J Geod (2005) 79: 1–10
DOI 10.1007/s00190-005-0437-2

ORIGINAL ARTICLE

A. Borkowski · W. Keller

Global and local methods for tracking the intersection curve between two surfaces

Received: 20 January 2004 / Accepted: 14 January 2005 / Published online: 25 April 2005

Abstract At present, the modelling of terrain edges from discrete data clouds $\{x, y, z\}$ is one of the 'hot topics' in the processing of laser scanning data. This paper proposes two different methods for the three-dimensional modelling of terrain edges. Common to both methods is the idea to describe the terrain edge as the intersection line of two surface patches $z_i = z(x, y), i = 1, 2$. The first method is based on numerical integration of a differential equation describing the intersection line. The second method uses the snakes algorithm for the identification of the terrain edge. Both methods are tested for synthetic and real-world data examples, which shows that they are suitable for the practical extraction of edges from laser scanning data.

Keywords Surface discontinuities · Intersection curve · Numerical integration · Snakes

1 Introduction and motivation

In recent years, laser scanning has significantly changed the process of data collection for the modelling of topographic, and other, surfaces. The enormous density of discrete information provided by laser scanning, which have the form of irregularly sampled point clouds $\{x, y, z\} \subset \mathbb{R}^3$ open new possibilities for the modelling of surfaces. Due to the irregular data distribution, the problem of modelling surfaces with discontinuities in the first-order derivatives can even be tackled.

In Borkowski and Keller (2003) an approach was proposed, which was based on a variational principle. The parameters of the variational method are locally controllable and therefore allow the preservation of surface discontinuities during the modelling process. Nevertheless, the method does not provide vectorial information about the location of those discontinuities. For many applications, vectorial information is indispensable such as digital terrain models for disaster management in flooded areas.

At present, the extraction of vectorial discontinuity information is mainly used for the extraction of terrain edges from laser scanning point clouds. In these methods, the originally irregularly sampled data are interpolated to a regular quadratic grid and the height information is converted into grey-scale values. The resulting height image is treated by standard methods of digital image processing for edge detection (Brügelmann 2000; Sui 2002).

In contrast to this digital image processing approach, the direct use of the laser scanner data allows three-dimensional modelling of terrain edges. First results for the three-dimensional edge detection are presented in the works of Kraus and Pfeifer (2001) and Briese et al. (2002). The method described in this paper starts from the two-dimensional edge location information, which was obtained by digital image processing methods. This two-dimensional model K divides the original data into two classes. One class consists of all points on the left side of K, and the other consists of all points on the right side of K.

Both classes are then extended by points of the other class, which are in the close neighbourhood of the dividing edge K. Sliding along K from each subset, a smaller patch $P_i \subset \mathbb{R}^3$, $i = 1, 2$ is selected. For each patch P_i, an approximating plane E_i is computed. The intersection $K := E_1 \cap E_2$ coincides with the local tangent vector to the three-dimensional terrain edge. The alignment of all these tangent vectors gives the three-dimensional terrain edge model. This process is repeated for every patch, when sliding along the preliminary two-dimensional edge model K.

In a second computation step the resulting intersection lines t are used for a reclassification of the points of the data set into 'left' and 'right' of K. If necessary, this process can be iterated. The tangent vector t gives only the direction but not the location of the terrain edge. Therefore, a point representer in the centre of each patch is chosen. These point

A. Borkowski (✉)
Department of Geodesy and Photogrammetry,
Agricultural University of Wroclaw,
Grunwaldzka 53, PL-50-357 Wroclaw, Poland
E-mail: borkowski@ar.wroc.pl

W. Keller
Geodätisches Institut, Universität Stuttgart,
Geschwister-Scholl-Strasse 24/D, D-70174 Stuttgart, Germany
E-mail: wolfgang.keller@gis.uni-stuttgart.de

representers are then connected by a spline, thus delivering the three-dimensional vectorial description of the terrain edge.

The paper aims to generalise the idea presented in (Kraus and Pfeifer 2001; Briese et al. 2002). Instead of finding the terrain edge as the intersection of sliding planes, here the terrain edge is to be found as the intersection of two curved surfaces. This requires that, in a pre-processing step, data blunders are eliminated and that a rough classification of the data into the two sides of the terrain edge has been carried out.

Finding the intersection of surfaces (or space-curves) belongs to the standard techniques of computer graphics and visualization. Hoschek and Lasser (1992) give an excellent overview of the available algorithms. Algebraic methods, which for simple surfaces provide closed solutions of the intersection problem, are of particular importance in the field of computer graphics. Unfortunately, they are not applicable to our problem since the analytic description of the surfaces E_1, E_2 is far too complicated for algebraic solution techniques.

In this paper, two new approaches are presented, which are based on the usual explicit terrain representation of digital terrain models. The first of the two methods is the implementation of a numerical integrator for the solution of an exact differential equation describing the terrain edge. The second uses a deformable curve model (snake) in order to find the projection of the terrain edge into the xy plane. Both methods are numerically tested and their performances are compared.

2 Intersection curve of two surfaces

Let us assume that the original data have already been classified into points to the 'left' and points to the 'right' of the edge to be determined. This can be done using methods of digital image processing. The data from each class are now approximated by surfaces, having the explicit expressions $z = f_i(x, y)$; $i = 1, 2$. The terrain edge is the intersection curve $[x(s), y(s), z(s)]^\top$ of the two surfaces.

Hence, the problem of finding the three-dimensional edge representation can be reduced to the problem of finding the projection of this curve into the xy coordinate plane. The z-component of the three-dimensional intersection line is obtained by inserting the projection into one of the surfaces f_1 or f_2

$$[x(s), y(s), z(s]]^\top = [x(s), y(s), f_i(x(s), y(s))]^\top, \quad i = 1, 2 \tag{1}$$

Obviously, along the projection $[x(s), y(s)]^\top$ of $[x(s), y(s), z(s)]^\top$ in the xy-plane

$$F(x(s), y(s)) := f_2(x(s), y(s)) - f_1(x(s), y(s)) = 0 \tag{2}$$

holds. This condition (Eq. 2) can be reformulated as an ordinary differential equation (ODE)

$$\frac{\mathrm{d}}{\mathrm{d}s}F(x(s), y(s)) = F_x(x(s), y(s))\frac{\mathrm{d}x}{\mathrm{d}s} + F_y(x(s), y(s))\frac{\mathrm{d}y}{\mathrm{d}s} = 0 \tag{3}$$

Hence, the equations

$$\frac{\mathrm{d}y}{\mathrm{d}x} = -\frac{F_x(x, y)}{F_y(x, y)}, \quad F_y \neq 0 \tag{4}$$

$$\frac{\mathrm{d}x}{\mathrm{d}y} = -\frac{F_y(x, y)}{F_x(x, y)}, \quad F_x \neq 0. \tag{5}$$

hold.

3 A line-tracking algorithm

3.1 Second-order integrator

If an initial point (x_0, y_0) is given, the projection $[x(s), y(s)]^\top$ can be computed either as a solution of the ODE in Eq. (4), or as a solution of Eq. (5). In the first case, y as a function of x and in the second case x as a function of y is obtained.

In general, it will be necessary to switch between Eqs. (4) and (5) if $F_x \approx 0$ or $F_y \approx 0$ holds, which can lead to a loss of continuity of the curve $[x(s), y(s)]^\top$. This difficulty can be overcome in the following way: An equivalent representation of Eq. (3) is

$$F_x\frac{\mathrm{d}x}{\mathrm{d}s} + F_y\frac{\mathrm{d}y}{\mathrm{d}s} = 0 \tag{6}$$

$$\mathrm{d}x^2 + \mathrm{d}y^2 = \mathrm{d}s^2. \tag{7}$$

Equation (6) is clearly equivalent to the first-order differential equation system

$$\begin{bmatrix} \frac{\mathrm{d}x(u)}{\mathrm{d}u} \\ \frac{\mathrm{d}y(u)}{\mathrm{d}u} \end{bmatrix} = \begin{bmatrix} F_y(x(u), y(u)) \\ -F_x(x(u), y(u)) \end{bmatrix}, \tag{8}$$

which describes the direction of the tangent vector in every point of the intersection line. In this form, the problem has been treated frequently in literature (Ascher and Petzold 1998; Hairer et.al. 2002). Nevertheless, in this form the curve parameter u can be chosen arbitrarily. By adding Eq. (7) to the problem, the curve parameter is forced to be the arc-length s. From the mathematical point of view this might be unimportant, but for practical application is very important to preserve the metric properties of the intersection line.

By discretizing Eq. (8) with step-size $h = s_{n+1} - s_n$, the following difference equations are obtained

$$F_x\mathrm{d}x_{n+1} + F_y\mathrm{d}y_{n+1} = 0 \tag{9}$$

$$\mathrm{d}x_{n+1}^2 + \mathrm{d}y_{n+1}^2 = h^2 \tag{10}$$

where $\mathrm{d}x_{n+1} := x_{n+1} - x_n$ and $\mathrm{d}y_{n+1} := y_{n+1} - y_n$. The geometric interpretation of Eqs. (9) and (10) is the intersection of a straight line with a circle of radius h. The situation is visualized in Fig. 1.

The gradient $\nabla F = (F_x, F_y)$ equals the negative main-normal vector n of the intersection line $F(x, y) = C$. The tangent vector t is orthogonal to the main-normal vector

$$t^\top \nabla F = 0, \tag{11}$$

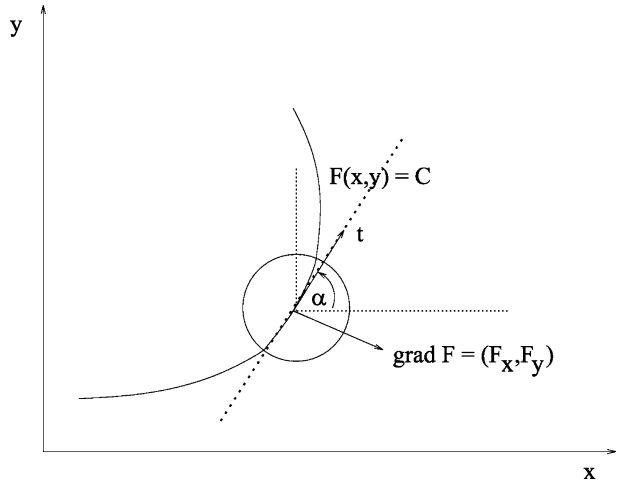

Fig. 1 Geometrical interpretation of the second-order integrator

hence

$$t = (\pm F_y, \mp F_x). \tag{12}$$

is true. The vector t is the direction vector of the straight line described by Eq. (9). Therefore, the two intersection points between a straight line and a circle are

$$\mathrm{d}x_{n+1} = \pm h\cos(\alpha),\ \ \mathrm{d}y_{n+1} = \pm h\sin(\alpha) \tag{13}$$

with

$$\alpha = \arctan\left(\frac{t_y}{t_x}\right) = \arctan\left(\frac{-F_x}{F_y}\right). \tag{14}$$

Depending on the choice of the sign, the level line is followed in the one or in the opposite direction. This leads to the iteration

$$\alpha_n = \arctan\frac{-F_x(x_n, y_n)}{F_y(x_n, y_n)} \tag{15}$$

$$x_{n+1} = x_n \pm h\cos(\alpha_n) \tag{16}$$

$$y_{n+1} = y_n \pm h\sin(\alpha_n) \tag{17}$$

3.2 Consistency and third-order integrator

The concept of consistency describes the degree of consistence of the original with the discretized differential equation. If the exact solution of the differential equation is inserted in the corresponding difference equation, a certain residual $\varepsilon(h)$ will remain. Its magnitude will depend on the chosen step-size h. If the residual is developed into a power series with respect to h

$$\varepsilon(h) = \sum_{n=p}^{\infty} c_n h^n \tag{18}$$

the first non-vanishing power p of h is called the order of consistency of the integrator (Roos and Schwetlick 1999).

Lemma 1. *The integrator (Eqs. 9 and 10) has the order of consistency $p = 2$.*

Proof.

$$\begin{aligned}
&F_x(x_n, y_n)(x_{n+1} - x_n) + F_y(x_n, y_n)(y_{n+1} - y_n)\\
&\quad = F_x(x_n, y_n)\left(x_n' h + \frac{1}{2}x_n'' h^2\right)\\
&\qquad + F_y(x_n, y_n)\left(y_n' h + \frac{1}{2}y_n'' h^2\right) + \mathcal{O}(h^3)\\
&\quad = F_x(x_n, y_n)\left(\frac{1}{2}x_n'' h^2\right) + F_y(x_n, y_n)\left(\frac{1}{2}y_n'' h^2\right) + \mathcal{O}(h^3)
\end{aligned}$$

□

This order of consistency is very low, and forces the use of very small step-sizes h in order to achieve a given accuracy level. Therefore, it is desirable to have an integrator of a higher order of consistency. One possibility to achieve this goal is to find the next solution point as the intersection of the circle $\mathrm{d}x_n^2 + \mathrm{d}y_n^2 = h^2$ with a straight line, which has the arithmetic mean of the tangent vectors in the points n and $n+1$ as direction vector. Figure 2 describes the underlying idea.

The resulting integrator is

$$\begin{aligned}
&\frac{1}{2}(F_x(x_n, y_n) + F_x(x_{n+1}, y_{n+1}))\mathrm{d}x_{n+1}\\
&\qquad + \frac{1}{2}(F_y(x_n, y_n) + F_y(x_{n+1}, y_{n+1}))\mathrm{d}y_{n+1} = 0
\end{aligned} \tag{19}$$

$$\mathrm{d}x_{n+1}^2 + \mathrm{d}y_{n+1}^2 = h^2 \tag{20}$$

Lemma 2. *The integrator (Eqs. 19 and 20) has the order of consistency $p = 3$.*

Proof.

$$\begin{aligned}
&F_x(x_n, y_n)(x_{n+1} - x_n) + F_y(x_n, y_n)(y_{n+1} - y_n)\\
&\quad + F_x(x_{n+1}, y_n)(x_{n+1} - x_n) + F_y(x_{n+1}, y_{n+1})(y_{n+1} - y_n)\\
&= F_x(x_n, y_n)\left[x_n' h + \frac{1}{2}x_n'' h^2\right] + F_y(x_n, y_n)\left[y_n' h + \frac{1}{2}y_n'' h^2\right]
\end{aligned}$$

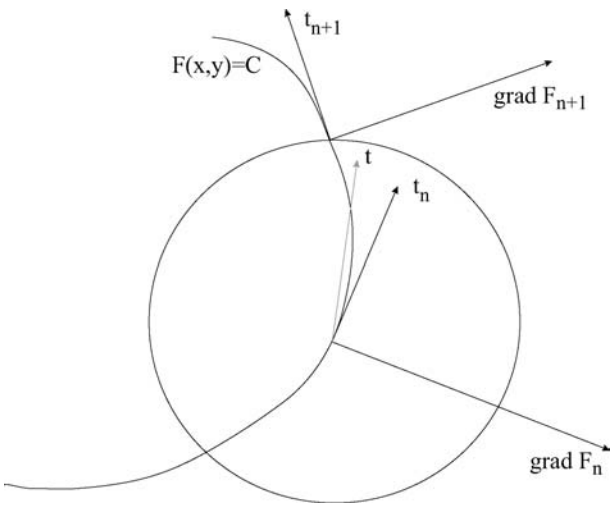

Fig. 2 Geometrical interpretation of the third-order integrator

$$
\begin{aligned}
&- F_x(x_{n+1}, y_{n+1}) \left[-x'_{n+1}h + \frac{1}{2}x''_{n+1}h^2 \right] \\
&- F_y(x_{n+1}, y_{n+1}) \left[-y'_{n+1}h + \frac{1}{2}y''_{n+1}h^2 \right] + \mathcal{O}(h^3) \\
&= \frac{1}{2}F_x(x_n, y_n)x''_n h^2 + \frac{1}{2}F_y(x_n, y_n)y''_n h^2 \\
&\quad - \frac{1}{2}F_x(x_{n+1}, y_{n+1})x''_{n+1}h^2 \\
&\quad - \frac{1}{2}F_y(x_{n+1}, y_{n+1})y''_{n+1}h^2 + \mathcal{O}(h^3) \\
&= \frac{1}{2}F_x(x_n, y_n)x''_n h^2 + \frac{1}{2}F_y(x_n, y_n)y''_n h^2 \\
&\quad - \frac{1}{2}[F_x(x_n, y_n) + F_{xx}(x_n, y_n)(x_{n+1} - x_n) \\
&\qquad + F_{xy}(x_n, y_n)(y_{n+1} - y_n) \\
&\qquad + \mathcal{O}(h^2)][x''_n + x'''_n h + \mathcal{O}(h^2)]h^2 \\
&\quad - \frac{1}{2}[F_y(x_n, y_n) + F_{yx}(x_n, y_n)(x_{n+1} - x_n) \\
&\qquad + F_{yy}(x_n, y_n)(y_{n+1} - y_n) + \mathcal{O}(h^2)] \\
&\quad \times [y''_n + y'''_n h + \mathcal{O}(h^2)]h^2 + \mathcal{O}(h^3) \\
&= -\frac{1}{2}[F_{xx}(x_n, y_n)x'_n h + F_{xy}(x_n, y_n)y'_n h] \cdot x''_n h^2 \\
&\quad - \frac{1}{2}[F_{yx}(x_n, y_n)x'_n h + F_{yy}(x_n, y_n)y'_n h] \cdot y''_n h^2 + \mathcal{O}(h^3) \\
&= \mathcal{O}(h^3)
\end{aligned}
$$

□

Equations (19) and (20) are implicit because the unknown point (x_{n+1}, y_{n+1}) occurs as an argument of F_x and F_y. Therefore, Eqs. (19) and (20) have to be solved iteratively as a so-called *predictor-corrector* scheme.

Predictor Step:

$$\alpha_n^{(0)} = \arctan \frac{-F_x(x_n, y_n)}{F_y(x_n, y_n)} \tag{21}$$

$$x_{n+1}^{(0)} = x_n \pm h\cos(\alpha_n^{(0)}) \tag{22}$$

$$y_{n+1}^{(0)} = y_n \pm h\sin(\alpha_n^{(0)}) \tag{23}$$

Corrector Step:

$$\alpha_n^{(k+1)} = \arctan \frac{-\left(F_x(x_n, y_n) + F_x\left(x_{n+1}^{(k)}, y_{n+1}^{(k)}\right)\right)}{F_y(x_n, y_n) + F_y\left(x_{n+1}^{(k)}, y_{n+1}^{(k)}\right)} \tag{24}$$

$$x_{n+1}^{(k+1)} = x_n \pm h\cos(\alpha_n^{(k+1)}) \tag{25}$$

$$y_{n+1}^{(k+1)} = y_n \pm h\sin(\alpha_n^{(k+1)}), \tag{26}$$

$$k = 0, \ldots, K-1$$

$$x_{n+1} = x_{n+1}^{(K)}, \;\; y_{n+1} = y_{n+1}^{(K)} \tag{27}$$

3.3 Higher-order integrator

Naturally, it is desirable to have an integrator with an order of consistency that is comparable to that of the classical Runge-Kutta integrator. Adapting the ideas of the Runge-Kutta method, a fourth-order integrator for our line-tracking problem is

$$F_x(x_n, y_n)\mathrm{d}\xi_1 + F_y(x_n, y_n)\mathrm{d}\eta_1 = 0 \tag{28}$$

$$(\mathrm{d}\xi_1)^2 + (\mathrm{d}\eta_1)^2 = h^2 \tag{29}$$

$$
\begin{aligned}
&F_x\left(x_n + \frac{1}{2}\mathrm{d}\xi_1, y_n + \frac{1}{2}\mathrm{d}\eta_1\right)\mathrm{d}\xi_2 \\
&\quad + F_y\left(x_n + \frac{1}{2}\mathrm{d}\xi_1, y_n + \frac{1}{2}\mathrm{d}\eta_1\right)\mathrm{d}\eta_2 = 0
\end{aligned} \tag{30}
$$

$$(\mathrm{d}\xi_2)^2 + (\mathrm{d}\eta_2)^2 = h^2 \tag{31}$$

$$
\begin{aligned}
&F_x\left(x_n + \frac{1}{2}\mathrm{d}\xi_2, y_n + \frac{1}{2}\mathrm{d}\eta_2\right)\mathrm{d}\xi_3 \\
&\quad + F_y\left(x_n + \frac{1}{2}\mathrm{d}\xi_2, y_n + \frac{1}{2}\mathrm{d}\eta_2\right)\mathrm{d}\eta_3 = 0
\end{aligned} \tag{32}
$$

$$(\mathrm{d}\xi_3)^2 + (\mathrm{d}\eta_3)^2 = h^2 \tag{33}$$

$$
\begin{aligned}
&\frac{1}{6}\left(F_x(x_n, y_n) + 2F_x\left(x_n + \frac{1}{2}\mathrm{d}\xi_1, y_n + \frac{1}{2}\mathrm{d}\eta_1\right)\right. \\
&\quad + 2F_x\left(x_n + \frac{1}{2}\mathrm{d}\xi_2, y_n + \frac{1}{2}\mathrm{d}\eta_2\right) \\
&\quad \left. + F_x(x_n + \mathrm{d}\xi_3, y_n + \mathrm{d}\eta_3)\right)\mathrm{d}x_{n+1} \\
&\quad + \frac{1}{6}\left(F(x_n, y_n) + 2F_x\left(x_n + \frac{1}{2}\mathrm{d}\xi_1, y_n + \frac{1}{2}\mathrm{d}\eta_1\right)\right. \\
&\qquad + 2F_x\left(x_n + \frac{1}{2}\mathrm{d}\xi_2, y_n + \frac{1}{2}\mathrm{d}\eta_2\right) \\
&\qquad \left. + F_x\left(x_n + \mathrm{d}\xi_3, y_n + \mathrm{d}\eta_3\right)\right)\mathrm{d}y_{n+1} = 0
\end{aligned} \tag{34}
$$

$$(\mathrm{d}x_{n+1})^2 + (\mathrm{d}y_{n+1})^2 = h^2 \tag{35}$$

3.4 Starting point

Let $z = f_i(\mathbf{x}),\ i = 1, 2$ be the equation of two surfaces with $\mathbf{x} = [x\ y]$. The problem of finding a point $\mathbf{x}^*$ on the intersection line $0 = F(\mathbf{x}) := f_1(\mathbf{x}) - f_2(\mathbf{x})$ of the two surfaces is discussed. It is assumed that two initial points $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}$ are known fulfilling

$$F(\mathbf{x}^{(1)}) \cdot F(\mathbf{x}^{(2)}) < 0$$

Define

$$\mathbf{x}^{(3)} := \frac{\mathbf{x}^{(1)} + \mathbf{x}^{(2)}}{2} \tag{36}$$

then a sequence $\left\{\mathbf{x}^{(k)}\right\}$ is constructed in the following way:

$$s_k := \frac{\left\|\mathbf{x}^{(k)} - \mathbf{x}^{(1)}\right\|}{\left\|\mathbf{x}^{(2)} - \mathbf{x}^{(1)}\right\|} \tag{37}$$

$$p_k(s) = \sum_{i=k-2}^{k} \prod_{\substack{j \neq i \\ j = k-2}}^{k} \frac{s - s_j}{s_i - s_j} F(\mathbf{x}^{(i)}) \tag{38}$$

Let be $\lambda_1^{(k)}, \lambda_2^{(k)}$ be the roots of p_k. Then compute

$$\xi_l^{(k)} = \mathbf{x}(1) + \lambda_l^{(k)} \left(\mathbf{x}^{(2)} - \mathbf{x}^{(1)}\right), \quad l = 1, 2 \tag{39}$$

Set

$$\mathbf{x}^{(k+1)} := \left\{\xi_1^{(k)}, \xi_2^{(k)}\right\} \bigcap \left[\mathbf{x}^{(k)}, \mathbf{x}^{(k-2)}\right] \tag{40}$$

$$\mathbf{x}^{(k)} = \begin{cases} \mathbf{x}^{(k-1)}, & F(\mathbf{x}^{(k-1)}) \cdot F(\mathbf{x}^{(k+1)}) < 0 \\ \mathbf{x}^{(k-2)}, & \text{otherwise} \end{cases} \tag{41}$$

For the unknown initial point $\mathbf{x}^*$, it always holds that

$$\mathbf{x}^* \in \left[\min\left\{\mathbf{x}^{(k+1)}, \mathbf{x}^{(k)}\right\}, \max\left\{\mathbf{x}^{(k+1)}, \mathbf{x}^{(k)}\right\}\right] \tag{42}$$

4 Snakes-based algorithm

The model of deformable curves, the so-called snakes, was introduced by Kass et al. (1987) and used for the extraction of fuzzy line-objects in digital images. Due to the universality of the underlying principle, the method has found many applications in almost all domains of geo-data processing.

A certain kind of external energy is assigned to the curve to be modelled. This energy defines the resulting shape of the curve. On the other hand, the medium containing the curve has some internal energy (potential), reflecting the internal structure of the medium. The interaction between internal and external energy places the deformable curve at an optimal position, which is characterized by a balance between the two kinds of energy.

If the curve is normalized to the interval [0, 1], parameterised by s, its parameter representation reads $\mathbf{v} := [x = x(s)\ \ y = y(s)]^\top$. The optimal snake's position can be found by minimizing the total energy of the curve

$$I[\mathbf{v}(s)] = \int_0^1 (E_{\text{int}} + E_{\text{ext}}) \mathrm{d}s. \tag{43}$$

The internal energy E_{int} describes the geometric properties of the curve

$$E_{\text{int}} := \frac{[\alpha \mathbf{v}_s^2 + \beta \mathbf{v}_{ss}^2]}{2}, \tag{44}$$

where $\mathbf{v}_s := \left[\frac{\mathrm{d}x}{\mathrm{d}s} \ \ \frac{\mathrm{d}y}{\mathrm{d}s}\right]^\top$ is the elastic term and $\mathbf{v}_{ss} := \left[\frac{\mathrm{d}^2x}{\mathrm{d}s^2} \ \ \frac{\mathrm{d}^2y}{\mathrm{d}s^2}\right]^\top$ is the viscosity term. The elastic term of the internal energy is weighted by the parameter α, and the viscosity term is weighted by the parameter β. Varying the parameters α and β leads to curves with the desired geometrical properties. The second term in Eq. (43), the external energy E_{ext}, *activates* the snake and causes a shift of the snake as long as the energy terms are not in balance. The external energy is generated by the data and is therefore context-sensitive.

A necessary condition for the solution of the extremal problem (Eq. 43) with the boundary conditions $\mathbf{v}(\mathbf{0}) = \mathbf{v}_a$ and $\mathbf{v}(\mathbf{1}) = \mathbf{v}_b$ is the stationarity of the first variation of $I[\mathbf{v}(s)]$:

$$\delta I[\mathbf{v} + \delta \mathbf{v}] = 0 \tag{45}$$

This leads to a differential equation of fourth order, the so called Euler equation, which after insertion of internal and external energy has the following form:

$$\text{grad}(E_{\text{ext}}) - \alpha \mathbf{v}_{ss} + \beta \mathbf{v}_{ssss} = 0. \tag{46}$$

After discretization of the Euler equation (Eq. 46) by finite differences one obtains the following linear system of equations

$$\mathbf{A}\mathbf{x} + \mathbf{E}_{\mathbf{v}} = 0 \tag{47}$$

with the pentadiagonal coefficient matrix

$$\mathbf{A} = \begin{bmatrix} a & b & c & 0 & 0 & 0 & \dots \\ b & a & b & c & 0 & 0 & \dots \\ c & b & a & b & c & 0 & \dots \\ 0 & c & b & a & b & c & \dots \\ 0 & 0 & c & b & a & b & \dots \\ 0 & 0 & 0 & c & b & a & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{48}$$

where $\mathbf{E}_{\mathbf{v}} = \text{grad}(E_{\text{ext}})$ and the coefficients are given by

$$a = 2\alpha + 6\beta, \quad b = -\alpha - 4\beta, \quad c = \beta. \tag{49}$$

Since $\det(\mathbf{A}) = 0$, the linear system of equations cannot be solved directly. Instead, it is transformed into the semi-linear system

$$\mathbf{v}_t = (\mathbf{A} + \gamma \mathbf{I})^{-1} \left(\gamma \mathbf{v}_{t-1} - \mathbf{E}_{\mathbf{v}}|_{t-1}\right) \tag{50}$$

and solved iteratively. The parameter γ is an additional elimination parameter, which in most cases is set equal to unity.

Now the snakes algorithm will be adapted to the problem of the identification of the intersection line of two surfaces. The idea behind the algorithm is shown in Fig. 3 (Borkowski 2004): the process starts with an almost arbitrary initial snake in the xy plane. This initial snake is moved and deformed as long as an optimal approximation of the projection of the intersection is not reached.

For this purpose, the external energy is defined as the sum of the squares of the distances between two surfaces at the instantaneous position of the snake.

$$E_{\text{ext}} \propto \frac{\mu}{2}(z_2 - z_1)^2, \tag{51}$$

where $z_1 = f_1(x, y)$ and $z_2 = f_2(x, y)$ are the snake-heights on the left and the right surface patch, respectively. In order to guarantee convergence, the external energy is scaled with the weight μ. The derivatives needed in the snakes approach are:

$$\mathbf{E}_{\mathbf{v}} := \begin{bmatrix} \frac{\partial E_{\text{ext}}}{\partial x} \\ \frac{\partial E_{\text{ext}}}{\partial y} \end{bmatrix} = \mu(z_2 - z_1) \begin{bmatrix} \frac{\partial f_2(x,y)}{\partial x} - \frac{\partial f_1(x,y)}{\partial x} \\ \frac{\partial f_2(x,y)}{\partial y} - \frac{\partial f_1(x,y)}{\partial y} \end{bmatrix} \tag{52}$$

In contrast to the line-tracking algorithm, the snakes approach does not need an initial point. The initial snake can be placed arbitrarily.

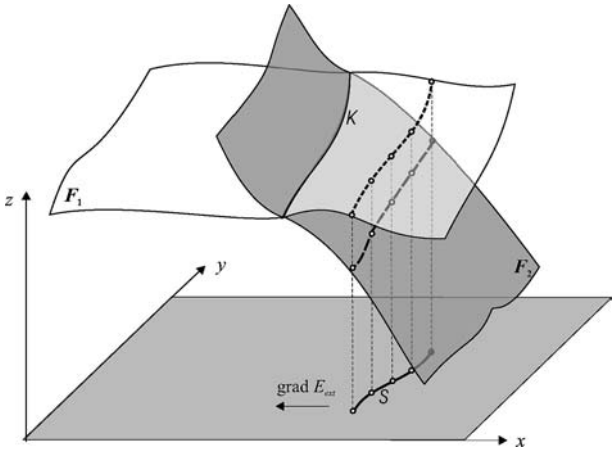

Fig. 3 Identification of the intersection curve K with snake S

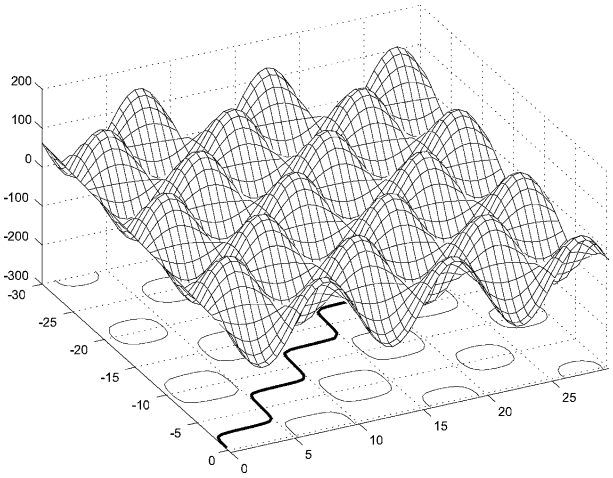

Fig. 4 Synthetic test data

5 Examples

5.1 Synthetic examples

Firstly, numerical integrators of different orders for the solution of the line-tracking problem are compared to each other. The intersection line of the plane $z = 0$ with the surface $z = 50.0 \cdot \cos\left(\frac{\pi}{5}x\right) \cdot \sin\left(\frac{\pi}{5}x\right)$ (Fig. 4) is computed twice: first with an integrator of second order, and then with an integrator of third order.

The surface has several closed and one non-closed zero-level lines. The initial point for the line-tracking algorithm is placed on the non-closed line and therefore this line is computed by the numerical integration.

The second-order integrator is given by Eqs. (9) and (10). The third-order integrator is the Predictor-Corrector approach, mentioned before, which is implemented in two different ways:

1. One corrector step for each predictor step.
2. Five corrector steps for each predictor step.

The intersection lines computed with both integrators are displayed in Fig. 5. Visually, there is no difference between the two solutions. If the neighbourhood of the curvature maximum is enlarged, the differences become visible (see Fig. 6). From Figs. 5 and 6, the third-order integrator follows the curvature more closely while the second-order integrator prefers the secant direction.

Another comparison is based on the computation of the z-values along the computed projection xy of the intersection line. If the computed projection were error-free, the z-values would be identical to zero. The deviation of the computed z-values from zero can be used as a measure for the accuracy of the method. Figure 7 shows the z-values along the two projections, computed with the second- and third-order integrators.

It is clearly visible in Fig. 7 that the second-order integrator generates a considerably larger error than the third-order integrator. Besides the dependence of the integration error

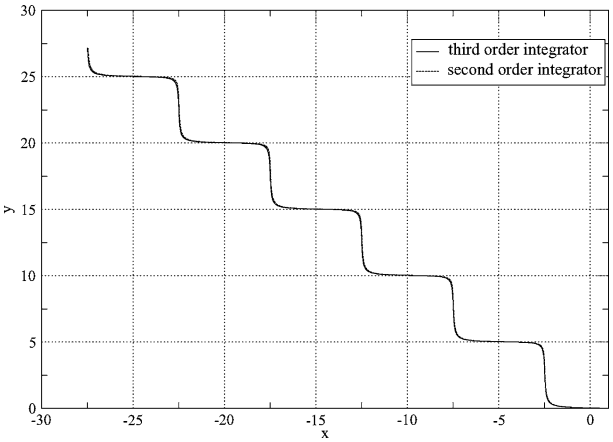

Fig. 5 Location of the computed intersection lines

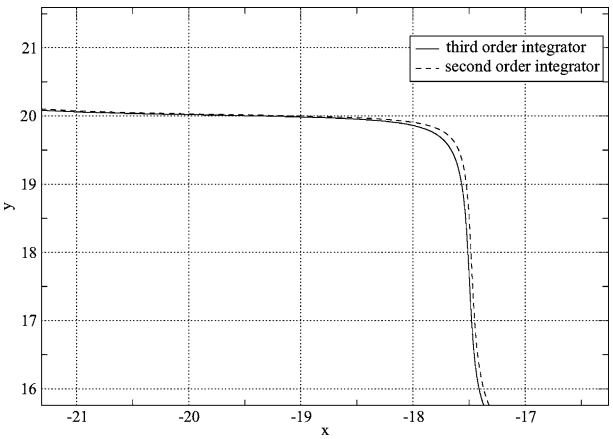

Fig. 6 Location of the computed intersection lines (*zoomed in*)

from the curvature of the projection, an additional undesired error trend is visible.

Even the third-order integrators differ in their performance, depending on the number of corrector steps carried out. Figure 8 shows the z values along the xy projections,

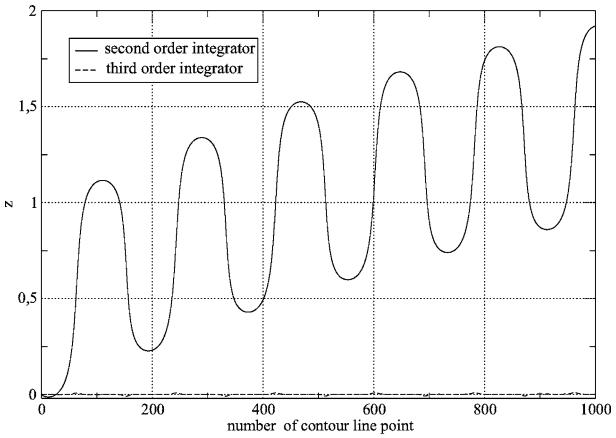

Fig. 7 Accuracy of the computed intersection lines

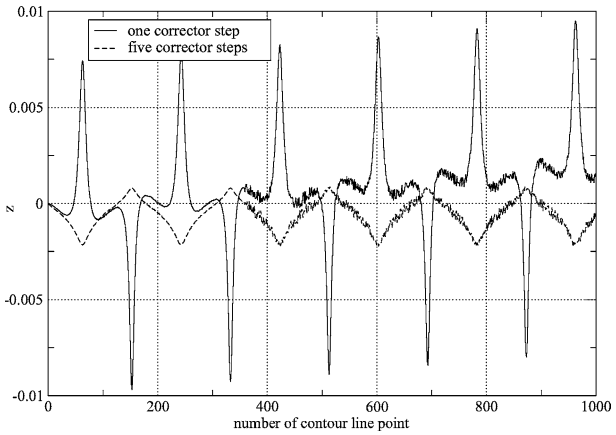

Fig. 8 Accuracy of the computed intersection line

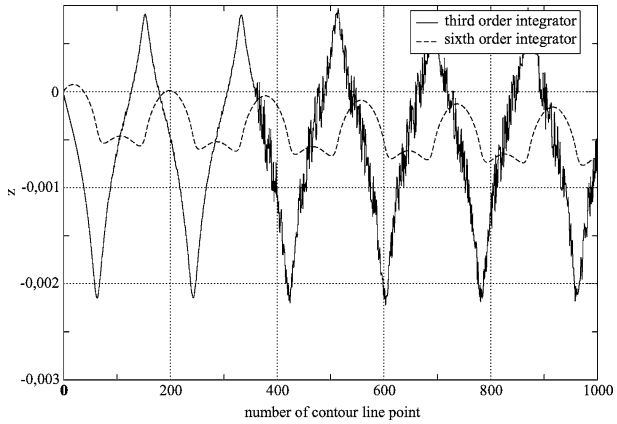

Fig. 9 Accuracy of integrators with different orders of accuracy

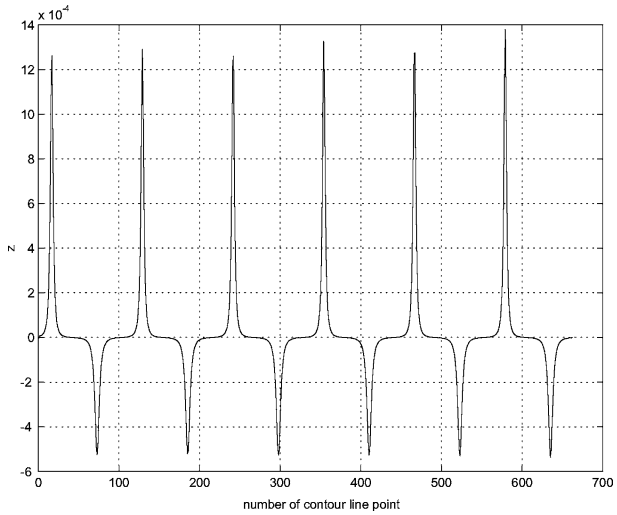

Fig. 10 Height accuracy of the snakes method

which were computed once with one and five corrector steps. As expected, the iterated Predictor-Corrector method is much more stable than the simple third-order integrator.

If the same contour line as in the previous example is computed with the fourth-order integrator, again the level error can be considered as an indicator of accuracy. In Fig. 9, the level errors of the third-order predictor–corrector method and the sixth-order method are displayed.

The same example has also been tackled using the snakes approach. Figure 10 displays the z-level accuracy of the identified intersection line. Obviously, the snakes approach is able to provide the same accuracy as the third-order integrator, but uses snake-elements (snaxels), which are shorter than the step-size h of the integrator. Both methods have their specific advantages and disadvantages, as follows.

- Numerical integration is a local method. Therefore, the integration error increases with increasing length of the intersection line. In contrast, snakes have global approximation properties. The accuracy is almost constant along the intersection line.
- Snakes do not need an initial point. The initial snake can be placed almost arbitrarily. On the other hand, numerical integration is much faster, since it does not require the solution of a system of linear equations, as this is the case for the snakes. For intersection lines with large curvature changes, several iterations are necessary, which can slow down the extraction process considerably.
- By a suitable choice of the parameters α and β, the snake approach can produce intersection lines with desired geometrical properties. This allows for a simultaneous feature extraction and noise reduction. In the numerical integration the measurement errors propagate freely into the shape of the extracted intersection line.

5.2 Real-world examples

The above numerical examples with the synthetic data have shown that the snakes-algorithm and the differential-equation-contouring lead to results that are comparable in terms of accuracy and computational efficiency. In order to demonstrate the practical value of both methods, they now will be applied to edge detection for real data sets. For this purpose, two data sets are used: Data set 1 (see Fig. 11) represents a section of the embankment of the Oder river. Data set 2 (see

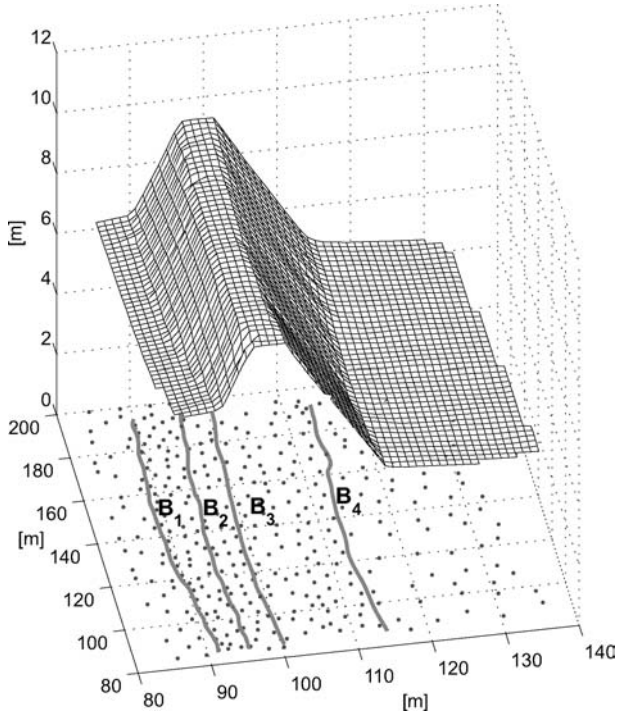

Fig. 11 Test data 1

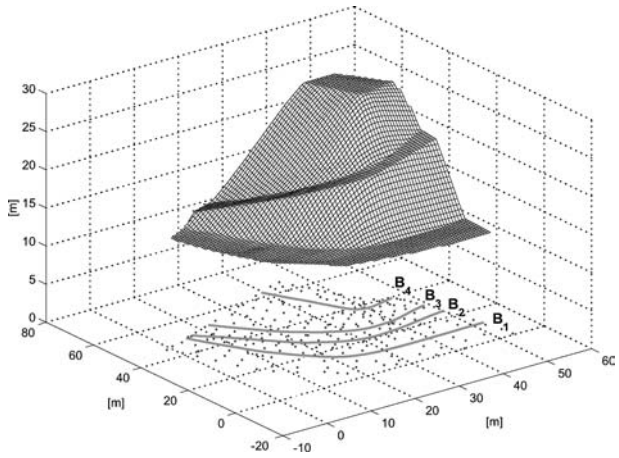

Fig. 12 Test data 2

Fig. 12) comes from the Polish city Wroclaw, where an artificially arisen terrain-form with unequivocally identifiable edges was scanned.

The first data set was measured primarily for this paper, the second was taken from Borkowski (2004). Since real laser scanner data do not provide a ground-truth for the location of the terrain edges, in the test data sets, the laser-scanner was emulated by a total-station. The spacing of the measured points was irregular in order to generate a similar (irregular) data distribution as laser-scanning would have done (a point cloud). For the establishment of ground truth, the edges were measured separately.

Since the data are given as a point cloud, a functional representation of the two surface patches has to be derived first. A thin-plate spline representation (Duchon 1976) is used for this purpose:

$$z(x, y) = \frac{1}{2}\sum_{i=1}^{n} \lambda_i r_i^2 \ln r_i^2 + \nu_{00} + \nu_{10} x + \nu_{01} y, \tag{53}$$

where $r_i^2 = (x - x_i)^2 + (y - y_i)^2$ and λ, μ are control parameters. The $n + 3$ unknowns are related to the n data w by the following linear system of equations

$$\begin{bmatrix} \mathbf{A} & \mathbf{T} \\ \mathbf{T}^\top & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\nu} \end{bmatrix} = \begin{bmatrix} \mathbf{w} \\ \mathbf{0} \end{bmatrix}, \tag{54}$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & 0 & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & 0 & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & 0 \end{bmatrix}, \qquad \mathbf{T} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \\ \vdots & \vdots & \vdots \\ 1 & x_n & y_n \end{bmatrix},$$

$$\boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & \dots & \lambda_n \end{bmatrix}^\top, \qquad \boldsymbol{\nu} = \begin{bmatrix} \nu_{00} & \nu_{10} & \nu_{01} \end{bmatrix}^\top,$$

$$\mathbf{w} = \begin{bmatrix} w_1 & w_2 & w_3 & \dots & w_n \end{bmatrix}^\top$$

and $a_{ij} = a_{ji} = r_{ij}^2 \ln r_{ij}$ with $r_{ij}^2 = (x_i - x_j)^2 + (y_i - y_j)^2$, $i, j = 1, 2, ..., n$.

For the identification of the terrain edges as the intersections of two adjacent thin-plate spline surface patches, both algorithms were used: The snakes algorithm and the contouring algorithm based on differential equation solver of second-order. In the horizontal projections the location of the data points are marked in Figs. 11 and 12. Also, the projection of the ground-truth edges B_i, $i = 1, 2, 3, 4$ into the xy-plane are drawn in Figs. 15 and 16.

For the edge detection with snakes, the parameters α, β, μ were chosen as $\alpha = \beta = \mu = 1$. The length of the snakes-elements was set to $h = 0.2$ m, which corresponds to the step-size of the differential equation contouring algorithm. The deviations between the identified edges and the ground-truth are shown in Figs. 13–16.

In Figs. 13–16, the left column always shows the deviation (d) in the horizontal plane and the right column the deviation (Δz) in height. The compression of this information into statistical parameters can be found in Tables 1–4. For each edge, the maximal horizontal distance $d_{\max}$, the mean horizontal distance $\overline{d}$ and the standard deviation σ_d are given. The same applies to the height deviation.

The statistical parameters in Tables 1–4 show that both methods are comparable in terms of accuracy. Both methods show a deviation from the ground-truth, which corresponds to the data accuracy. The accuracy is influenced by the data density and the intersection angle of the two surface patches. The differences between the results of the two methods are much smaller and are mainly determined by the errors of numerical approximation. Nevertheless, both methods fulfill the requirements for practical edge detection.

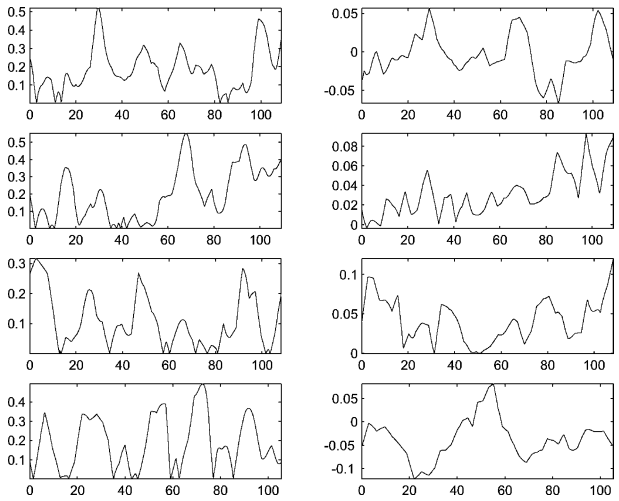

Fig. 13 Test data 1. Horizontal (d) and vertical (Δz) deviations versus arc length for the snakes edge detection (all units in meter)

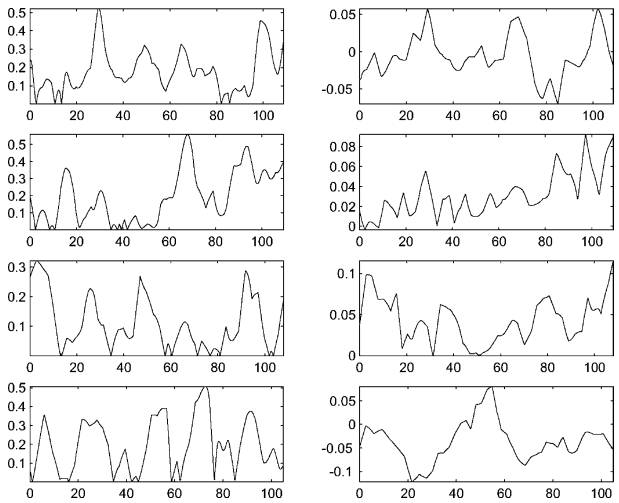

Fig. 14 Test data 1. Horizontal (d) and vertical (Δz) deviations versus arc length for the differential equation contouring edge detection (all units in meter)

Table 1 Test data 1. Accuracy measures of edge detection by snakes (all units in meter)

Edge	$d_{\max}$	$\overline{d}$	σ_d	$\Delta z_{\max}$	$\overline{\Delta z}$	$\sigma_{\Delta z}$
B_1	0,51	0,18	0,11	0,06	0,00	0,03
B_2	0,55	0,20	0,15	0,09	0,03	0,02
B_3	0,31	0,11	0,09	0,12	0,04	0,03
B_4	0,49	0,19	0,13	−0,12	−0,04	0,04

Table 2 Test data 1. Accuracy parameter of edge detection by differential equation contouring (all units in meter)

Edge	$d_{\max}$	$\overline{d}$	σ_d	$\Delta z_{\max}$	$\overline{\Delta z}$	$\sigma_{\Delta z}$
B_1	0,52	0,18	0,11	−0,07	0,00	0,03
B_2	0,56	0,19	0,15	0,09	0,03	0,02
B_3	0,32	0,11	0,09	0,12	0,04	0,03
B_4	0,51	0,20	0,13	−0,12	−0,04	0,04

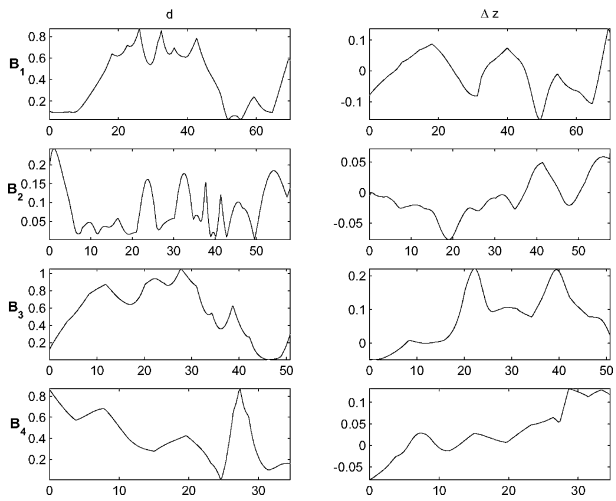

Fig. 15 Test data 2. Horizontal (d) and vertical (Δz) deviations versus arc length for the snakes edge detection (all units in meter)

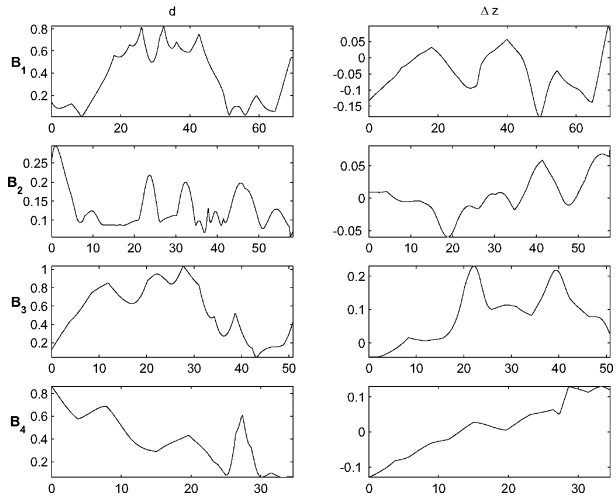

Fig. 16 Test data 2. Horizontal (d) and vertical (Δz) deviations versus arc length for the differential equation contouring edge detection (all units in meter)

Table 3 Test data 2. Accuracy measures of edge detection by snakes (all units in meter)

Edge	$d_{\max}$	$\overline{d}$	σ_d	$\Delta z_{\max}$	$\overline{\Delta z}$	$\sigma_{\Delta z}$
B_1	0,87	0,40	0,26	−0,16	−0,01	0,06
B_2	0,24	0,08	0,06	−0,08	−0,01	0,03
B_3	1,05	0,56	0,30	0,22	0,07	0,08
B_4	0,87	0,42	0,22	0,13	0,03	0,05

Table 4 Test data 2. Accuracy parameter of edge detection by differential equation contouring (all units in meter)

Edge	$d_{\max}$	$\overline{d}$	σ_d	$\Delta z_{\max}$	$\overline{\Delta z}$	$\sigma_{\Delta z}$
B_1	0,83	0,36	0,25	−0,18	−0,04	0,06
B_2	0,29	0,13	0,05	0,08	0,01	0,03
B_3	1,04	0,55	0,28	0,23	0,08	0,07
B_4	0,86	0,39	0,22	0,13	0,01	0,07

6 Conclusion

The paper proposes two methods for the extraction of a three-dimensional vector model of a terrain edge from laser scanning data. The point of departure for both methods is the same: the data have to be classified into points left and right from the edge and a surface patch model $z = f_1(x, z)$ and $z = f_2(x, y)$ has to be fitted to the data of each class. Any kind of surface patch fitting can be used. The only requirement for the surface patch models is that their extrapolation behaviour does not tend to have strong oscillations. The intersection line of the two surface patches is considered the terrain edge to be modelled. This means the intersection line is the solution of the equation

$$F(z) := f_1(x, y) - f_2(x, y) = 0. \tag{55}$$

The two methods differ by the way in which Eq. (55) is solved.

- One possibility is to derive a differential equation, expressing the fact that the value $F(z)$ along intersection line does not change and to solve the differential equation by a numerical integrator.
- The other possibility is to define the external energy as the sum of squares of the differences $E_{\text{ext}} = \sum(f_1(x, y) - f_2(x, y))^2$ in z-values of both surface patches and to find a snake that minimizes the sum of this external and its own internal energy.

Both methods have their specific advantages and disadvantages.

- The integration method is fast and inexpensive, but it needs an initial point on the edge to be extracted. The method is independent of curvatures and curvature changes, but it cannot react to any side condition about the geometrical nature of the edge it computes.
- The snakes method does not require an initial point on the edge to be extracted. Already a very coarse initial location of the snake is sufficient. The snake approach is an iterative method where the number of necessary iterations strongly depends on the curvature of the edge to be found. Since in each step a linear system of equations has to be solved, the iteration process can become rather costly. By a suitable choice of the weighting parameters α, β the geometrical nature of the resulting terrain edge can be controlled.

Both methods provide a numerical accuracy that meets practical requirements. The numerical tests show that from discrete laser-scanning data, the form-lines of the terrain can be determined with a high accuracy. In particular, the accuracy of the height information has to be emphasized (see Tables 1–4).

In practice, the horizontal projection of the terrain edge is often known, e.g. from photogrammetry. Photogrammetry always provides a high accuracy of the horizontal projection, but only a low accuracy of the height information. This prior information gained from photogrammetry can be used to classify the laser scanning data into two classes: on the one and on the other side of the terrain edge. Based on this classification, the proposed methods can be used to improve the accuracy of the height information (Kraus 2004). Therefore, the combination of photogrammetry, point classification and one of the proposed methods are a useful tool to improve the overall accuracy of edge detection from laser scanning data.

References

Ascher UM, Petzold LR (1998) Computer methods for ordinary differential equations and differential-algebraic equations. SIAM, Philadelphia

Borkowski A (2004) Modellierung von Oberflächen mit Diskontinuitäten. Deutsche Gedätische Kommission, Reihe C, Nr 575

Borkowski A, Keller W (2003) Modelling of irregularly sampled surfaces by two-dimensional snakes. J Geod 77(9):543–553

Briese C, Kraus K, Pfeifer N (2002) Modellierung von dreidimensionalen Geländekanten in Laser-Scanner-Daten. Festschrift zum 65. Geburtstag von Prof. Dr-Ing habil Sigfried Meier. Technische Universität Dresden, pp 47–52

Brügelmann R (2000) Automatic breaklines detection from airborne laser scanner data. Int Arch Photogrammetry Remote Sensing XXXIII(B3):109–116

Duchon J (1976) Interpolation des fonctions de deux varianles suivant le principle de la flexion des plaques minces. R.A.I.R.O. Analyse Numérique 10:5–12

Hairer E, Lubich Ch, Wanner G (2002) Geometric numerical integration: structure preserving algorithms for ordinary differential equations. Springer, Berlin Heidelberg New York

Hoschek J, Lasser D (1992) Grundlagen der geometrischen Datenverarbeitung B.G. Teubner, Stuttgart Leipzig

Kass M, Witkin A, Terzopoulos D (1987) Snakes: active contour models. In: Proceedings of the first international conference on computer vision. IEEE Comput Soc Press, pp 259–268

Kraus K (2004) Personal communication

Kraus K, Pfeifer N (2001) Advanced DTM Generation from LIDAR Data. Int Arch Photogrammetry Remote Sensing XXXIV-3(W4):23–30

Roos H-G, Schwetlick H (1999) Numerische Mathematik. Das Grundwissen für jedermann. B.G. Teubner, Stuttgart Leipzig

Sui L (2002) Ableitung topographischer Strukturlinien aus Laserscannerdaten mit Methoden der Bildverarbeitung. Zeitschrift für Photogrammetrie, Fernerkundung, Geoinformation, 423–434