of the csa. Figure 5 to 7 show the solutions with the smallest Ω_{best} over 30 runs for the method “ranking”. The best results of the other methods weren’t displayed because the visual differences are too small.

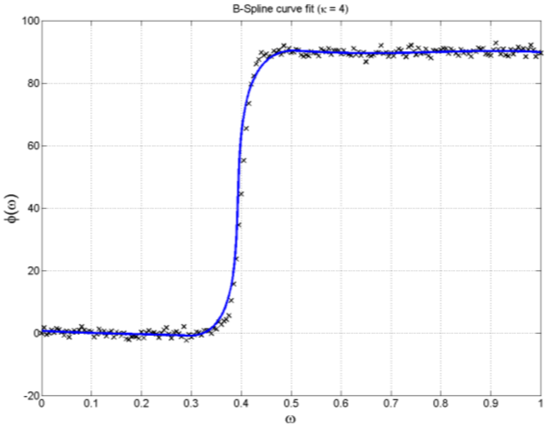

Figure 5: Best 3rd-order B-Spline fitting for function $\varphi_1(\omega)$ with method “ranking” ($\kappa = 4$)

Table 2. Ω_{best} (±σ) for function $\varphi_2(\omega)$ from κ=1 to κ=17

κ	location	curvature	ranking	csa
1	**43262.28**	**43262.28**	**43262.28**	**43262.28**
	(±0.00)	**(±0.00)**	**(±0.00)**	**(±0.00)**
2	23595.93	23595.72	**23595.57**	**23595.57**
	(±1.12)	(±0.27)	**(±0.00)**	(±161.74)
3	1771.39	1747.55	**1643.68**	1643.69
	(±20.18)	(±11.84)	**(±0.00)**	(±0.01)
4	1050.54	1038.23	998.13	**995.18**
	(±23.47)	(±27.82)	(±6.96)	**(±2.86)**
5	834.02	818.52	**772.96**	778.14
	(±33.83)	(±16.75)	**(±11.82)**	(±40.79)
6	466.34	438.82	**354.27**	424.73
	(±94.50)	(±69.66)	**(±6.88)**	(±160.59)
7	207.52	220.28	157.59	**153.52**
	(±28.80)	(±24.51)	**(±5.18)**	(±7.98)
8	162.83	166.93	**139.54**	141.07
	(±14.25)	(±16.26)	**(±3.80)**	(±7.11)
9	137.11	141.92	**124.49**	132.51
	(±8.30)	(±10.77)	**(±3.45)**	(±9.32)
10	125.14	131.93	**112.65**	124.10
	(±6.49)	(±9.05)	**(±5.19)**	(±9.01)
11	115.70	122.42	**103.17**	118.86
	(±7.04)	(±8.50)	**(±3.87)**	(±9.08)
12	108.01	113.30	**99.64**	112.75
	(±5.90)	(±6.34)	**(±1.43)**	(±10.71)
13	103.60	106.93	**97.03**	104.39
	(±4.30)	(±6.60)	**(±1.48)**	(±10.22)
14	97.23	101.35	**93.22**	96.55
	(±3.10)	(±5.25)	(±3.15)	(±6.23)
15	94.39	95.35	**90.39**	93.26
	(±2.89)	(±3.72)	(±3.62)	(±8.06)
16	91.49	93.56	**85.04**	90.94
	(±3.11)	(±3.94)	(±4.11)	(±6.11)
17	89.47	89.45	**82.21**	87.44
	(±3.95)	**(±3.47)**	(±3.85)	(±5.06)

That means the results of the method “ranking” are more stable than the results of the csa, which converge, in a not negligible amount of runs, into a local optimum instead of the global optimum. Especially for large κ our algorithm yields better results than the csa.

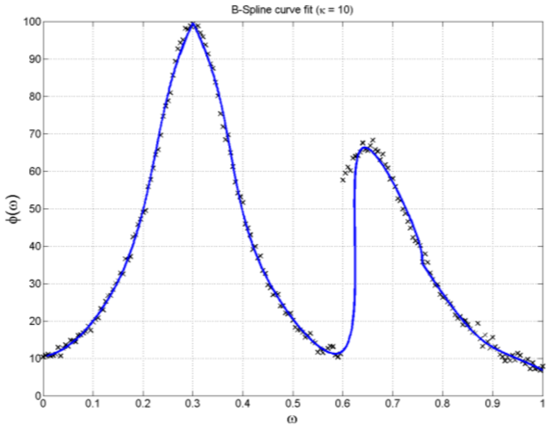

Figure 6: Best 3rd-order B-spline fitting for function $\varphi_2(\omega)$ with method “ranking” ($\kappa = 10$)

Table 3. Ω_{best} (±σ) for function $\varphi_3(\omega)$ from κ=1 to κ=17

κ	location	curvature	ranking	csa
1	4912.60	4912.59	**4912.55**	**4912.55**
	(±0.01)	**(±0.00)**	**(±0.00)**	**(±0.00)**
2	2672.91	2672.93	**2672.87**	**2672.87**
	(±0.15)	(±0.25)	**(±0.00)**	**(±0.00)**

3	594.68 (±1.03)	594.39 (±0.87)	**593.56** **(±0.05)**	593.57 (±1.73)
4	441.67 (±3.69)	439.69 **(±2.59)**	438.07 (±8.34)	**437.60** (±10.45)
5	262.24 (±10.10)	258.80 (±6.57)	248.95 **(±0.26)**	**242.85** (±1.49)
6	231.87 (±8.23)	226.09 (±4.85)	**215.02** **(±0.72)**	221.91 (±7.19)
7	205.72 (±6.19)	202.51 (±5.33)	**191.65** **(±0.82)**	199.57 (±10.99)
8	185.58 (±6.78)	186.30 (±5.07)	**168.72** **(±1.41)**	177.19 (±10.87)
9	171.85 (±3.84)	170.65 (±4.45)	**161.09** **(±0.57)**	164.67 (±10.25)
10	161.70 (±4.24)	162.02 (±3.77)	**150.88** **(±1.18)**	154.83 (±6.82)
11	155.04 (±3.36)	155.04 (±4.40)	**145.52** **(±2.20)**	149.20 (±5.15)
12	148.17 (±2.92)	149.29 (±3.95)	**142.10** **(±1.01)**	142.31 (±5.28)
13	145.84 (±2.07)	146.21 (±2.55)	**138.26** **(±1.93)**	142.04 (±4.89)
14	143.15 (±4.09)	143.15 (±3.87)	**128.90** **(±3.48)**	138.55 (±5.05)
15	139.71 (±3.96)	141.00 (±3.76)	**123.14** **(±1.83)**	133.35 (±6.70)
16	132.76 (±4.41)	133.76 (±4.67)	**121.37** **(±1.18)**	128.22 (±7.61)
17	128.99 (±4.00)	127.85 (±4.29)	**118.52** **(±2.98)**	126.58 (±6.85)

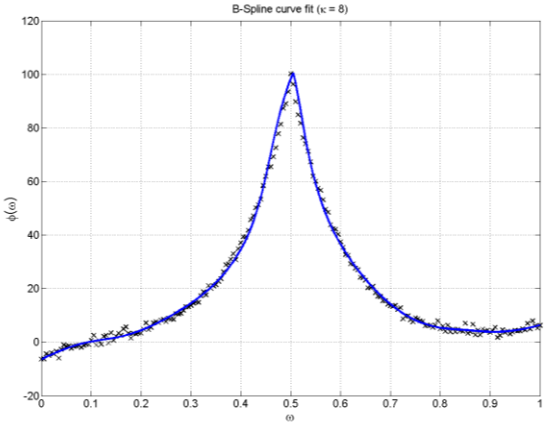

Figure 7: Best 3rd-order B-spline fitting for function φ3(ω) with method "ranking" (κ = 8)

5.2 Robust parameter estimation

The results shown in the previous Section 5.1 are obtained by a least squares estimation of the control points. Due to the fact that the test functions are perturbed by a normally distributed noise that is sufficient. In order to check the performance of robust estimation, we generated a point cloud of 911 points out of the desired values for a rail track. These desired values or true values, respectively, are stored in the vector $\tilde{\mathbf{l}}$. Again $\tilde{\mathbf{l}}$ is perturbed by a normally distributed noise *N*~[0,0.067], which conforms with the data sheet of a usual profile scanner for rail track inspection. Additionally, 3 outliers are inserted, arranged about 4 mm above the desired values. The resulting points are stored in **l**. Figure 8 shows the comparison between an estimation with least squares and the estimation using a robust estimator, in this case the Huber-estimator (with the tuning constant c = 1). Both results are obtained using the method "ranking" for knot adjustment. In the area of the 3 outliers the least squares estimation is distorted in the direction of the outliers, whereas the fit using the Huber-estimator stays closer to the data without outliers.

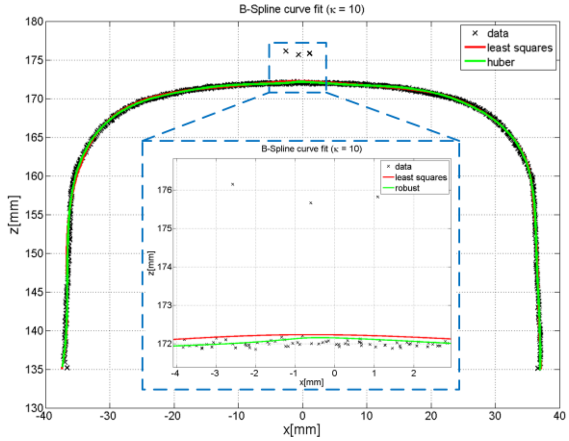

Figure 8: Comparison estimation with least squares and Huber

In order to validate that visual result, we calculated the residual sum of squares of the estimated parameters to the "measured" values **l** (Ω) and to the true values $\tilde{\mathbf{l}}$ ($\tilde{\Omega}$).

$$\begin{aligned}\mathbf{v}&=\mathbf{A}\cdot\hat{\mathbf{x}}-\mathbf{l} \rightarrow \Omega=\mathbf{v}^T\mathbf{v}\\ \tilde{\mathbf{v}}&=\mathbf{A}\cdot\hat{\mathbf{x}}-\tilde{\mathbf{l}} \rightarrow \tilde{\Omega}=\tilde{\mathbf{v}}^T\tilde{\mathbf{v}}\end{aligned} \qquad (11)$$

Table 4. Comparison of least squares and Huber-estimation

	least squares	Huber
Ω	86.0781	86.9692
$\tilde{\Omega}$	43.6250	41.8616

Table 4 shows that the estimation using the Huber-estimator has a smaller $\tilde{\Omega}$ thus the effect of the outliers is lower than using the least squares estimation.

6 Conclusion and Outlook

In this paper we introduced an algorithm which determines the knot vector of a B-Spline with a mixture of Monte-Carlo methods and an evolutionary algorithm and simultaneously is robust against outliers. The results of knot adjustment are slightly better than the results of comparable algorithms. Especially for an increasing number of internal knots our algorithm yields better results. Furthermore the proposed algorithm produces more stable results, because the deviation of the results is significantly smaller. We also showed that robust parameter estimation for B-Splines obtains good results and is essential in case of an outlier-contaminated point cloud.

Beside the extension of our approach on B-Spline surfaces, the input and tuning parameters, as the chosen assumptions at probability calculation (e.g. size of the ranking, thresholds and partitioning), are going to be implemented in a closed loop simulation. Especially, the extension with respect to a more sophisticated introduction of prior knowledge is planned.

Acknowledgement

A large part of the shown results were developed in the research project “Advanced Rail Track Inspection System (ARTIS)” supported by the Federal Ministry for Economic Affairs and Energy on the basis of a decision by the German Bundestag.

References

Cox, M.G., 1972. The Numerical Evaluation of B - Splines. IMA J Appl Math 10 (2), 134–149. doi:10.1093/imamat/10.2.134.

de Boor, C., 1972. On calculating with B-splines. Journal of Approximation Theory 6 (1), 50–62. doi:10.1016/0021-9045(72)90080-9.

Gálvez, A., Iglesias, A., Avila, A., Otero, C., Arias, R., Manchado, C., 2015. Elitist clonal selection algorithm for optimal choice of free knots in B-spline data fitting. Applied Soft Computing 26, 90–106. doi:10.1016/j.asoc.2014.09.030.

Hartung, J., Elpelt, B., Klösener, K.-H., 2009. Statistik: Lehr- und Handbuch der angewandten Statistik, 15th ed. Oldenbourg, München, XXXIII, 1145.

Huber, P.J., 1981. Robust statistics. Wiley, New York, ix, 308.

Jupp, D.L., 1978. Approximation to Data by Spline with Free Knots. SIAM Journal on Numerical Analysis 15 (2), 328–343.

Koch, K.-R., 2009. Fitting Free-Form Surfaces to Laserscan Data by NURBS. Allgemeine Vermessungs-Nachrichten (AVN) (4), 134–140.

Lai, J.-Y., Lu, C.-Y., 1996. Reverse Engineering of Composite Sculptured Surfaces. Int J Advanced Manufacturing Technology (12), 180–189.

Liu, A., Wang, Y., 2004. Hypothesis testing in smoothing spline models. Journal of Statistical Computation and Simulation 74 (8), 581–597. doi:10.1080/00949650310001623416.

Park, H., Lee, J.-H., 2007. B-spline curve fitting based on adaptive curve refinement using dominant points. Computer-Aided Design 39 (6), 439–451. doi:10.1016/j.cad.2006.12.006.

Piegl, L., Tiller, W., 1997. The NURBS book, 2nd ed. Springer, Berlin, New York, xiv, 646.

Rice, J.R., 1969. The Approximation of Function, 2nd ed. Addison-Wesley Publishing Company, Reading, MA.

Sarfraz, M., Raza, S.A., 2001. Capturing outline of fonts using genetic algorithm and splines, in: Proceedings. IEEE Computer Soc., Los Alamitos, Calif. [et al.], pp. 738–743.

Schmitt, C., Neuner, H., 2015. Knot estimation on B-Spline curves. Österreichische Zeitschrift für Vermessung & Geoinformation 103 (2+3), 188–197.

Simon, D., 2006. Optimal state estimation: Kalman, H [infinity] and nonlinear approaches. Wiley-Interscience, Hoboken, N.J., xxvi, 526.

Ülker, E., Arslan, A., 2009. Automatic knot adjustment using an artificial immune system for B-spline curve approximation. Information Sciences 179 (10), 1483–1494. doi:10.1016/j.ins.2008.11.037.

Wicki, F., 1998. Robuste Schätzverfahren für die Parameterschätzung in geodätischen Netzen. Institut für Geodäsie und Photogrammetrie, Eidgenössische Technische Hochschule, Zürich, 185.

Yoshimoto, F., Moriyama, M., Harada, T., 1999. Automatic knot placement by a genetic algorithm for data fitting with a spline, in: Proceedings of the international conference on shape modeling and applications, pp. 162–169.

Zhao, X., Zhang, C., Yang, B., Li, P., 2011. Adaptive knot placement using a GMM-based continuous optimization algorithm in B-spline curve approximation. Computer-Aided Design 43 (6), 598–604. doi:10.1016/j.cad.2011.01.015.