Computational and Applied Mathematics (2023) 42:332
https://doi.org/10.1007/s40314-023-02457-w

Multichannel segmentation of planar point clouds using evolving curves

Lukáš Tomek[1] · Branislav Beran[1] · Ján Erdélyi[1] · Richard Honti[1] · Karol Mikula[1]

Received: 14 April 2023 / Revised: 11 August 2023 / Accepted: 2 September 2023 /
Published online: 13 October 2023

Abstract

The paper deals with the multichannel curve segmentation of planar point clouds, scanned by a terrestrial laser scanner (TLS), which usually represent walls of buildings. Information derived from the point cloud can be used for quality check of a building (e.g. wall flatness or mutual perpendicularity of walls) or creation of virtual models of buildings. Therefore, segmentation of the planar point cloud into subsets representing the surface of the wall and other structures (e.g. door, electrical outlets) is desired. We describe a mathematical model of evolving planar curves, which is used for the segmentation of the point cloud. The evolution of curves is controlled by the properties of the scanned walls, such as colour and intensity. We discretize the model using the finite-volume method and the semi-implicit Inflow-Implicit/Outflow-Explicit (IIOE) scheme. We demonstrate the functionality of segmentation on real data.

Keywords Numerical methods · Finite-volume methods · Point cloud · Image segmentation · Evolving curves

Mathematics Subject Classification 53Z30 · 68U10 · 65M08 · 65D17

Communicated by Antonio José Silva Neto.

✉ Lukáš Tomek
tomek@math.sk

Branislav Beran
branislav.beran@gmail.com

Ján Erdélyi
jan.erdelyi@stuba.sk

Richard Honti
richard.honti@stuba.sk

Karol Mikula
mikula@math.sk

[1] Faculty of Civil Engineering, Slovak University of Technology in Bratislava, Radlinského 11, Bratislava 811 07, Slovak Republic

1 Introduction

One of the most effective methods for collecting spatial data is terrestrial laser scanning (TLS). The resulting data are provided as a 3D point cloud, possibly including radiometric information, namely colour and intensity. In general, point cloud data can serve as a geometric basis for modelling objects and phenomena. Point clouds are used for visualisation, measurement, and many other surveying tasks such as 3D model creation or surface reconstruction (Previtali et al. 2014), digitalisation of cultural heritage (Adamopoulos and Rinaudo 2021; Haličková and Mikula 2016; Remondino and Rizzi 2010), deformation analysis of structures (Holst et al. 2015), flood modelling or landslide surveying (Giussani and Scaioni 2012).

In the present paper, we work with point clouds resulting from TLS scans of buildings. Nowadays, virtual models are created for managing the whole lifecycle of a building, from the design, construction, operation to facility management and demolition using Building Information Modelling (BIM). A BIM model is an object-oriented virtual model defined as a combination of graphic and non-graphic data. The graphic data in most cases have the form of a 3D model. The use of information from the 3D point cloud and the information derived from the BIM model can be used for the monitoring and quality check of a given structure during the construction process or after its completion (Bariczová et al. 2021).

Our point cloud data include 3D coordinates of the points, colour channels captured by a digital camera, and the intensity channel containing information related to the strength of the reflected laser signal. Our goal is to find planar subsets of the original point cloud containing points that represent the surface of walls. These subsets could be used, e.g. for automation of the creation of as-built models of buildings.

In our typical workflow, we perform a two-phase segmentation. Primary segmentation is performed by plane fitting. Planar subsets of the entire point cloud are selected. Each subset contains points lying within a specified distance from the fitting plane and will be referred to as planar point cloud. The planar point cloud corresponding to a wall often contains objects that do not represent the surface of the wall (e.g. door and electrical outlets) and may affect the result of the quality check (Bariczová et al. 2021). The role of the secondary segmentation based on evolving curves is to segment each planar point cloud into subsets representing the wall surface and other objects, e.g. door, whiteboard, and the remaining unsegmented points (waste).

For the primary segmentation, many methods have been developed, see the review by Grilli et al. (2017). We usually use a modification of the Random Sample Consensus (RANSAC) algorithm (Fischler and Bolles 1981) called M-estimated Sample Consensus (MSAC) (Torr and Zisserman 2000), possibly followed by some filters, e.g. a normal filter which discards points whose local normals deviate from the normal of the fitting plane by more than a given threshold angle. In addition to the MSAC method, we also use the approach developed in Honti et al. (2022).

For the purpose of this paper, we consider a planar point cloud as the input data for the secondary segmentation, which will be described in detail. As we will see, the basic idea of our approach is to create a set of images representing the point cloud data[1] and subsequently segment these images. In the field of image processing, many segmentation methods based on curve and surface evolution models have been developed. We distinguish two main approaches to handle evolution problems, the so-called direct (or Lagrangian) approach (Dziuk 1999; Mikula and Ševčovič 2004; Balažovjech and Mikula 2011) and level

[1] One image for each channel, e.g. colour channels R, G, B or intensity channel I. The images are obtained by orthogonal projection of the point cloud into the meshed fitting plane.

set (or Eulerian) approach (Sethian 1999; Osher et al. 2004; Caselles et al. 1997; Zhao et al. 2000; Corsaro et al. 2006). Our secondary segmentation is based on a Lagrangian curve evolution model, which is computationally fast because it solves only 1D curve evolution problem. The Eulerian approach treats the segmentation curve as the zero-level set of the level set function and the whole function is evolving. Therefore, one has to solve 2D evolution problem which is computationally more expensive.

The evolution of segmentation curves is driven by information from multiple channels of the point cloud data, e.g. colour, intensity, or distance from the fitting plane. The quality of the discretisation mesh is crucial for stability and accuracy in the Lagrangian approach. Various techniques for tangential redistribution of points improving the mesh quality have been developed (Hou et al. 1994; Kimura 1997; Ševčovič and Mikula 2001; Barrett et al. 2011; Ševčovič and Yazaki 2011; Benninghoff and Garcke 2014), and we adopt asymptotically uniform redistribution of points from (Mikula and Ševčovič 2004). Moreover, an algorithm for the detection and treatment of the topological changes is needed (Benninghoff and Garcke 2014; Frei and Garcke 2016). We use an efficient $O(n)$ algorithm developed in (Mikula and Urbán 2012; Balažovjech et al. 2012; Ambroz et al. 2019).

The mathematical model for curve evolution is an extension of the model from the paper (Mikula et al. 2021a) which deals with one channel segmentation of forests from satellite images. Contributions of the present paper are mainly the fully automatic methodology for segmentation of the point clouds (meaning two segmentation phases, creation of the representative images, curve segmentation and creation of the final segments of the point cloud), the multichannel approach using a novel construction of the aggregated edge detector, and simultaneous segmentation of multiple regions (requiring a different detection of merging of curves). Minor contributions include construction of the initialisation mask, a simpler approximation of the signed curvature (without using arccos as in Mikula et al. (2021a)) and a local weight for the curvature regularisation. To make the text self-content, we included some ideas from Mikula et al. (2021a) in this paper. A preliminary version of the curve segmentation of planar point clouds is outlined in the paper (Bariczová et al. 2021) which deals with quality check of walls in buildings. In the paper (Bariczová et al. 2021), only a brief intuitive description of the model is presented, and the functionality of the segmentation is shown only on one example. In this paper, we present the final version of the methodology with the full mathematical description of the model and all computational details of our approach. The functionality of the model is demonstrated on multiple representative examples.

The text of this paper is organised into three main sections. In Sect. 2, we introduce the main ideas of our segmentation methodology, mainly the creation of the representative planar point cloud images (Sect. 2.1), mathematical model (Sect. 2.2), construction of the edge detectors (Sect. 2.3), and our approach to the multichannel segmentation (Sect. 2.7). In Sect. 3, we present the numerical discretisation of the mathematical model. The Sect. 4 shows several numerical experiments with real data that illustrate the functionality of our approach.

2 Point cloud segmentation methodology

In this section, we describe our method for segmentation of the point clouds. We obtain our input point cloud data using the Trimble TX5 scanner. The point clouds usually contain millions of points with the average resolution about $2 - 6$ mm. A typical example is a flat

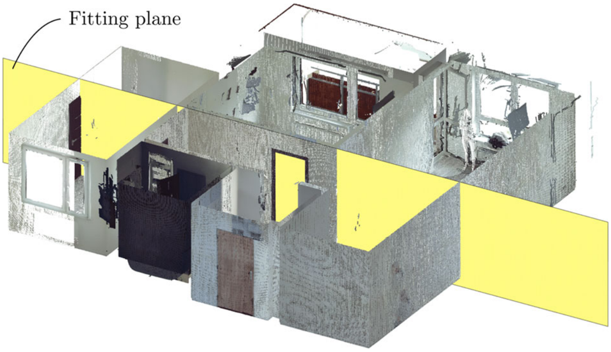
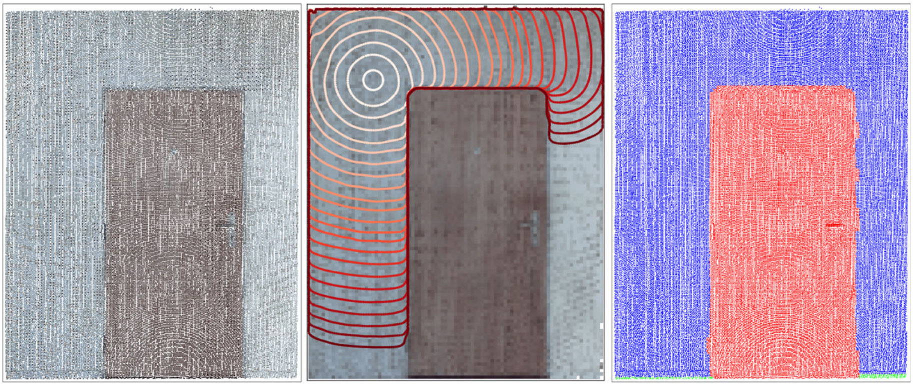


Fig. 1 Primary segmentation of an apartment scanned by TLS. A planar subset corresponding to a wall was segmented



(a) The planar point cloud. (b) Beginning of the curve segmentation. (c) Segments of the point cloud.

Fig. 2 An example of the secondary segmentation phase of a planar point cloud corresponding to a wall with a door

in Fig. 1.[2] The overall workflow of our segmentation process consists of the following main steps.

1. **Primary segmentation by plane fitting.** Selection of planar subsets of the point cloud using the MSAC method, Fig. 1. Each subset consists of points located within a specified threshold distance from the fitting plane. The fitting can be possibly followed by a normal filter removing points whose local normals deviate from the normal of the fitting plane by more than a specified threshold angle. The resulting planar point clouds[3] serve as input data for secondary segmentation.
2. **Secondary segmentation by curve evolution.** Segmentation of each planar subset of the point cloud using evolving curves (Fig. 2).

[2] It might seem like a continuous surface, but it is an unorganised cloud of points without any structure. Figure 1 shows a very dense point cloud with points visualised as tiny square planar patches.

[3] For ease of expression, we call these subsets planar point clouds, although in reality they are only approximately planar.

(a) Preprocessing: Creation of representative images corresponding to the channels (such as colour or intensity) of the planar point cloud.
(b) Evolution: Segmentation by evolving curves using representative images.
(c) Postprocessing: Extraction of points in the point cloud corresponding to the segmented regions.

In the following subsections, we describe the steps of the secondary segmentation in detail. As we mentioned, the same procedure is performed for each planar point cloud. Therefore, in the following, we will work with a single planar point cloud. In Sects. 2.2–2.6, we present the method for one-channel segmentation and in Sect. 2.7, we propose a technique to use information from multiple channels.

2.1 Preprocessing: creation of the representative images from planar point cloud

To reduce the dimension of the problem, we transform (rotate and move) our planar point cloud to the new coordinate system in which the first two dimensions span the fitting plane. If we have a transformed point with coordinates $\mathbf{x} = (x_1, x_2, x_3)$, the coordinates x_1 and x_2 describe the position of the point in the plane and the x_3 axis is orthogonal to the fitting plane, i.e. the absolute value of x_3 is the distance from the fitting plane, denoted as $\mathrm{D} = |x_3|$.

Subsequently, we create a regular square mesh in the x_1, x_2 plane. We denote the pixel size (edge length) as h and usually set it as a multiple of the resolution of the point cloud (e.g. 3×resolution). Then, we represent the point cloud data by a set of bitmap images, one image for each available channel. Usually, we use the colour channels R, G, B, the intensity channel I, and the distance channel D. For example, we take the R channel of colour and compute the value in each pixel (square of the mesh) as the mean value of the R channel of the points which lie in the pixel, see Fig. 3. Finally, we rescale the image values to the range [0, 1]. The domain of the images will be denoted by $\Omega \subset \mathbb{R}^2$.

The resulting images are pixel representations of the input planar point cloud and will be segmented using evolving curves.

2.2 Mathematical model of the curve evolution

In this section, we present a Lagrangian curve evolution model for segmentation of the planar point clouds obtained by TLS. The input data is an image $\mathcal{I}$, which is one of the channels.

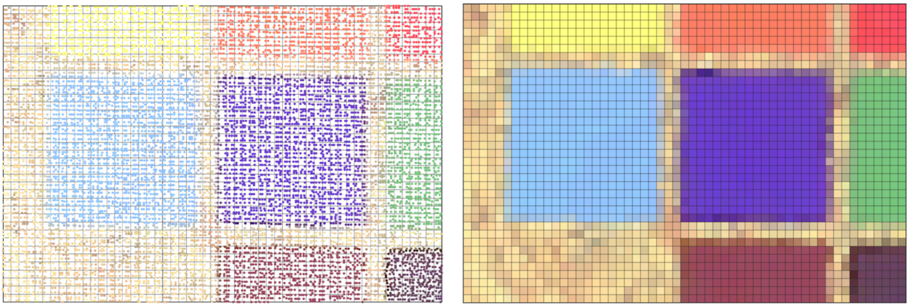

(a) Transformed point cloud and mesh. (b) Corresponding bitmap image.

Fig. 3 Creation of the bitmap images from the point cloud data visualised for R, G, B channels

The use of multiple channels will be described in the Sect. 2.7. The mathematical model for curve evolution is an extension of the model from the paper (Mikula et al. 2021a) and we employ a similar notation. The segmentation curve is an evolving closed planar curve with a position vector of its points denoted by $\mathbf{x}(u, t)$, where $u \in [0, 1]$ is a parameter that goes along the curve and $t \in [0, t_f]$ is the time (with t_f being a final time). The fact that the curve is closed means $\mathbf{x}(0, t) = \mathbf{x}(1, t)$.

The evolution is driven by a suitably designed velocity field $\mathbf{v}(\mathbf{x}, t)$, therefore, the basic evolution model is

$$\partial_t \mathbf{x}(u, t) = \mathbf{v}(u, t), \tag{1}$$

where $\partial_t \mathbf{x} := \frac{\partial \mathbf{x}}{\partial t}$ denotes the time derivative of the position vector, that is, the velocity of point $\mathbf{x}$. Equation (1) is coupled with an initial condition $\mathbf{x}(u, 0) = \mathbf{x}_0(u)$, $u \in [0, 1]$. The initial curve $\mathbf{x}_0$ is a small circle (Fig. 4) automatically placed inside the segmented region (e.g. the small white circle in Fig. 2b). Details on automatic insertion of the initial curve are provided in Sect. 2.8. We consider the velocity $\mathbf{v}(u, t)$ in the following form (see also Mikula et al. (2021a)):

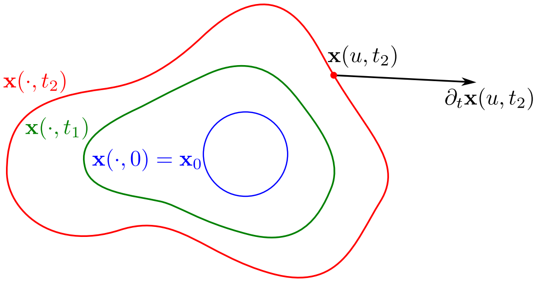
$$\mathbf{v}(u, t) = \big(1 - \lambda(t)\big) \underbrace{B(u, t)\, \mathbf{N}(u, t)}_{\text{Expansion}} + \lambda(t) \underbrace{\big(- \nabla E(\mathbf{x}(u, t))\big)}_{\text{Edge attraction}} + \delta(u, t) \underbrace{k(u, t)\, \mathbf{N}(u, t),}_{\text{Curvature regularisation}} \tag{2}$$

where $\mathbf{N}$ denotes the positively oriented unit normal vector defined by rotation of the unit tangential vector $\mathbf{T} = \frac{\partial \mathbf{x}}{\partial s}$ (also denoted as $\partial_s \mathbf{x} = (\partial_s x_1, \partial_s x_2)$) by $\frac{\pi}{2}$ in the positive direction, with $s = \int_0^u \|\partial_u \mathbf{x}\| \, du$ denoting the arc length parameter of the curve. Therefore, the normal can be expressed as $\mathbf{N} = \mathbf{T}^{\perp} = (-\partial_s x_2, \partial_s x_1)$. The function k in (2) is the signed curvature and ∇ is the gradient operator. Functions E and B will be properly defined later. Now, we give a brief description of the terms in (2). The role of the first term $B(u, t)\, \mathbf{N}(u, t)$ is to expand the segmentation curve in the normal direction from its initial shape through the segmented region towards its border, and the 'blowing' function $B(u, t)$ controls the speed of expansion.

The second term attracts the points of the curve towards the borders of the segmented region. Information about the borders is contained in the edge detector function $E(\mathbf{x}) \in [0, 1]$, $\mathbf{x} \in \Omega$ which has values close to 0 near the edges and close to 1 in homogeneous regions of the planar point cloud (see Fig. 5d). The negative gradient of E points towards the lower values and, therefore, is suitable to attract the segmentation curve towards the edges.

The time-dependent parameter $\lambda(t) \in [0, 1]$ serves as a weight between the expansion and the edge attraction term. The standard approach is to set $\lambda(t)$ at the beginning ($t = 0$) to a number $0 < \lambda_0 \ll 1$ (i.e. the edge attraction does not dominate), keep it unchanged until the curve is close to the border (moving very slowly) and then switch $\lambda(t)$ to 1, which turns

Fig. 4 An evolving closed planar curve plotted at times $t = 0, t_1, t_2$, where $0 < t_1 < t_2$



$\mathbf{x}(\cdot, t_2)$
$\mathbf{x}(\cdot, t_1)$
$\mathbf{x}(\cdot, 0) = \mathbf{x}_0$
$\mathbf{x}(u, t_2)$
$\partial_t \mathbf{x}(u, t_2)$

off the expansion term. The setting $\lambda(t) = \lambda_0$ will be called the expansion phase and the setting $\lambda(t) = 1$ we call the attraction phase.

The last term in (2) is called curvature regularisation and has a smoothing effect. We use it to deal with the noise and to smooth the sharp edges of the segmentation curve during the expansion phase. The importance of the curvature regularisation was studied, e.g. in Mikula et al. (2021b). The parameter $\delta(u, t)$ weights the influence of the term and will be properly defined in Sect. 2.7.

The first two terms in (2) could be also called together as 'external velocity', because they substantially depend on external quantities which are properties of the segmented image, unlike the curvature regularisation term, which depends on the signed curvature k and normal $\mathbf{N}$, which are properties of the curve. However, we may set the weight δ to be dependent also on the speed of the curve evolution, as we shall see later.

2.3 Edge detector

The (smoothed) edge detector function $E(\mathbf{x})$ is constructed as follows (Mikula et al. 2021a). First, we prefilter the original image $\mathcal{I}$, for example, using the Gaussian filter to get $\mathcal{I}_{\sigma_0} = G_{\sigma_0} * \mathcal{I}$ (the convolution with the Gaussian kernel G_{σ_0} with variance σ_0) and then compute the norm of the gradient of the filtered image $\left\|\nabla\mathcal{I}_{\sigma_0}(\mathbf{x})\right\|$. If the original image is very noisy and strong filtration is needed, we can use the anisotropic diffusion (Perona–Malik) filter (Perona and Malik 1990) instead. However, in some situations, the prefiltration is not necessary because some filtration is actually done in the preprocessing step (Sect. 2.1) when computing pixel representations of the point cloud data, which is akin to a mean filter. In such cases, one may directly compute the norm of the gradient of the original image $\left\|\nabla\mathcal{I}(\mathbf{x})\right\|$.

Next, we find edges in the image $\mathcal{I}$ by computing an edge detector function

$$g(\mathbf{x}) = \frac{1}{1 + \mu\left\|\nabla\mathcal{I}_{\sigma_0}(\mathbf{x})\right\|^2}. \tag{3}$$

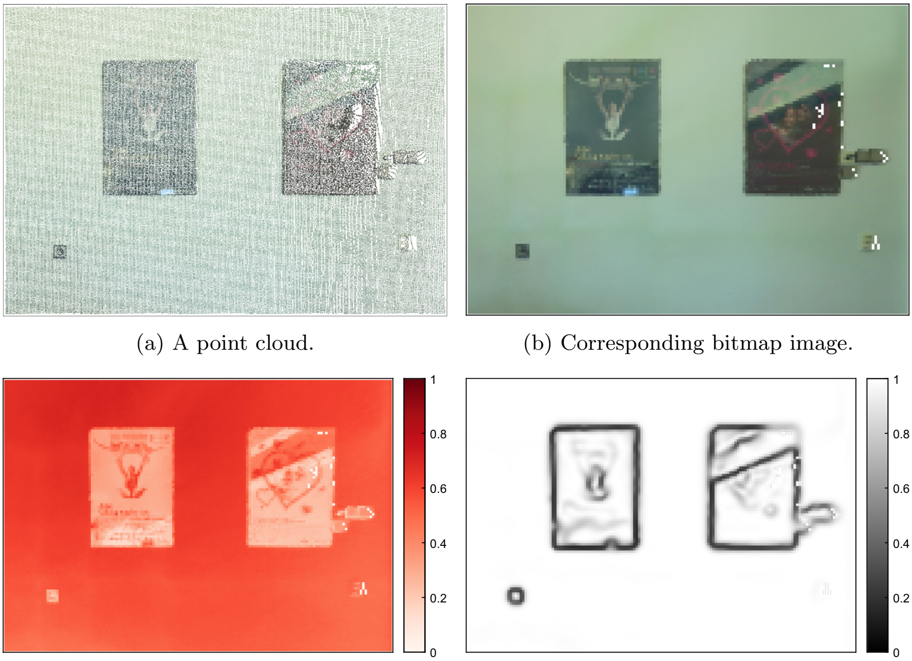
The gradients are large near the edges, and therefore, $g(\mathbf{x})$ is close to zero in such regions. In homogeneous areas, the gradients are small, and thus $g(\mathbf{x})$ is close to 1. The parameter $\mu > 0$ controls the sensitivity of edge detection. If it is small, the value of $g(\mathbf{x})$ is close to zero (detects an edge) only for very sharp edges (i.e. very large gradients). If the parameter μ is large, the function $g(\mathbf{x})$ has values close to zero even for smaller gradients, which means that $g(\mathbf{x})$ is more sensitive to edges. The reasonable values of μ are from the interval (0, 10].

Finally, we apply the Gaussian filter (convolve the function $g(\mathbf{x})$ with the Gaussian kernel with variance σ) to obtain the smoothed edge detector function

$$E(\mathbf{x}) = (G_\sigma * g)(\mathbf{x}). \tag{4}$$

Smoothing makes the edges 'wider' and causes the edge attraction term $-\nabla E(\mathbf{x})$ in (2) to point in the right direction (towards the edge) in a larger neighbourhood of the actual edge, which helps the edge attraction. The edges become too wide and blurry for large σ, so small values of σ are reasonable, we use $\sigma = 0.5$. In Fig. 5a, we can see a point cloud of a wall with two pictures. Image $\mathcal{I}$ for the red channel is in Fig. 5c and the corresponding smoothed edge detector function $E(\mathbf{x})$ is plotted in Fig. 5d.

We note that if image $\mathcal{I}$ contains holes (as in Fig. 7b), i.e. not a number (NaN) values corresponding to empty pixels (containing no points of the point cloud), the gradients and filtration need to be computed carefully. For example, if we carelessly calculate the derivative $\partial_x\mathcal{I}$ in the (i, j)-th pixel by the finite difference $(\partial_x\mathcal{I})_{ij} \approx (\mathcal{I}_{i+1,j} - \mathcal{I}_{i-1,j})/2h$ and the value



(a) A point cloud.

(b) Corresponding bitmap image.

(c) Image $\mathcal{I}$ corresponding to the red channel.

(d) Smoothed edge detector function $E(\mathbf{x})$.

Fig. 5 Construction of the smoothed edge detector function for the red channel

$\mathcal{I}_{i-1,j}$ or $\mathcal{I}_{i+1,j}$ would be NaN, the result would be NaN. Therefore, we first interpolate (or extrapolate) empty pixels, then compute the finite difference, and eventually reset the NaN values to preserve the information about the holes (if $\mathcal{I}_{ij}$ was NaN, then reset $(\partial_x \mathcal{I})_{ij}$ to NaN). If a point $\mathbf{x}(u,t)$ of the curve lies in an empty (NaN) pixel, we set both the expansion and edge attraction term in (2) to zero.

2.4 Blowing function

The blowing function B used in the expansion term in (2) is defined as (see also Mikula et al. (2021a))

$$B(u,t) = H(u,t)\, E(\mathbf{x}(u,t)), \tag{5}$$

where H is called the homogeneity function and is defined as

$$H(u,t) = \begin{cases} 1, & d(u,t) < d_0, \\ 0, & \text{otherwise}. \end{cases} \tag{6}$$

where d_0 is a threshold value and $d(u,t) = \left|\mathcal{I}(\mathbf{x}(u,t)) - \langle \mathcal{I} \rangle^t\right|$ is the difference function which computes the (absolute value of the) difference between the value $\mathcal{I}(\mathbf{x}(u,t))$ and the average value $\langle \mathcal{I} \rangle^t := \frac{1}{|V(t)|} \int_{V(t)} \mathcal{I}\, d\mathbf{x}$ in the region $V(t)$, which denotes the set of points $\mathbf{x} \in \Omega$ visited by the curve since the beginning of the evolution, together with points inside the initial curve (i.e. $V(0)$ is the interior of the initial curve). If the value of the image $\mathcal{I}$ at a point $\mathbf{x}(u, t)$ of the curve is similar to the points $\mathbf{x} \in \Omega$ already segmented by the curve, then the difference $d(u, t)$ is small and the point $\mathbf{x}(u, t)$ should move, which is provided by homogeneity $H(u, t) = 1$ for small $d(u, t)$. On the contrary, if the difference $d(u, t)$ greater than threshold d_0, the definitions (5), (6) give zero expansion of the curve at the point $\mathbf{x}(u, t)$.

A natural definition of the blowing function could simply be $B = H$. However, then the expanding term could sometimes drive the points of the curve across the edges. This is the reason why we multiply the homogeneity function by the edge detector function E (which is close to zero in the neighbourhood of edges).

2.5 Normal and tangential speed

The basic evolution model (1) moves the points on the curve both in the normal and tangential directions. From a continuous (analytical) point of view, only the normal component $\beta = \mathbf{v} \cdot \mathbf{N}$ of the velocity (2) has an effect on the shape of the curve. Therefore, we could use the evolution model $\partial_t \mathbf{x} = \beta \mathbf{N}$. However, in the discrete setting, it is convenient to enrich this model with a suitable tangential speed α which can be designed to control the distribution of the discretisation points along the curve. This becomes crucial in numerical computations because inappropriate positions of the points can lead to unacceptable errors. Therefore, instead of the basic model (1), we use the model

$$\partial_t \mathbf{x} = \beta \mathbf{N} + \alpha \mathbf{T}, \tag{7}$$

where $\beta = \mathbf{v} \cdot \mathbf{N}$ is the normal speed calculated from (2) as

$$\beta = \beta_e + \delta k, \qquad \text{where} \qquad \beta_e = \big(1 - \lambda\big)\, B - \lambda \nabla E \cdot \mathbf{N} \tag{8}$$

is the so-called external normal speed.

The tangential speed α is constructed to achieve a uniform distribution of the points in the discrete setting. We follow the approach from Mikula and Ševčovič (2004). The length of the curve segment between the points $\mathbf{x}(\hat{u}, t)$ and $\mathbf{x}(\hat{u} + \Delta u, t)$ is $\int_{\hat{u}}^{\hat{u}+\Delta u} g\, du$, where $g := \|\partial_u \mathbf{x}\|$. We want this length to be asymptotically (for $t \to \infty$) same for every $\hat{u}$ (each pair of points), since this corresponds to a uniform distribution of grid points on the discretised curve. However, the total length of the curve $L = \int_0^1 g\, du$ changes during evolution, so we have to study the relative length of the segment (with respect to the total length). For an asymptotically uniform redistribution of points, the relative length of the segment approaches a constant

$$\frac{\int_{\hat{u}}^{\hat{u}+\Delta u} g\, du}{L} \to \text{const.} \qquad \text{for every } \hat{u}.$$

This is satisfied if $\frac{g}{L} \to c$, where $c > 0$ is a constant. Since $\int_0^1 \frac{g}{L} du = 1$, we have to set $c = 1$ and the condition is $\lim\limits_{t \to \infty} \frac{g}{L} = 1$. This is fulfilled if the ratio $\frac{g}{L}$ obeys a relaxation equation

$$\partial_t \left(\frac{g}{L}\right) = \left(1 - \frac{g}{L}\right) \omega, \tag{9}$$

where the relaxation parameter $\omega > 0$ controls the speed of relaxation. On the left-hand side, we need a formula for the derivative of g and L. The first one is

$$\begin{aligned}\partial_t g &= \partial_t \|\partial_u \mathbf{x}\| \overset{1.}{=} \frac{\partial_u \mathbf{x}}{\|\partial_u \mathbf{x}\|} \cdot \partial_t \partial_u \mathbf{x} \overset{2.}{=} \partial_s \mathbf{x} \cdot \partial_u (\partial_t \mathbf{x}) \overset{3.}{=} \mathbf{T} \cdot \partial_u (\beta \mathbf{N} + \alpha \mathbf{T}) \\ &\overset{2.}{=} g\mathbf{T} \cdot \partial_s (\beta \mathbf{N} + \alpha \mathbf{T}) = g\mathbf{T} \cdot \big((\partial_s \beta)\mathbf{N} + \beta \partial_s \mathbf{N} + (\partial_s \alpha)\mathbf{T} + \alpha \partial_s \mathbf{T}\big) \overset{4.}{=} -gk\beta + g\partial_s \alpha,\end{aligned} \tag{10}$$

where we used:

1. Derivative of the norm $\partial_t \|\mathbf{w}\| = \partial_t \sqrt{\mathbf{w} \cdot \mathbf{w}} = \frac{\mathbf{w}}{\|\mathbf{w}\|} \cdot \partial_t \mathbf{w}$.
2. Chain rule $\frac{\partial}{\partial u} = \frac{\partial s}{\partial u} \frac{\partial}{\partial s} = \|\partial_u \mathbf{x}\| \partial_s = g \partial_s$ and $\partial_t \partial_u = \partial_u \partial_t$.
3. Equation (7) and the definition of the unit tangent vector $\mathbf{T} = \partial_s \mathbf{x}$.
4. Frenet–Serret formulas $\partial_s \mathbf{T} = k\mathbf{N}$, $\partial_s \mathbf{N} = -k\mathbf{T}$ and orthonormality of the basis $\mathbf{T}$, $\mathbf{N}$ (i.e. $\mathbf{T} \cdot \mathbf{T} = 1$, $\mathbf{N} \cdot \mathbf{N} = 1$ and $\mathbf{T} \cdot \mathbf{N} = 0$).

Integrating (10), we obtain the equation for the derivative of the curve length

$$\partial_t L = \int_0^1 \partial_t g \, du = -\int_0^1 gk\beta \, du + \int_0^1 g\partial_s \alpha \, du = -\int_0^L k\beta \, ds = -L\langle k\beta \rangle, \tag{11}$$

where $\langle k\beta \rangle := \frac{1}{L} \int_0^L k\beta \, ds$ denotes the average value of $k\beta$ over the curve. In the above calculation we used $ds = \partial_u s \, du = g \, du$ and $\int_0^1 g\partial_s \alpha \, du = \int_0^1 \partial_u \alpha \, du = \alpha(1,t) - \alpha(0,t) = 0$, which is provided by periodic boundary conditions $\alpha(1,t) = \alpha(0,t)$.

The application of the quotient rule in (9) and the use of formulas (10), (11) give

$$\partial_t \left(\frac{g}{L}\right) = \frac{(\partial_t g)L - g\,\partial_t L}{L^2} = \frac{(-gk\beta + g\partial_s \alpha)L + gL\langle k\beta \rangle}{L^2} = \left(1 - \frac{g}{L}\right)\omega.$$

After multiplication by $\frac{L}{g}$ and rearrangement, we obtain the equation for the tangential speed α

$$\partial_s \alpha = k\beta - \langle k\beta \rangle + \left(\frac{L}{g} - 1\right)\omega. \tag{12}$$

To ensure a unique solution, we can simply fix the value of α in one point (e.g. $\alpha(0,t) = 0$). However, this can lead to a larger tangential motion than is needed for uniform redistribution. Therefore, to minimise the tangential motion, we ensure uniqueness by requiring zero average tangential motion $\langle \alpha \rangle = \frac{1}{L} \int_0^L \alpha \, ds = 0$ for each $t \in [0, t_f]$, see also Ambroz et al. (2019).

2.6 Curvature regularisation

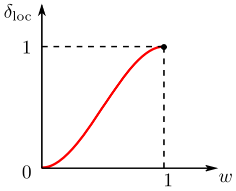
The weight $\delta(u,t)$ of the curvature regularisation term in (2) and (8) can be simply a constant $\delta(u,t) = \delta_k$ (usually a small number, e.g. 0.01). However, it is convenient to use a more sophisticated setting that considers the phase of the evolution and the relative speed of the point $\mathbf{x}(u,t)$

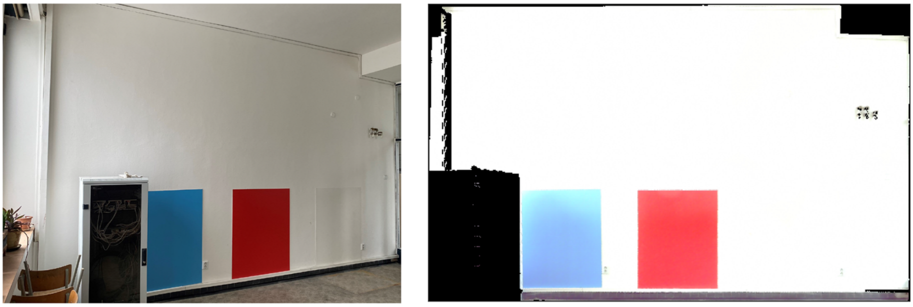
$$\delta(u,t) = \begin{cases} \delta_k \delta_{\text{loc}}(u,t), & \lambda < 1, \quad \text{(Expansion phase)} \\ 0, & \lambda = 1, \quad \text{(Attraction phase)} \end{cases} \tag{13}$$

where the local weight $\delta_{\text{loc}}(u,t)$ is defined as follows (Fig. 6):

$$\delta_{\text{loc}} = 3w^2 - 2w^3, \qquad \text{where} \qquad w(u,t) = \frac{\beta_e(u,t)}{\max_{u \in [0,1]} \beta_e(u,t)}.$$

Fig. 6 Local weight





(a) Photo of the scene.

(b) Image corresponding to R, G, B channels.

Fig. 7 A white wall with three plastic plates

Thus, in the expansion phase, the proposed weight $\delta(u, t)$ is a function of the relative external normal speed $w(u, t)$ of the point $\mathbf{x}(u, t)$. For slow points, we apply a small regularisation[4] weight, and for fast points, we set a high regularisation.[5] In the attraction phase, we turn off curvature regularisation, which enables sharp corners of the final segmentation curve.

2.7 Multichannel segmentation

So far, we have only discussed single-channel segmentation. Now, we propose a technique to use information from multiple channels and demonstrate it in a simple but illustrative example in Fig. 7. There is a white wall with three plastic plates: blue, red, and white. There are holes (NaN values, black) in Fig. 7b in parts of the planar point cloud in which there are no point cloud points within the threshold distance from the fitting plane. Let us denote the images corresponding to the channels R, G, B, I and D as $\mathcal{I}^l$, $l = 1, \ldots, 5$, respectively, see the left column in Fig. 8.

First, we will discuss the construction of the smoothed edge detector function E and then the computation of the blowing function B. In the right column of Fig. 8, we can see the edge detector functions $g^l(\mathbf{x})$ corresponding to the images $\mathcal{I}^l$ computed using (3). In practice, we can set different sensitivity parameter μ^l for each channel. Notice that some edges are discovered in one edge detector $g^l(\mathbf{x})$ but not in another, e.g. the red plate (with (R,G,B) $\approx (1, 0, 0)$) is 'invisible' for the R channel (because the wall is white, i.e. (R,G,B) $\approx (1, 1, 1)$, see first row of Fig. 8), but it is clearly distinguished by the B channel (and vice versa for the blue panel). The intensity edge detector (fourth row in Fig. 8) revealed multiple edges that are not seen in the colour (R, G, B) channels[6]: the edge of the white plate, electrical outlets,

[4] This prevents the curve to overcome an edge (on which the points should stop) by means of curvature.

[5] Fast-moving parts of the curve will have a more rounded shape.

[6] The colour channels can hardly distinguish white objects on a white wall.

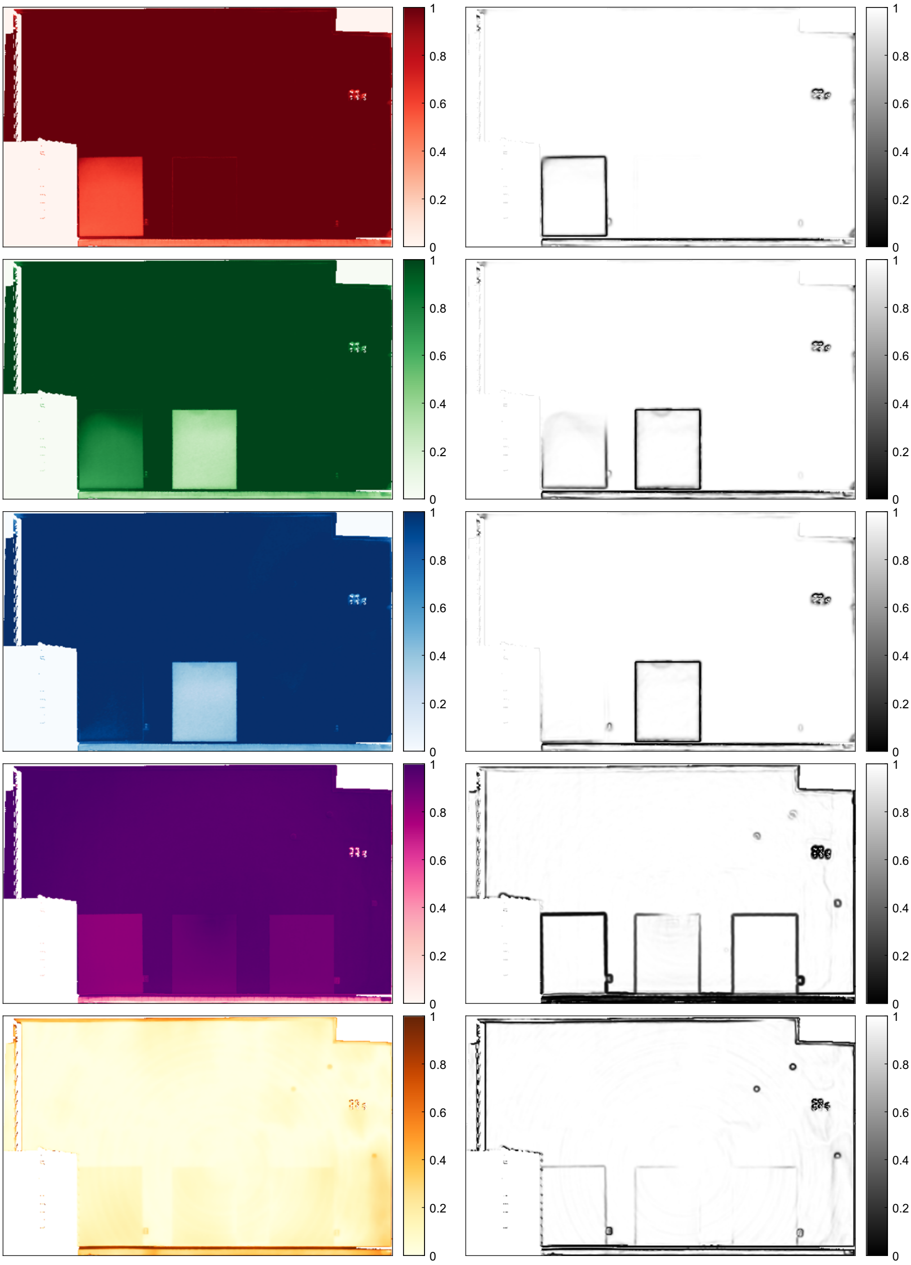


Fig. 8 Left column: Bitmap images $\mathcal{I}^l$, $l = 1, \ldots, 5$ generated from the channels R, G, B, I, D of the point cloud. Right column: corresponding edge detector functions g^l, $l = 1, \ldots, 5$

light switch, and electrical box covers. These objects have different reflexivity than the wall, and this is captured by the intensity channel which is related to the strength of the reflected laser signal. The electrical outlets, light switch and box covers were also clearly detected by the D channel, see the last row of Fig. 8.

A natural idea is to use the data from all available channels R, G, B, I, D (sometimes the colour data might be missing) to construct one edge detector. Therefore, we aggregate the edge detector functions $g^l(\mathbf{x}), l = 1, \dots, 5$ into a single edge detector function

$$g(\mathbf{x}) = \prod_{l=1}^{5} g^l(\mathbf{x}), \tag{14}$$

which contains aggregated information about all edges in all channels, see Fig. 9. Indeed, if at a point $\mathbf{x}$ any (even one) of the edge detectors has a value $g^l(\mathbf{x})$ close to zero, then the value $g(\mathbf{x})$ will be close to zero. Moreover, if multiple channels have values close to zero at a point $\mathbf{x}$, then the resulting value $g(\mathbf{x})$ will be much closer to zero. This approach is also convenient, in the situation if multiple edge detectors agree on some mild edge, the resulting aggregated edge will be more distinct. Finally, we smooth the edge detector function (14) using (4) to obtain the final smoothed aggregated edge detector function $E(\mathbf{x})$ and use it in the curve segmentation.

Since the range of the edge detector function (3) is $(0, 1]$, the aggregation (14) can be interpreted as fuzzy logic aggregation using the product aggregation operator $A : [0, 1]^n \to [0, 1]$ defined as $A(x_1, \dots, x_n) = \prod_{i=1}^{n} x_i$, see, e.g. van Krieken et al. (2022). Fuzzy logic in general is a many-valued logic in which the truth values of variables x_i are real numbers from the interval $[0, 1]$, i.e. there is a concept of partial truth while $x_i = 0$ represents completely false and $x_i = 1$ is completely true. The product is a generalisation of the AND operator from Boolean logic: the conjunction $x_1 \wedge x_2$ with $x_1, x_2 \in \{0, 1\}$ generalises to $x_1 \cdot x_2$ for $x_1, x_2 \in [0, 1]$.

Now, we shall briefly discuss the aggregated blowing function B. We compute it using (5) as in the single-channel case. However, now E is the smoothed aggregated edge detector function described above and H is the aggregated homogeneity function computed as

$$H(u, t) = H^1(u, t) \wedge \cdots \wedge H^5(u, t), \tag{15}$$

where $H^l(u, t) \in \{0, 1\}$ denotes the homogeneity function computed using (6) for the image $\mathcal{I}^l$. Equation (15) guarantees that the value $B(u, t)$ of the blowing function is nonzero only if the difference $d^l(u, t) = \left|\mathcal{I}^l(\mathbf{x}(u, t)) - \langle \mathcal{I}^l \rangle^t\right|$ is smaller than the threshold d_0^l for each channel. If $d^l(u, t) \geq d_0^l$ for some channel,[7] then the blowing function (expansion term) vanishes at the point $\mathbf{x}(u, t)$ of the curve. The threshold value d_0^l can be set differently for each channel. However, in this paper, we will use $d_0^l = 0.5$ for all channels.

2.8 Inserting initial curves

At the beginning of the segmentation, we insert a small circle (or multiple circles[8]) into the image. We use the radius $r_0 = 3h$, with h denoting the pixel size (edge length). In order to find suitable candidates for positions of the initial circles, we construct an initialisation mask M, Fig. 10. First, we filter the smoothed edge detector $E_{\sigma_M} = G_{\sigma_M} * E$, where we use the

[7] Meaning the value $\mathcal{I}^l(\mathbf{x}(u, t))$ is not similar to the average of $\mathcal{I}^l$ in already segmented region $V(t)$.

[8] A reasonable number is one to several dozen.

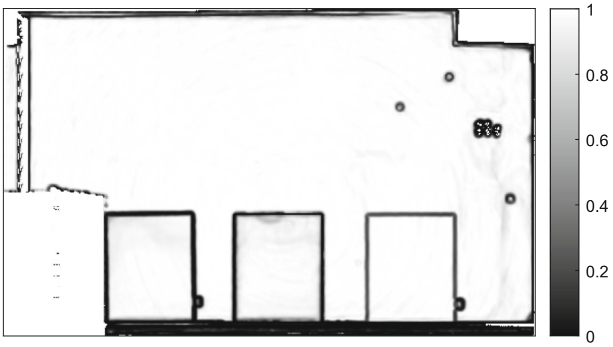


Fig. 9 The smoothed aggregated edge detector function $E(\mathbf{x})$

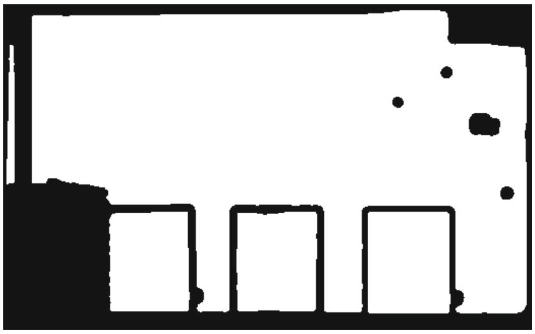


Fig. 10 Initialisation mask computed for the wall with three plates. White regions with value $M(\mathbf{x}) = 1$ are suitable for placing initial curves

Gaussian filter with a higher standard deviation (e.g. $\sigma_M = 2$) and a larger kernel size (e.g. 7×7). Then, we threshold the function E_{σ_M} as follows:

$$M(\mathbf{x}) = \begin{cases} 0, & E_{\sigma_M}(\mathbf{x}) < 0.7 \quad \text{or} \quad E_{\sigma_M}(\mathbf{x}) = \text{NaN}, \\ 1, & \text{otherwise}. \end{cases} \tag{16}$$

The region where $M(\mathbf{x}) = 1$ will be referred to as segmentable region $\Omega_S \subset \Omega$. We note that here we perform a careless filtration, which intentionally spreads NaN values to their neighbourhood (differently from the Sect. 2.3). The centres of the initial circles are randomly placed in the segmentable region Ω_S (more precisely to random pixel centres in Ω_S). The construction of mask M prevents the insertion of circles on the edges, holes (NaN values), and their neighbourhood.

If all curves stop evolving, we remove the segmented region $V(t)$ of each curve from the segmentable region by setting $M(\mathbf{x}) = 0$ for all $\mathbf{x} \in V(t)$. Subsequently, we randomly add a new circle (or multiple circles) into the segmentable region. If we insert multiple circles (either at the beginning or during the process of segmentation), it is convenient to remove the interior and possibly some neighbourhood of a circle from the segmentable region after each curve insertion. This prevents the generation of intersecting or too close curves. When a segmentation target is reached, e.g. 99 % of the initial segmentable area is segmented, we end the segmentation process.

2.9 Postprocessing: creation of the point cloud segments

When the curve segmentation is finished, the planar point cloud is divided into segments. We take the segmented region $V(t_f)$ of each segmentation curve and select all points in the point cloud that lie in $V(t_f)$. The last segment is a 'waste after segmentation' and corresponds to regions of Ω which were not visited by any curve.

3 Numerical scheme

In this section, we discuss the numerical discretisation of our method in great detail. For the purpose of the derivation of the numerical scheme, let us rewrite the model (7), (8) using $\mathbf{T} = \partial_s \mathbf{x}$, $\mathbf{N} = \partial_s \mathbf{x}^{\perp}$ and the Frenet–Serret formula $k\mathbf{N} = \partial_s \mathbf{T} = \partial_s^2 \mathbf{x}$. We obtain the following equation:

$$\partial_t \mathbf{x} - \alpha \partial_s \mathbf{x} = \delta \partial_s^2 \mathbf{x} + \beta_e \partial_s \mathbf{x}^{\perp}, \tag{17}$$

with is actually a system of two PDEs, since $\mathbf{x} = (x_1, x_2)$ is the unknown position vector. Equation (17) has a form of so-called intrinsic PDE. The tangential term $-\alpha \partial_s \mathbf{x}$ represents an intrinsic advection along the curve, with $-\alpha$ being the speed of the advection, and the curvature term $\delta \partial_s^2 \mathbf{x}$ can be regarded as an intrinsic diffusion with diffusion coefficient δ.

Now, we will discretize equation (17). First, we perform the space discretisation which is based on the finite-volume method (Mikula et al. 2021a). Then, we present the time discretisation both by the standard semi-implicit scheme (Ševčovič and Mikula 2001) and semi-implicit scheme with Inflow-Implicit/Outflow-Explicit (IIOE) technique to treat the advection term (Mikula and Ohlberger 2011; Balažovjech et al. 2012; Mikula et al. 2014).

3.1 Space discretisation

The mesh will consist of m grid points $\mathbf{x}_i$, $i = 1, \ldots, m$, where $\mathbf{x}_i(t) \approx \mathbf{x}(u_i, t)$, $u_i = (i-1)\Delta u$ and $\Delta u = \frac{1}{m}$. The construction of the finite volume $\mathcal{V}_i$ corresponding to the point $\mathbf{x}_i$ consists of two line segments connecting the points $\mathbf{x}_{i-\frac{1}{2}} := \frac{\mathbf{x}_{i-1}+\mathbf{x}_i}{2}$, $\mathbf{x}_i$ and $\mathbf{x}_{i+\frac{1}{2}} := \frac{\mathbf{x}_i+\mathbf{x}_{i+1}}{2}$, Fig. 11. The size of $\mathcal{V}_i$ is $|\mathcal{V}_i| = \frac{h_{i-1}+h_i}{2}$, where $h_i := \|\mathbf{x}_{i+1} - \mathbf{x}_i\|$ is the length of the segment $\{\mathbf{x}_i, \mathbf{x}_{i+1}\}$. We integrate equation (17) over the finite volume $\mathcal{V}_i$ and obtain

$$\partial_t \mathbf{x}_i \int_{\mathbf{x}_{i-\frac{1}{2}}}^{\mathbf{x}_{i+\frac{1}{2}}} ds - \alpha_i \int_{\mathbf{x}_{i-\frac{1}{2}}}^{\mathbf{x}_{i+\frac{1}{2}}} \partial_s \mathbf{x}\, ds = \delta_i \int_{\mathbf{x}_{i-\frac{1}{2}}}^{\mathbf{x}_{i+\frac{1}{2}}} \partial_s^2 \mathbf{x}\, ds + \beta_{e,i} \int_{\mathbf{x}_{i-\frac{1}{2}}}^{\mathbf{x}_{i+\frac{1}{2}}} \partial_s \mathbf{x}^{\perp}\, ds,$$

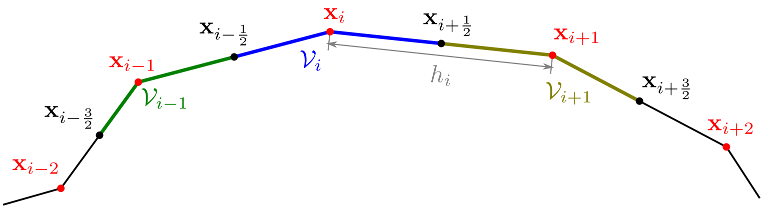


Fig. 11 The notation on the curve mesh

where we approximate some quantities by constants on $\mathcal{V}_i$, namely $\partial_t \mathbf{x} \approx \partial_t \mathbf{x}_i$, $\alpha \approx \alpha_i$, $\delta \approx \delta_i$ and $\beta_e \approx \beta_{e,i}$. After using the Newton–Leibniz formula, we have the following:

$$\partial_t \mathbf{x}_i |\mathcal{V}_i| - \alpha_i [\mathbf{x}]_{\mathbf{x}_{i-\frac{1}{2}}}^{\mathbf{x}_{i+\frac{1}{2}}} = \delta_i [\partial_s \mathbf{x}]_{\mathbf{x}_{i-\frac{1}{2}}}^{\mathbf{x}_{i+\frac{1}{2}}} + \beta_{e,i} \left([\mathbf{x}]_{\mathbf{x}_{i-\frac{1}{2}}}^{\mathbf{x}_{i+\frac{1}{2}}} \right)^{\perp},$$

which evaluates to

$$\partial_t \mathbf{x}_i \frac{h_{i-1} + h_i}{2} - \alpha_i \left(\mathbf{x}_{i+\frac{1}{2}} - \mathbf{x}_{i-\frac{1}{2}} \right) = \delta_i [\partial_s \mathbf{x}]_{\mathbf{x}_{i-\frac{1}{2}}}^{\mathbf{x}_{i+\frac{1}{2}}} + \beta_{e,i} \left(\mathbf{x}_{i+\frac{1}{2}} - \mathbf{x}_{i-\frac{1}{2}} \right)^{\perp}.$$

Finally, we approximate the derivative $\partial_s \mathbf{x}$ in the midpoints $\mathbf{x}_{i-\frac{1}{2}}$, $\mathbf{x}_{i+\frac{1}{2}}$ using the finite differences $\partial_s \mathbf{x}_{i-\frac{1}{2}} \approx \frac{\mathbf{x}_i - \mathbf{x}_{i-1}}{h_{i-1}}$ and $\partial_s \mathbf{x}_{i+\frac{1}{2}} \approx \frac{\mathbf{x}_{i+1} - \mathbf{x}_i}{h_i}$, respectively. The space discretisation of the PDE (17) reads

$$\begin{aligned} \frac{h_{i-1} + h_i}{2} \partial_t \mathbf{x}_i - \alpha_i \left(\frac{\mathbf{x}_{i+1} - \mathbf{x}_{i-1}}{2} \right) &= \delta_i \left(\frac{\mathbf{x}_{i+1} - \mathbf{x}_i}{h_i} - \frac{\mathbf{x}_i - \mathbf{x}_{i-1}}{h_{i-1}} \right) \\ &\quad + \beta_{e,i} \left(\frac{\mathbf{x}_{i+1} - \mathbf{x}_{i-1}}{2} \right)^{\perp}, \end{aligned} \tag{18}$$

for each $i = 1, \ldots, m$, with $\mathbf{x}_0 := \mathbf{x}_m$ and $\mathbf{x}_{m+1} := \mathbf{x}_1$.

3.2 Time discretisation by semi-implicit scheme

In this paper, we show two different time discretisations. In this section, we perform the time discretisation using a semi-implicit scheme (Ševčovič and Mikula 2001) and motivate the need for a more sophisticated semi-implicit IIOE scheme (Mikula and Ohlberger 2011; Balažovjech et al. 2012; Mikula et al. 2014), which is presented in the Sect. 3.3 and used in our implementation. The motivation is given from a different point of view from those of (Mikula and Ohlberger 2011; Mikula et al. 2014; Balažovjech et al. 2012).

We discretize the time interval $[0, t_f]$ using discrete times $t^n, n = 0, \ldots, n_{\max}$ with $t^0 = 0$ and $t^{n_{\max}} = t_f$. Let us denote the length of the time step by $\tau^n = t^{n+1} - t^n$. The discretisation is not necessarily uniform in time, and we usually choose a smaller τ^n in the attraction phase. We approximate the time derivative in (18) by the forward finite difference $\partial_t \mathbf{x}_i \approx \frac{\mathbf{x}_i^{n+1} - \mathbf{x}_i^n}{\tau^n}$, where $\mathbf{x}_i^n \approx \mathbf{x}_i(t^n)$. Then, we take the coefficients $h_i, \alpha_i, \delta_i, \beta_{e,i}$ from the n-th (old) time step, the advection and diffusion terms are approximated implicitly, that is, we take $\mathbf{x}_i$ from $(n+1)$-th (new) time step, and the expansion term explicitly, meaning that $\mathbf{x}_i$ is taken from the old time step. We obtain the following:

$$\begin{aligned} &\frac{h_{i-1}^n + h_i^n}{2} \frac{\mathbf{x}_i^{n+1} - \mathbf{x}_i^n}{\tau^n} - \alpha_i^n \left(\frac{\mathbf{x}_{i+1}^{n+1} - \mathbf{x}_{i-1}^{n+1}}{2} \right) \\ &\quad = \delta_i^n \left(\frac{\mathbf{x}_{i+1}^{n+1} - \mathbf{x}_i^{n+1}}{h_i^n} - \frac{\mathbf{x}_i^{n+1} - \mathbf{x}_{i-1}^{n+1}}{h_{i-1}^n} \right) + \beta_{e,i}^n \left(\frac{\mathbf{x}_{i+1}^n - \mathbf{x}_{i-1}^n}{2} \right)^{\perp}, \end{aligned}$$

which can be rearranged as

$$-A_i^n \mathbf{x}_{i-1}^{n+1} + B_i^n \mathbf{x}_i^{n+1} - C_i^n \mathbf{x}_{i+1}^{n+1} = \mathbf{D}_i^n, \qquad i = 1, \ldots, m \tag{19}$$

for $n = 0, \ldots, n_{\max} - 1$. We denoted

$$
\begin{aligned}
A_i^n &:= \tfrac{\delta_i^n}{h_{i-1}^n} - \tfrac{\alpha_i^n}{2}, \qquad & B_i^n &:= \tfrac{h_{i-1}^n + h_i^n}{2\tau^n} + \delta_i^n \left(\tfrac{1}{h_i^n} + \tfrac{1}{h_{i-1}^n} \right), \\
C_i^n &:= \tfrac{\delta_i^n}{h_i^n} + \tfrac{\alpha_i^n}{2}, \qquad & \mathbf{D}_i^n &:= \tfrac{h_{i-1}^n + h_i^n}{2\tau^n} \mathbf{x}_i^n + \beta_{e,i}^n \left(\tfrac{\mathbf{x}_{i+1}^n - \mathbf{x}_{i-1}^n}{2} \right)^{\perp}.
\end{aligned}
\tag{20}
$$

The formula (19) is a system of m vector equations ($2m$ scalar equations) with unknowns $\mathbf{x}_i^{n+1}$, $i = 1, \ldots, m$. The matrix form of the system is

$$
\begin{bmatrix}
B_1^n & -C_1^n & 0 & \ldots & 0 & -A_1^n \\
-A_2^n & B_2^n & -C_2^n & 0 & \ldots & 0 \\
0 & -A_3^n & B_3^n & -C_3^n & 0 & \\
\vdots & & & \ddots & & \vdots \\
-C_m^n & & \ldots & & -A_m^n & B_m^n
\end{bmatrix}
\begin{bmatrix}
\mathbf{x}_1^{n+1} \\ \\ \vdots \\ \\ \mathbf{x}_m^{n+1}
\end{bmatrix}
=
\begin{bmatrix}
\mathbf{D}_1^n \\ \\ \vdots \\ \\ \mathbf{D}_m^n
\end{bmatrix}.
\tag{21}
$$

It is a cyclic tridiagonal system.[9] Such systems can be solved by a modified Thomas algorithm (also known as tridiagonal matrix algorithm) (Press et al. 2007).[10] If the system matrix is diagonally dominant, the Thomas algorithm is stable. The diagonal dominance in our case reads $|B_i^n| \geq |A_i^n| + |C_i^n|$, for all $i = 1, \ldots, m$, that is,

$$
\frac{h_{i-1}^n + h_i^n}{2\tau^n} + \delta_i^n \left(\frac{1}{h_i^n} + \frac{1}{h_{i-1}^n} \right) \geq \left| \frac{\delta_i^n}{h_{i-1}^n} - \frac{\alpha_i^n}{2} \right| + \left| \frac{\delta_i^n}{h_i^n} + \frac{\alpha_i^n}{2} \right|,
\tag{22}
$$

where we cancelled the absolute value on the LHS because all quantities are nonnegative. However, we cannot cancel it in the terms on the RHS, because $\alpha_i^n \in \mathbb{R}$. Therefore, the condition (22) is not always satisfied. We can achieve diagonal dominance with the appropriate choice of the time step length τ^n. From (22), we have the following:

$$
\tau^n \leq \frac{1}{2} \frac{h_{i-1}^n + h_i^n}{\left| \left(\frac{\delta_i^n}{h_{i-1}^n} - \frac{\alpha_i^n}{2} \right) \right| + \left| \left(\frac{\delta_i^n}{h_i^n} + \frac{\alpha_i^n}{2} \right) \right| - \delta_i^n \left(\frac{1}{h_i^n} + \frac{1}{h_{i-1}^n} \right)},
\tag{23}
$$

which must hold for each $i = 1, \ldots, m$, see also Balažovjech and Mikula (2011). Therefore, we can evaluate the RHS of (23) for each i and find the minimum that will give us the suitable time step length τ^n. Such computation of τ^n slows down the evolution algorithm and, moreover, if we obtain a very small τ^n, the evolution algorithm would be even slower. However, there is a trick (Mikula et al. 2014) which will guarantee stability without any restriction of the time step length.

Now, we give a motivation for the trick. Let us consider the case with no tangential speed, $\alpha_i^n = 0$. Then, we can cancel the absolute values also on the RHS of (22) and obtain the condition $\frac{h_{i-1}^n + h_i^n}{2\tau^n} \geq 0$ which holds (even strictly) for any τ^n, thus for zero tangential speed the system matrix is strictly diagonally dominant. In the case of non-vanishing tangential speed, the idea will be the same, i.e. to obtain the system matrix for which we can cancel all absolute values in diagonal dominance condition and then cancel out all terms except $\frac{h_{i-1}^n + h_i^n}{2\tau^n}$.

[9] Since the subsystems for x_1 and x_2 coordinates are independent of each other, we can solve the subsystems independently. If we would not treat the expansion term explicitly, we would end up with a more complicated system with $2m$ by $2m$ matrix with 5 nonzero elements in each row.

[10] The presence of the coefficients A_1^n and C_m^n does not allow to use the Thomas algorithm directly. However, one can use a trick from Press et al. (2007) to obtain the standard tridiagonal matrix and then use the Thomas algorithm.

3.3 Time discretisation by semi-implicit IIOE scheme

In this section, we present the semi-implicit Inflow-Implicit/Outflow-Explicit (IIOE) scheme (Mikula and Ohlberger 2011; Balažovjech et al. 2012; Mikula et al. 2014). The goal is to obtain a diagonally dominant matrix with no restriction on the time step length also in the case with nonzero tangential speed. We have seen that the tangential term causes problems. We will take a term containing α_i^n implicitly (i.e. it will occur in the system matrix) only if it has a suitable sign; otherwise, we will take it explicitly (move it to the RHS). By a 'suitable sign', we mean the sign that will allow us to cancel all absolute values in the diagonal dominance condition.

First, we rearrange the tangential term in (18) as follows:

$$\begin{aligned} -\alpha_i\left(\frac{\mathbf{x}_{i+1}-\mathbf{x}_{i-1}}{2}\right) &= \frac{1}{2}(-\alpha_i)(-\mathbf{x}_{i-1}) + \frac{1}{2}\alpha_i(-\mathbf{x}_{i+1}) + \underbrace{\frac{1}{2}(-\alpha_i)\mathbf{x}_i + \frac{1}{2}\alpha_i\mathbf{x}_i}_{0} \\ &= \frac{1}{2}(-\alpha_i)(\mathbf{x}_i - \mathbf{x}_{i-1}) + \frac{1}{2}\alpha_i(\mathbf{x}_i - \mathbf{x}_{i+1}) \\ &= \frac{1}{2}\left(b^{\text{in}}_{i-\frac{1}{2}} + b^{\text{out}}_{i-\frac{1}{2}}\right)(\mathbf{x}_i - \mathbf{x}_{i-1}) + \frac{1}{2}\left(b^{\text{in}}_{i+\frac{1}{2}} + b^{\text{out}}_{i+\frac{1}{2}}\right)(\mathbf{x}_i - \mathbf{x}_{i+1}), \end{aligned} \tag{24}$$

where we introduced the notations

$$\begin{aligned} b^{\text{in}}_{i-\frac{1}{2}} &:= \max(-\alpha_i, 0) \geq 0, \qquad & b^{\text{out}}_{i-\frac{1}{2}} &:= \min(-\alpha_i, 0) \leq 0, \\ b^{\text{in}}_{i+\frac{1}{2}} &:= \max(\alpha_i, 0) \geq 0, \qquad & b^{\text{out}}_{i+\frac{1}{2}} &:= \min(\alpha_i, 0) \leq 0. \end{aligned} \tag{25}$$

To clarify the notation: if the advection speed $-\alpha_i$ is positive, we have inflow into the finite volume $\mathcal{V}_i$ at the point $\mathbf{x}_{i-\frac{1}{2}}$ and outflow at $\mathbf{x}_{i+\frac{1}{2}}$, if the advection speed $-\alpha_i$ is negative, there is outflow at $\mathbf{x}_{i-\frac{1}{2}}$ and inflow at $\mathbf{x}_{i+\frac{1}{2}}$.

As we shall see, if the sign of α_i^n corresponds to the inflow (either at the point $\mathbf{x}_{i-\frac{1}{2}}$ or $\mathbf{x}_{i+\frac{1}{2}}$), the corresponding terms have a suitable sign and allow cancellation of absolute values in the diagonal dominance criterion. Therefore, to perform the time discretisation of the tangential term (24), we take the Inflow terms Implicitly and Outflow terms Explicitly (IIOE):

$$\frac{b^{\text{in},n}_{i-\frac{1}{2}}}{2}\left(\mathbf{x}_i^{n+1} - \mathbf{x}_{i-1}^{n+1}\right) + \frac{b^{\text{out},n}_{i-\frac{1}{2}}}{2}\left(\mathbf{x}_i^{n} - \mathbf{x}_{i-1}^{n}\right) + \frac{b^{\text{in},n}_{i+\frac{1}{2}}}{2}\left(\mathbf{x}_i^{n+1} - \mathbf{x}_{i+1}^{n+1}\right) + \frac{b^{\text{out},n}_{i+\frac{1}{2}}}{2}\left(\mathbf{x}_i^{n} - \mathbf{x}_{i+1}^{n}\right).$$

The final system has the form (19), i.e.

$$-A_i^n\mathbf{x}_{i-1}^{n+1} + B_i^n\mathbf{x}_i^{n+1} - C_i^n\mathbf{x}_{i+1}^{n+1} = \mathbf{D}_i^n, \qquad i = 1, \ldots, m \tag{26}$$

for $n = 0, \dots, n_{\max} - 1$. The resulting coefficients A_i^n, B_i^n, C_i^n, and $\mathbf{D}_i^n$ are

$$
\begin{aligned}
A_i^n &:= \frac{\delta_i^n}{h_{i-1}^n} + \frac{b_{i-\frac{1}{2}}^{\mathrm{in},n}}{2}, \quad B_i^n := \frac{h_{i-1}^n + h_i^n}{2\tau^n} + \delta_i^n \left(\frac{1}{h_i^n} + \frac{1}{h_{i-1}^n} \right) + \frac{b_{i-\frac{1}{2}}^{\mathrm{in},n}}{2} + \frac{b_{i+\frac{1}{2}}^{\mathrm{in},n}}{2}, \\
C_i^n &:= \frac{\delta_i^n}{h_i^n} + \frac{b_{i+\frac{1}{2}}^{\mathrm{in},n}}{2}, \qquad (27) \\
\mathbf{D}_i^n &:= \frac{h_{i-1}^n + h_i^n}{2\tau^n} \mathbf{x}_i^n + \beta_{e,i}^n \left(\frac{\mathbf{x}_{i+1}^n - \mathbf{x}_{i+1}^n}{2} \right)^{\perp} + \frac{1}{2} b_{i-\frac{1}{2}}^{\mathrm{out},n} (\mathbf{x}_i^n - \mathbf{x}_{i-1}^n) + \frac{1}{2} b_{i+\frac{1}{2}}^{\mathrm{out},n} (\mathbf{x}_i^n - \mathbf{x}_{i+1}^n).
\end{aligned}
$$

The diagonal dominance criterion $|B_i^n| \geq |A_i^n| + |C_i^n|$, now reads

$$
\left| \frac{h_{i-1}^n + h_i^n}{2\tau^n} + \delta_i^n \left(\frac{1}{h_i^n} + \frac{1}{h_{i-1}^n} \right) + \frac{b_{i-\frac{1}{2}}^{\mathrm{in},n}}{2} + \frac{b_{i+\frac{1}{2}}^{\mathrm{in},n}}{2} \right| \geq \left| \frac{\delta_i^n}{h_{i-1}^n} + \frac{b_{i-\frac{1}{2}}^{\mathrm{in},n}}{2} \right| + \left| \frac{\delta_i^n}{h_i^n} + \frac{b_{i+\frac{1}{2}}^{\mathrm{in},n}}{2} \right|.
$$

All terms are nonnegative, and therefore, we can cancel all absolute values and the condition reduces to $\frac{h_{i-1}^n + h_i^n}{2\tau^n} \geq 0$. This holds for any τ^n and the solution of the system (26) by the modified Thomas algorithm is guaranteed.

Let us briefly discuss the stopping criterion for evolution. If the segmentation is finished and the curve does not evolve, we should break the time loop even before the final $n_{\max}$-th time step is computed by (26). If a large percentage (e.g. 99 %) of the points do not move, we switch the evolution phase from expansion to attraction (setting $\lambda^n = 1$) and perform 20 evolution steps in this phase. By not moving points, we naturally mean points which move very slowly (e.g. $\|\mathbf{x}_i^{n+1} - \mathbf{x}_i^n\| < 0.05h$) but also the points jumping back and forth between two neighbouring pixels, which happens very often at an edge.

3.4 Discretisation of the normal and tangential speed

The normal speed is given by (8) and we approximate it at the point $\mathbf{x}_i^n$ as follows:

$$
\beta_i^n = \beta_{e,i}^n + \delta_i^n k_i^n, \qquad \text{with} \qquad \beta_{e,i}^n = \left(1 - \lambda^n\right) B_i^n - \lambda^n \nabla E(\mathbf{x}_i^n) \cdot \mathbf{N}_i^n, \tag{28}
$$

The evaluation of δ_i^n by (13) is straightforward. The discretisation of the signed curvature k_i^n will be discussed in Sect. 3.5. The blowing function is calculated as $B_i^n = H_i^n E(\mathbf{x}_i^n)$, with the difference d_i^n in the calculation of H_i^n (using (6)) approximated as $d_i^n = \left| \mathcal{I}(\mathbf{x}_i^n) - \frac{1}{|V^n|} \sum_{p \in V^n} \mathcal{I}_p \right|$ where V^n is the set of pixels p that approximates the region $V(t^n)$, $|V^n|$ denotes the number of pixels in V^n and $\mathcal{I}_p$ denotes the value of image $\mathcal{I}$ in the pixel p. The edge detector function E is computed using (3), (4) in the bitmap image $\mathcal{I}$ and the gradients $\nabla \mathcal{I}$ and ∇E are approximated by central differences. The positively oriented normal in (28) is calculated as $\mathbf{N}_i^n = \left(\frac{\mathbf{x}_{i+1}^n - \mathbf{x}_{i-1}^n}{\|\mathbf{x}_{i+1}^n - \mathbf{x}_{i-1}^n\|} \right)^{\perp}$.

The tangential speed is the solution of the equation (12) and we discretize it as follows. The derivative on the LHS is approximated by the finite difference, $\partial_s \alpha \approx \frac{\alpha_{i+1}^n - \alpha_i^n}{h_i^n}$. We regard it as a central difference at the point $\mathbf{x}_{i+\frac{1}{2}}^n$ and, therefore, approximate the RHS at that point. The average (integral) term is approximated by the Riemannian sum, $\langle k\beta \rangle (t^n) \approx \frac{1}{L^n} \sum_{j=1}^{m} k_{j+\frac{1}{2}}^n \beta_{j+\frac{1}{2}}^n h_j^n$ with approximate curve length computed as $L^n = \sum_{i=1}^{m} h_i^n$ and the normal speed at $\mathbf{x}_{i+\frac{1}{2}}^n$ evaluated as $\beta_{i+\frac{1}{2}}^n = \frac{1}{2}(\beta_i^n + \beta_{i+1}^n)$. The approximation of the signed curvature $k^n_{i+\frac{1}{2}}$ will be explained in the next section. The norm $g = \|\partial_u \mathbf{x}\|$ is approximated by $\frac{\|\mathbf{x}^n_{i+1}-\mathbf{x}^n_i\|}{\Delta u} = h^n_i m$. Putting all together, we have

$$\frac{\alpha^n_{i+1} - \alpha^n_i}{h^n_i} = k^n_{i+\frac{1}{2}}\beta^n_{i+\frac{1}{2}} - \frac{1}{L^n}\sum_{j=1}^{m} k^n_{j+\frac{1}{2}}\beta^n_{j+\frac{1}{2}} h^n_j + \left(\frac{L^n/m}{h^n_i} - 1\right)\omega, \tag{29}$$

for $i = 1, \ldots, m$, which is a two-diagonal system for unknown $\alpha^n_i, i = 1, \ldots, m$. The system can be easily solved by setting $\alpha^n_1 = 0$ and successively computing $\alpha^n_2, \ldots, \alpha^n_m$ by

$$\alpha^n_{i+1} = \alpha^n_i + h^n_i\left(k^n_{i+\frac{1}{2}}\beta^n_{i+\frac{1}{2}} - \frac{1}{L^n}\sum_{j=1}^{m} k^n_{j+\frac{1}{2}}\beta^n_{j+\frac{1}{2}} h^n_j + \left(\frac{L^n/m}{h^n_i} - 1\right)\omega\right). \tag{30}$$

Finally, we redefine the tangential speeds by subtracting their average $\alpha^n_i \leftarrow \alpha^n_i - \frac{1}{m}\sum_{j=1}^{m}\alpha^n_j$ to ensure zero average tangential motion.

3.5 Discretisation of the signed curvature

In the discretisation of the normal and tangential speeds, we need a formula for approximate curvature at the grid points $\mathbf{x}^n_i$ and at the midpoints $\mathbf{x}^n_{i+\frac{1}{2}}$ of the edges. We will discretize the formula

$$k = \frac{\partial^2_u \mathbf{x} \cdot \mathbf{N}}{\|\partial_u \mathbf{x}\|^2}. \tag{31}$$

It can be derived directly from the definition. Using the notation $g = \|\partial_u \mathbf{x}\|$, we have

$$\begin{aligned} k = \partial_s \mathbf{T} \cdot \mathbf{N} &\overset{1.}{=} \frac{1}{g}\partial_u \frac{\partial_u \mathbf{x}}{g} \cdot \mathbf{N} \overset{2.}{=} \frac{1}{g}\frac{(\partial^2_u \mathbf{x})g - \partial_u \mathbf{x}\left(\frac{\partial_u \mathbf{x}}{g}\cdot \partial^2_u \mathbf{x}\right)}{g^2}\cdot \mathbf{N} \\ &= \frac{\partial^2_u \mathbf{x} - (\partial^2_u \mathbf{x}\cdot \mathbf{T})\mathbf{T}}{g^2}\cdot \mathbf{N} \overset{3.}{=} \frac{\partial^2_u \mathbf{x}\cdot \mathbf{N}}{g^2}, \end{aligned}$$

where we used:

1. Definition of the unit tangent vector $\mathbf{T} = \frac{\partial_u \mathbf{x}}{g}$ and the chain rule $\partial_s = \frac{1}{g}\partial_u$ (see 2. under (10)).
2. Quotient rule and $\partial_u\|\mathbf{w}\| = \partial_u\sqrt{\mathbf{w}\cdot\mathbf{w}} = \frac{\mathbf{w}}{\|\mathbf{w}\|}\cdot\partial_u \mathbf{w}$ with $\mathbf{w} = \partial_u \mathbf{x}$.
3. Since the basis $\{\mathbf{T}, \mathbf{N}\}$ is orthonormal, we have $\mathbf{w} - (\mathbf{w}\cdot\mathbf{T})\mathbf{T} = (\mathbf{w}\cdot\mathbf{N})\mathbf{N}$, where $\mathbf{w} = \partial^2_u \mathbf{x}$.

To approximate the signed curvature at the grid point $\mathbf{x}^n_i$, we can use the central differences $\partial_u \mathbf{x} \approx \frac{\mathbf{x}^n_{i+1}-\mathbf{x}^n_{i-1}}{2\Delta u}$ and $\partial^2_u \mathbf{x} \approx \frac{\mathbf{x}^n_{i-1}-2\mathbf{x}^n_i+\mathbf{x}^n_{i+1}}{\Delta u^2}$ (see Fig. 11) which leads to the discretisation

$$k^n_i = \frac{4(\mathbf{x}^n_{i-1} - 2\mathbf{x}^n_i + \mathbf{x}^n_{i+1})\cdot \mathbf{N}^n_i}{\|\mathbf{x}^n_{i+1} - \mathbf{x}^n_{i-1}\|^2}. \tag{32}$$

At the midpoint $\mathbf{x}^n_{i+\frac{1}{2}}$ of the edge, we use the central differences $\partial_u \mathbf{x} \approx \frac{\mathbf{x}^n_{i+1}-\mathbf{x}^n_i}{\Delta u}$, $\partial^2_u \mathbf{x} \approx \frac{\mathbf{x}_{i+\frac{3}{2}}-2\mathbf{x}_{i+\frac{1}{2}}+\mathbf{x}_{i-\frac{1}{2}}}{\Delta u^2}$ (see Fig. 11) and the normal $\mathbf{N}^n_{i+\frac{1}{2}} = \left(\frac{\mathbf{x}^n_{i+1}-\mathbf{x}^n_i}{h^n_i}\right)^{\perp}$, which gives

$$k^n_{i+\frac{1}{2}} = \frac{\mathbf{x}^n_{i-1} - \mathbf{x}^n_i - \mathbf{x}^n_{i+1} + \mathbf{x}^n_{i+2}}{2(h^n_i)^3}\cdot(\mathbf{x}^n_{i+1} - \mathbf{x}^n_i)^{\perp}. \tag{33}$$

3.6 Topological changes

When dealing with curve evolution using the Lagrangian approach, one has to treat topological changes that occur during the evolution. The evolving curve can split into multiple curves, or several initial evolving curves can merge. The detection and treatment of the topological changes can be highly time-consuming because the natural approach has computational complexity $O(m^2)$, where m denotes the number of grid points (it includes computation of pairwise distances between all grid points). However, we adopt the method from Mikula and Urbán (2012); Balažovjech et al. (2012); Ambroz et al. (2019), which has complexity only $O(m)$. This makes the Lagrangian method for curve evolution efficient and applicable in complex situations.

The basic idea of Mikula and Urbán (2012); Balažovjech et al. (2012); Ambroz et al. (2019) is to construct a background mesh (we use the mesh described in Sect. 2.1). To detect splitting, we walk through all curve grid points and label the underlying cells by the grid point numbers. If the current i-th point belongs to a cell already labelled by the j-th point and the difference $i - j$ is large enough (e.g. $i - j > 3$), the splitting is detected. Since we pass the curve only once, the computational complexity is $O(m)$. The merging detection is similar, we walk through the points of all curves and label underlying cells by the curve number. If the current point belongs to the cell already labelled by a point of another curve, the merging is detected. The most complete explanation of the algorithms (including pseudocodes) is given in Ambroz et al. (2019).

For the reliable topological changes detection and treatment, it is required to maintain the mean curve segment length close to a specific constant h_d called desired curve segment length (desired distance between neighbouring grid points). We use setting $h_d = h/2$, where h is the pixel size. In our method, every curve is asymptotically uniformly discretised (see Sect. 2.5), i.e. all segment lengths are close to their mean value. However, the mean value can differ between different curves, and it increases for expanding curves and decreases for shrinking ones. Such difference is undesirable, especially in merging of curves, so we maintain the mean curve segment length close to h_d for all curves by adding or removing points, after which the asymptotically uniform redistribution is quickly obtained due to the tangential velocity α. The detailed algorithm for adding and removing grid points is in Ambroz et al. (2019). We note that small curves with less than $m = 15$ grid points are deleted.

The merging detection in this paper is slightly different. If two evolving curves numbered c_1 and c_2 get close to each other, we have to decide if they are segmenting regions with similar properties (curves should merge) or different properties (curves should not merge). First, let us discuss the single-channel case. We compute the average values $\langle \mathcal{I} \rangle_{c_1}^t$ and $\langle \mathcal{I} \rangle_{c_2}^t$ of image $\mathcal{I}$ in the regions $V_{c_1}(t)$, $V_{c_2}(t)$ already segmented by the curves c_1 and c_2, respectively, see definitions in the Sect. 2.4. If the difference $d_{c_1,c_2}(t) := |\langle \mathcal{I} \rangle_{c_1}^t - \langle \mathcal{I} \rangle_{c_2}^t|$ is smaller than the threshold value d_0, merging is detected.

For the correct topological changes detection, we use the time step length $\tau^n < h/\beta_{e,\max}^n$, where $\beta_{e,\max}^n := \max_{i \in \{1,\ldots,m\}} \beta_{e,i}^n$ denotes the maximal external normal speed in the n-th time step. Small time step length is crucial because it prevents the points of the curve from skipping a pixel one time step (and missing possible topological change).

In the case of multichannel segmentation, we compute the difference $d_{c_1,c_2}^l(t)$ for each channel $\mathcal{I}^l, l = 1, \ldots, 5$ and compare it with the corresponding threshold d_0^l. If $d_{c_1,c_2}^l(t) < d_0^l$ for all l, then the curves c_1 and c_2 segment the regions $V_{c_1}(t)$, $V_{c_2}(t)$ with similar properties (in all channels) and the merging is detected. If $d_{c_1,c_2}^l(t) \geq d_0^l$ for some l, merging is forbidden.

3.7 Summary of the entire algorithm

Now, we give a summary of the algorithm for the multichannel curve segmentation of the planar point clouds. The algorithm consists of the following main steps and is fully automatic.

1. Preprocessing:
   (a) Computation of the images $\mathcal{I}^l$ (for all channels) representing the input planar point cloud according to Sect. 2.1.
   (b) Computation of the edge detector functions g^l and smoothed aggregated edge detector function E as described in the Sects. 2.3 and 2.7.
   (c) Construction of the initialisation mask M as explained in the Sect. 2.8 and insertion of the initial curve(s).
2. Evolution: The main evolution cycle through $n = 1, \dots, n_{\max}$.
   (a) Detection and treatment of the topological changes according to the Sect. 3.6 and the paper (Ambroz et al. 2019).
      (i) Splitting (for each curve).
      (ii) Deleting small curves.
      (iii) Merging.
   (b) Adding and removing the grid points on all curves as described in Ambroz et al. (2019). (This is performed only in the expansion phase and only in every fifth step.)
   (c) For all (unstopped) curves:
      (i) Calculation of the necessary geometric quantities as described in the Sect. 3, computation of the blowing function B, external normal speed β_e and tangential speed α, curvature regularisation weights δ according to Sect. 3.4.
      (ii) Construction of the system matrix coefficients (27) and solution of the system (26).
      (iii) Check the stopping criterion discussed at the end of Sect. 3.3 and possibly switch to the attraction phase ($\lambda = 1$) or stop the curve.
   (d) Check if all curves are stopped. If so, update the initialisation mask M and check if the segmentation target is reached (see Sect. 2.8). If not, add a new curve(s) and continue in the segmentation process.
3. Postprocessing: Create the final segments of the point cloud as explained in Sect. 2.9.

4 Numerical experiments

In this section, we test our curve segmentation methodology in several numerical experiments. Data acquisition was carried out with the Trimble TX5 scanner with a priori distance measurement error ± 2 mm at 10 m (defined by the producer). In all experiments, we set the desired curve segment length to $h/2$ (see Sect. 3.6), where h is the pixel size. We use adaptive time step length. For each evolving curve, we set $\tau^n = 0.3\, h/\beta^n_{e,\max}$ in the expansion phase ($\lambda^n < 1$) and smaller $\tau^n = 0.1\, h/\beta^n_{e,\max}$ in the attraction phase ($\lambda^n = 1$). The Gaussian filter variance was set to $\sigma = 0.5$. The global weight of the curvature regularisation was set to $\delta_k = 0.01$, the tangential velocity relaxation parameter to $\omega = 100$, the threshold $d_0 = 0.5$ for all channels and the initial weight between the attraction and the expansion term to $\lambda_0 = 0.1$.

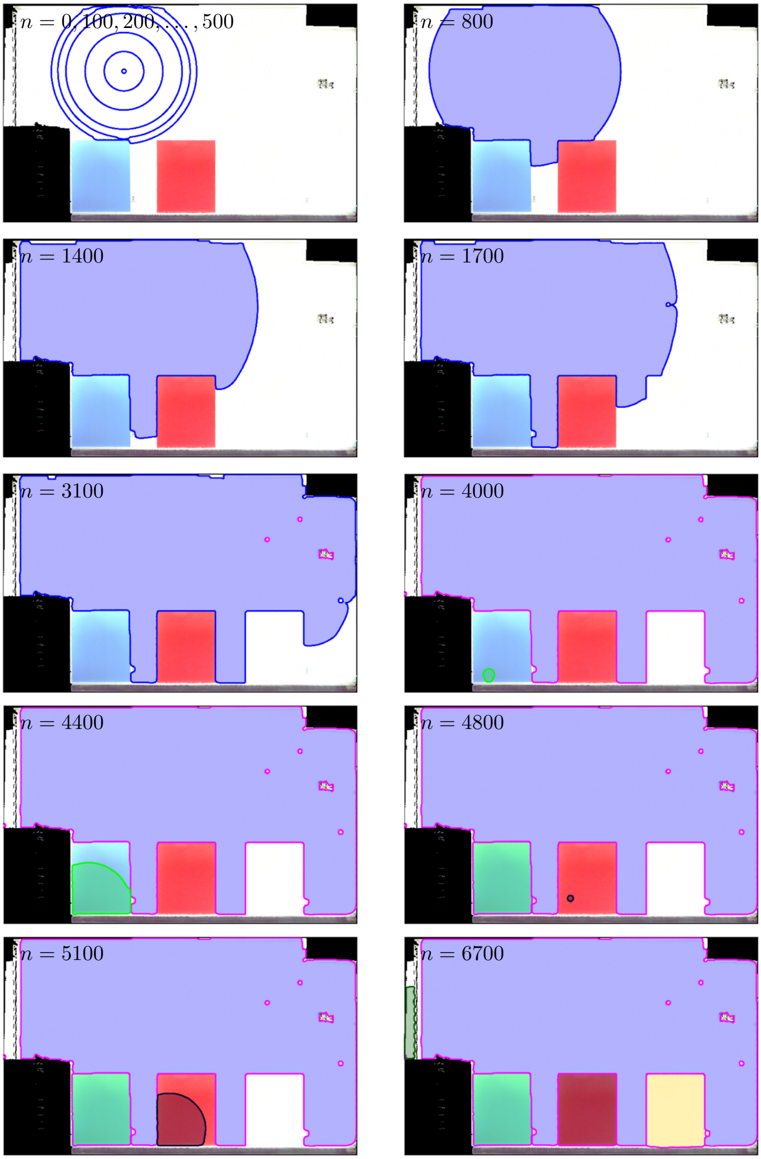


Fig. 12 Curve segmentation of the wall with three plastic plates, visualisation of the selected time steps

In the first experiment, we perform the curve segmentation of the planar point cloud corresponding to the white wall with three plastic plates depicted in Fig. 7. The point cloud consists of approximately 1.2 million points, its resolution is about 3 × 3 mm and the pixel size was set to $h = 12$ mm. The images and edge detectors for individual channels were discussed in the Sect. 2.7 (see Fig. 8). The smoothed aggregated edge detector is visualised in Fig. 9. Several steps of segmentation are shown in Fig. 12. The curves coloured in magenta are stopped. We start the segmentation with a single curve, and when all curves stop, we

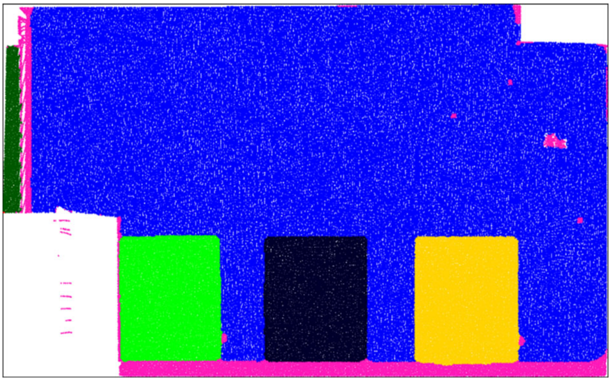
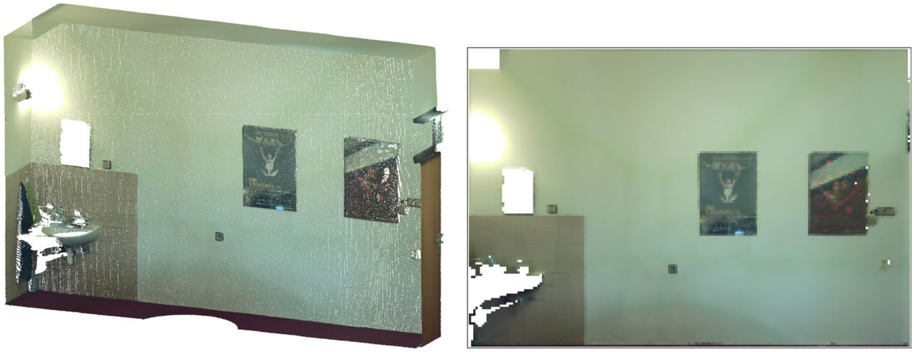


Fig. 13 The final segments of the planar point cloud

always insert only one new curve.[11] The evolution up to the time step $n = 500$ is shown in the upper left of Fig. 12. The curve meets the first edges around the time step $n = 400$ and the edge detector correctly prevents the curve from crossing the edge. In the next subfigures, we also colour already segmented region V^n. In the time step $n = 1700$, we encounter a surprising observation: the curve stops on the white plastic plate, which is not visible in the RGB image, but thanks to the intensity channel (fourth row of Fig. 8), it is clearly captured in the aggregated edge detector, Fig. 9. The right part of the curve is enclosing the electrical box cover and the curve splitting is going to occur. In the time step $n = 4000$, we see that the first segmentation curve correctly segmented the wall and excluded unwanted objects: the plates, but also the electrical elements (e.g. electrical outlets in the lower part of the wall). Another curve (green) was inserted onto the blue plate. It correctly segmented the plate and did not merge with the previous curve (see $n = 4400$). After it stopped, a new (black) curve was inserted onto the red plate. The last subfigure of Fig. 12 shows the segmentation just before the termination. After the dark green curve on the left stops, the segmentation target is reached and no more curves are inserted. The final segments of the planar point cloud are visualised in Fig. 13. The magenta points represent the waste after segmentation, i.e. the regions that were not segmented by any of the curves.

In the second experiment, we segment a wall from the interior of P.O. Hviezdoslav Theatre in Bratislava, Fig. 14. The planar point cloud has roughly 0.4 million points with an average resolution of about 6×6 mm. Pixel size was set to $h = 18$ mm. We can see two pictures on the wall, a sink with facing in the left corner. The hole above the facing corresponds to a mirror (from which the laser beam did not reflect directly to the scanner). The representative images $\mathcal{I}^l$ and the corresponding edge detectors g^l are shown in Fig. 15. We see the edges of the pictures in all channels. Some edges of these pictures are not complete in each channel, however, in the smoothed aggregated edge detector, Fig. 16, we see distinct and complete edges (mainly due to the intensity channel). The edge around the facing is mild in all edge detectors, but since the edge is present in multiple edge detectors and the aggregation is done by multiplication (14), the aggregated edge is strong in Fig. 16, lower left part. An interesting situation is in the top left area of the wall strongly illuminated by the lamp. There is a mild edge in all colour channels, but not precisely in the same region. As a result, the aggregated edge in Fig. 16 is mild and diffuse.

[11] This is just to keep the experiment simple. The segmentation would be much faster if we would start with multiple curves and insert multiple curves during the segmentation.



(a) The point cloud. (b) Corresponding bitmap image.

Fig. 14 A wall in P.O. Hviezdoslav Theatre in Bratislava

Several steps of the segmentation process are shown in Fig. 17. In this experiment, we start with four random initial curves. In the time steps $n = 300$ and $n = 352$, we see that three of them merged into one. However, as we see in the time step $n = 600$, the green and blue curves did not merge, which is correct. The evolution stopped in the time step $n = 924$. There are two main segments (the first corresponding to the wall and the second to the facing) and the waste after segmentation, see Fig. 18. The curve segmentation correctly excluded pictures, electrical outlets, and switch (above the facing). Interestingly, the blue curve did not reach the facing edge in the area to the left of the mirror. The reason is that the wall is crooked (not flat) in this region, which is slightly captured by the edge detector corresponding to the D channel (last row of Fig. 15). The resulting edge is mild, so this is not the reason for stopping the curve. The curve stopped because the homogeneity function (6) is zero in this region due to the large $\mathrm{D} = |x_3|$ compared to the average in the already segmented region V^n. The detail of this region is shown in Fig. 19.

The last experiment is the segmentation of computer classroom B306 at the Faculty of Civil Engineering of the Slovak University of Technology in Bratislava. The input point cloud with approximately 10 million points and an average resolution of about 2×2 mm is in Fig. 20. The ceiling and one wall (with windows) were removed from the original data in order to make the images more readable. The point cloud was scanned from four scanner positions. First, we performed primary segmentation to find planar point clouds. Secondary segmentation was performed only for vertical planes corresponding to the largest walls visualised in Fig. 21, left column: front wall with two whiteboards, the right one is partially covered by a projection screen; Side wall consisting of two planes, the first with a door and the second with a long wall hanger and alcove with a sink; and finally, the back wall. We also added a normal filter to remove points whose local normals deviate from the normal of the fitting plane by more than 30 degrees (coloured blue). The red points form the resulting planar point clouds. In the right column of Fig. 21, we can see corresponding smoothed edge detector functions. Note that some unwanted ‘artificial’ edges are also present, e.g. long inclined edge on the back wall caused by uneven illumination by natural light, edges on the whiteboard of the same origin, or edges due to merging of multiple scans (mainly on the front wall and bottom of the back wall). The final segments are shown in Fig. 22. Dark grey points represent waste from the primary segmentation (points that do not belong to any of the segmented planes). Light grey

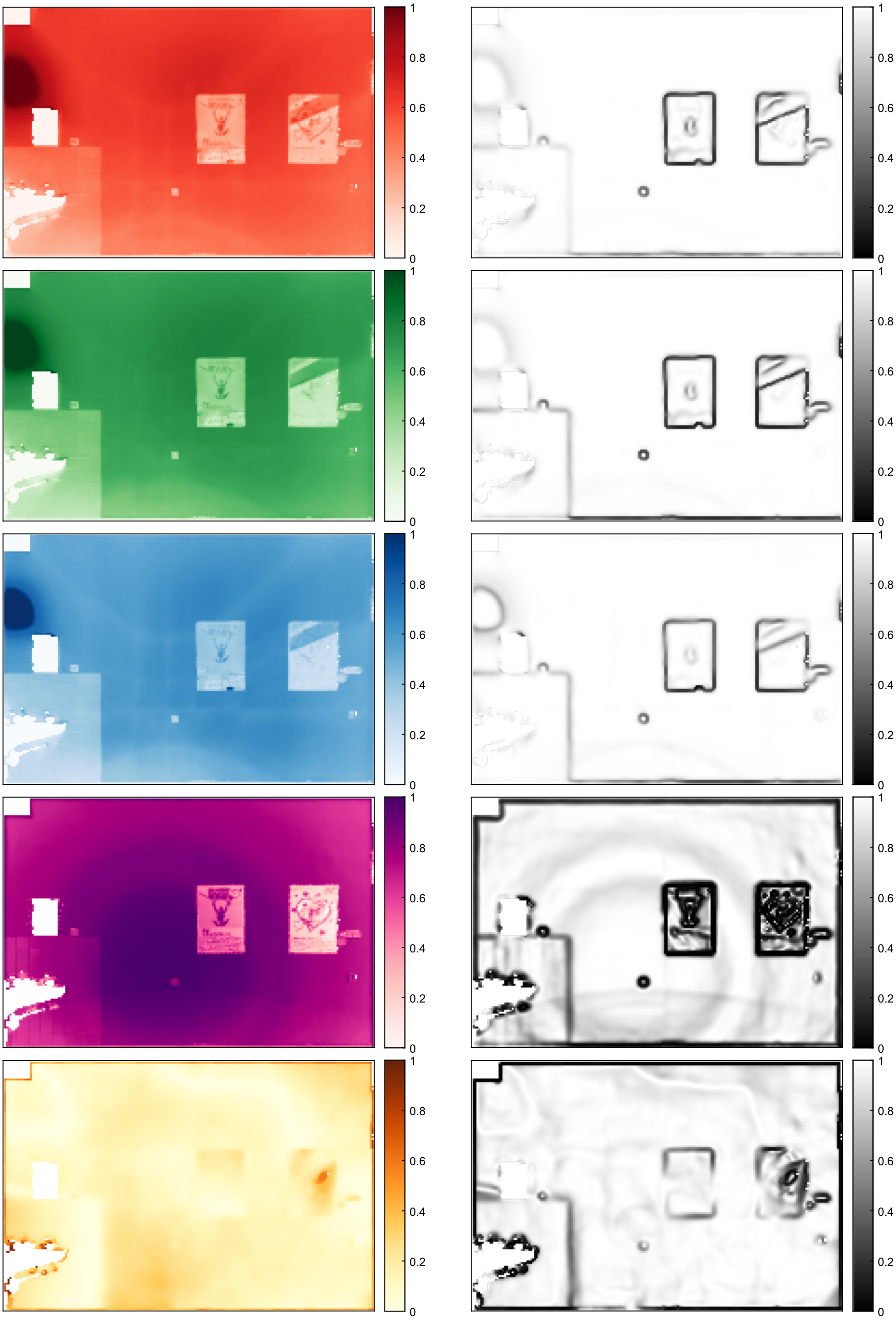


Fig. 15 Left column: Bitmap images generated from the channels R, G, B, I, D of the point cloud. Right column: Corresponding edge detector functions

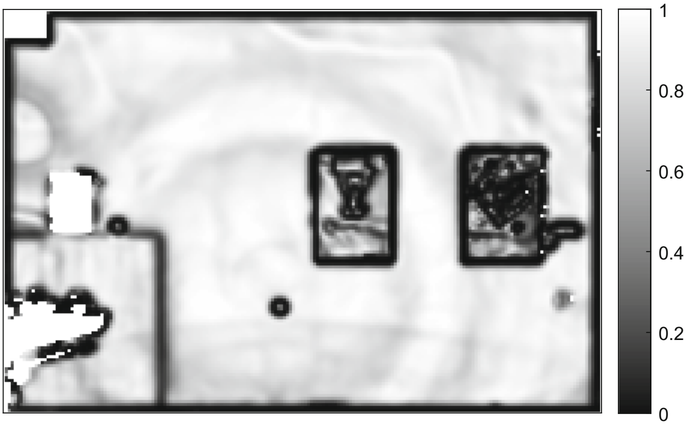


Fig. 16 The smoothed aggregated edge detector function $E(\mathbf{x})$

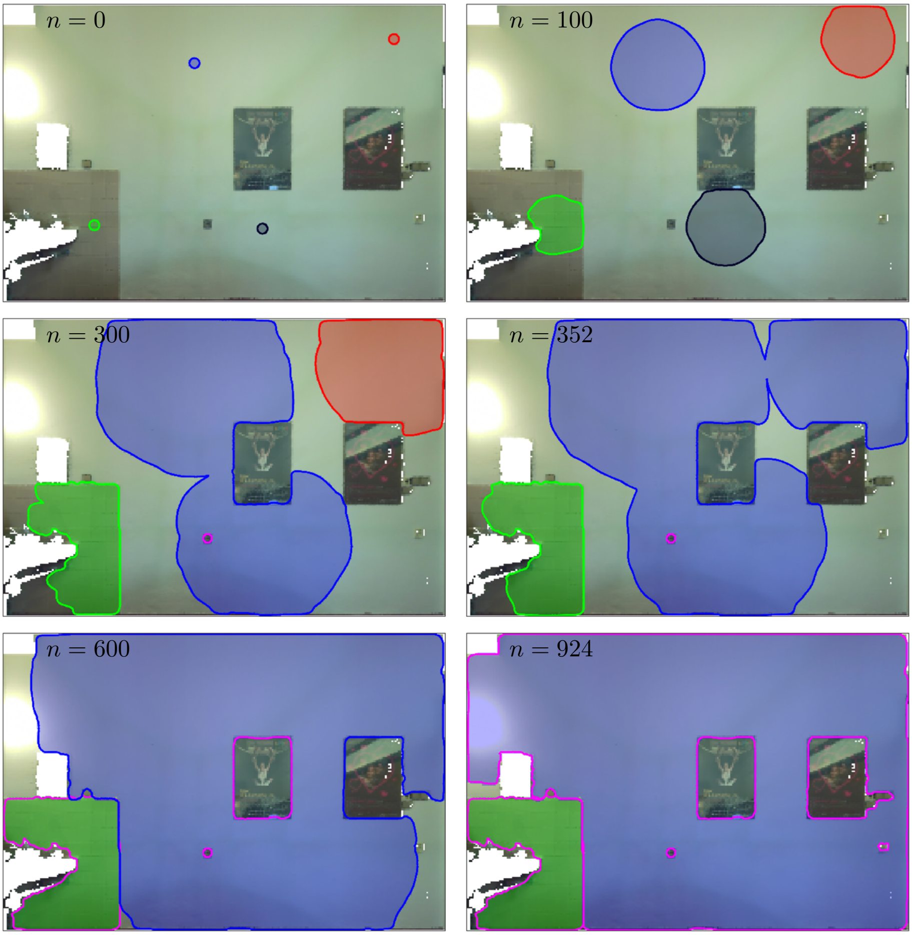


Fig. 17 Curve segmentation of the wall in P.O. Hviezdoslav Theatre in Bratislava

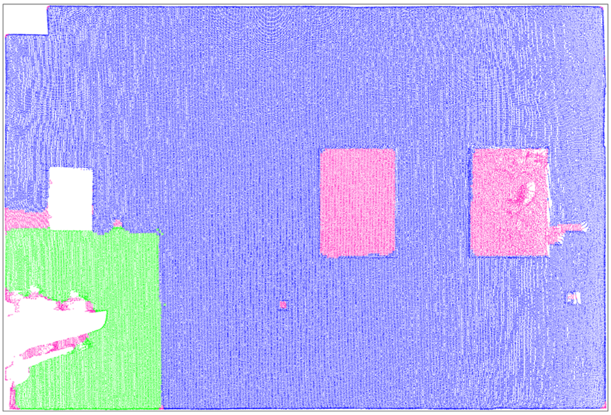


Fig. 18 The final segments of the planar point cloud

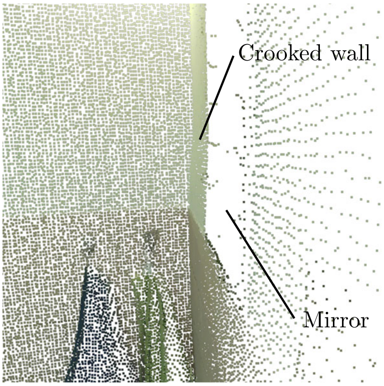


Fig. 19 Detail of crooked wall to the left of the mirror

points form the waste of the secondary segmentation.[12] We can see that on the front wall, our curve segmentation successfully found segments representing the wall, whiteboards, and projection screen, removed unwanted objects, such as electrical elements, wall hanger on the side wall, divided the side wall into top and bottom segments (corresponding to different wall colours). In the upper part of the door plane, we see separate segments due to a cable cover (which is barely visible in RGB channels, Fig. 20, but considerably filtered out by the normal filter, see the second row of Fig. 21). Some regions are not segmented as desired, e.g. the segment representing the darker part of the wall on the left from the door, Fig. 20, is merged with the upper segment, see the dark red segment in Fig. 22. The segmentation curve leaked through the weak part of the edge. Also, the situation on the back wall is strongly influenced by the edge caused by light and the low density of the point cloud in the part closer to us, see lower part of Fig. 20 and last row of Fig. 21. However, these are situations that cannot be avoided when working with real data. Therefore, we conclude that we are generally satisfied with the result of the segmentation.

[12] In the case of non-vertical planes, e.g. floor and table tops plane, whole planar point clouds are light grey since the secondary segmentation was not performed.

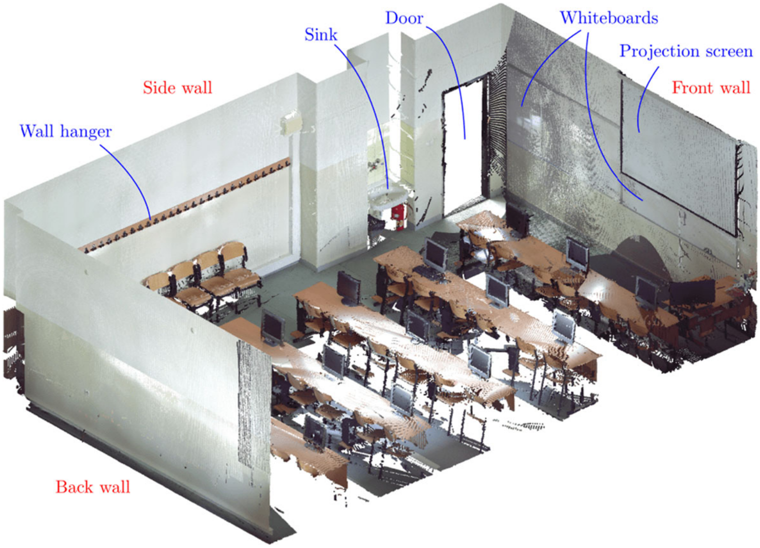


Fig. 20 Point cloud of the classroom B306 at the Faculty of Civil Engineering of the Slovak University of Technology in Bratislava (ceiling and one wall removed)

5 Conclusion

In this paper, we presented a method for multichannel segmentation of planar point clouds using evolving curves. The planar point clouds were represented by a set of images, one for each available channel (colour and intensity channels and distance from the fitting plane). We discussed details of the mathematical model of the Lagrangian curve evolution, construction of the edge detectors using all channels, the design of the normal velocity driving the evolution. The model was equipped with a suitable tangential motion to improve the quality of the mesh. We derived the numerical discretisation of the mathematical model and used the fast $O(n)$ approach to treat topological changes. The method was successfully tested on multiple examples of real data.

The method could be used to automate the processing of point clouds, especially in the area of as-built documentation and geometry verification of building structures. Further development of the method can include automatic tuning of the model parameters; correction of the rough TLS intensity channel (Höfle and Pfeifer 2007; Xu et al. 2017; Bolkas 2019); usage of other channels, e.g. a derived channel capturing point cloud curvature; examination of other possibilities for aggregation of the edge detector functions (e.g. choose different fuzzy logic aggregation operators); or automatic computation of threshold values d_0^l, possibly using information from histogram of the segmented image. The overall segmentation methodology could incorporate the segmentation of other geometric shapes, such as cylinders or rounded walls.

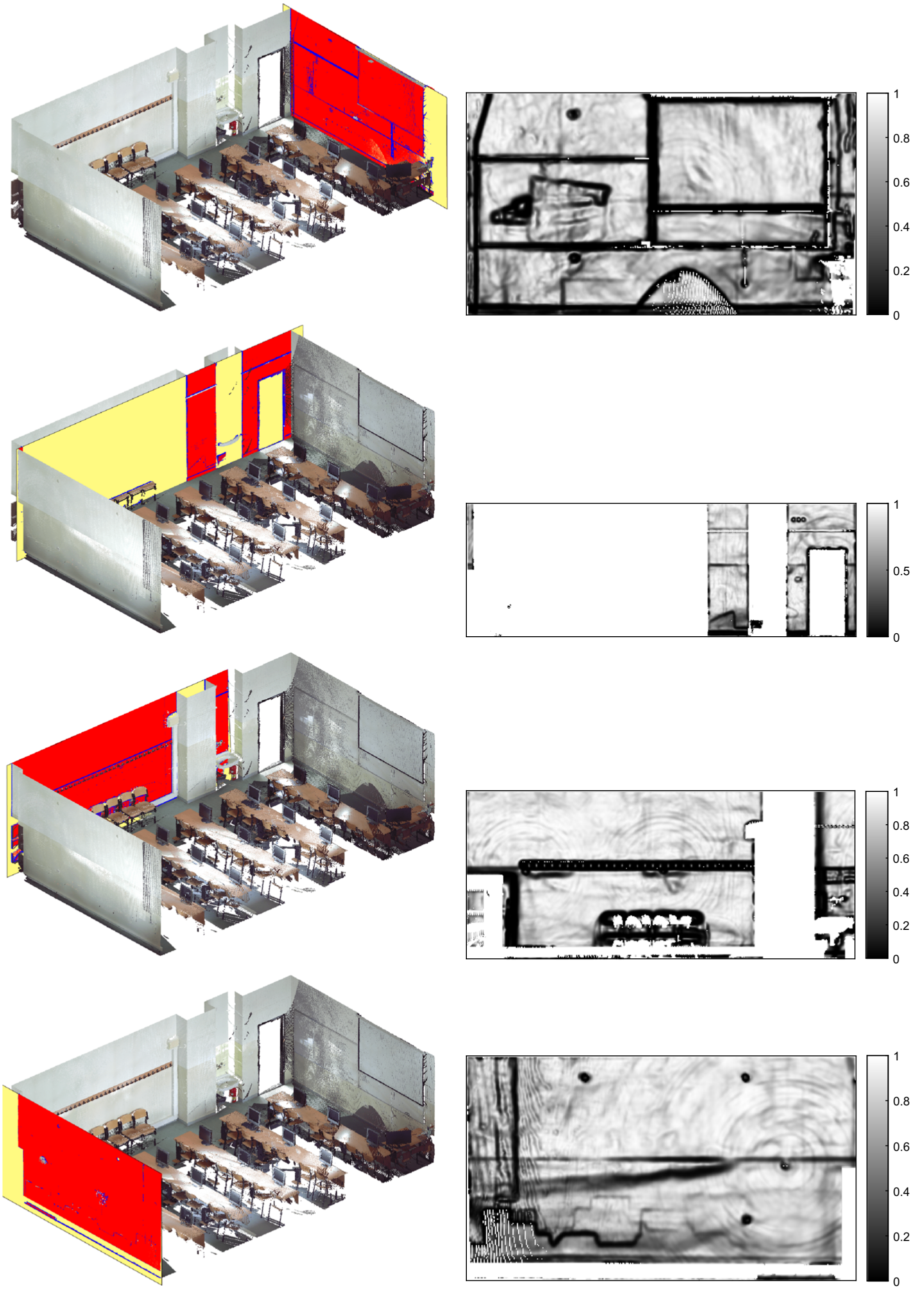


Fig. 21 Left column: primary segmentation of the point cloud. Red points belong to the resulting planar point clouds and the blue points were rejected by the normal filter. Right column: corresponding smoothed edge detector functions

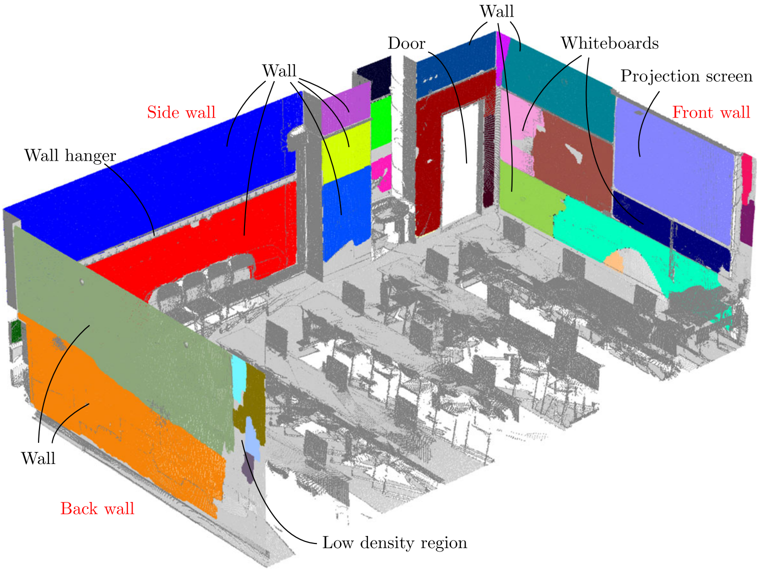


Fig. 22 Final segments of the point cloud after segmentation of the largest walls

Acknowledgements This work was supported by projects APVV-18-0247 and APVV-19-0460.

Funding Open access funding provided by The Ministry of Education, Science, Research and Sport of the Slovak Republic in cooperation with Centre for Scientific and Technical Information of the Slovak Republic

Data availability The data are available on request.

Declarations

Conflict of interest The authors declare that there is no conflict of interests regarding the publication of this article.

References

Adamopoulos E, Rinaudo F (2021) Close-range sensing and data fusion for built heritage inspection and monitoring-a review. Remote Sens. https://doi.org/10.3390/rs13193936

Ambroz, M., Balažovjech, M., Medl'a, M., Mikula, K. (2019) Numerical modeling of wildland surface fire propagation by evolving surface curves. Advances in Computational Mathematics 45

Balažovjech, M., Mikula, K., Šibíková, M., Urbán, J. (2012) Lagrangean method with topological changes for numerical modelling of forest fire propagation. Proceedings of ALGORITMY 2012, 19th Conference on Scientific Computing, 42–52

Balažovjech M, Mikula K (2011) A higher order scheme for a tangentially stabilized plane curve shortening flow with a driving force. SIAM Journal on Scientific Computing 33(5):2277–2294. https://doi.org/10.1137/100795309

Bariczová, G., Erdélyi, J., Honti, R., Tomek, L. (2021) Wall structure geometry verification using TLS data and BIM model. Applied Sciences 11(24). https://doi.org/10.3390/app112411804

Barrett J, Garcke H, Nürnberg R (2011) The approximation of planar curve evolutions by stable fully implicit finite element schemes that equidistribute. Numerical Methods for Partial Differential Equations 27:1–30

Benninghoff H, Garcke H (2014) Efficient image segmentation and restoration using parametric curve evolution with junctions and topology changes. SIAM Journal on Imaging Sciences 7(3):1451–1483. https://doi.org/10.1137/130932430

Bolkas D (2019) Terrestrial laser scanner intensity correction for the incidence angle effect on surfaces with different colours and sheens. International Journal of Remote Sensing 40(18):7169–7189. https://doi.org/10.1080/01431161.2019.1601283

Caselles V, Kimmel R, Sapiro G (1997) Geodesic active contours. International Journal of Computer Vision 22:61–79. https://doi.org/10.1109/ICCV.1995.466871

Corsaro S, Mikula K, Sarti A, Sgallari F (2006) Semi-implicit covolume method in 3d image segmentation. SIAM Journal on Scientific Computing 28(6):2248–2265. https://doi.org/10.1137/060651203

Dziuk G (1999) Discrete anisotropic curve shortening flow. SIAM Journal on Numerical Analysis 36(6):1808–1830. https://doi.org/10.1137/S0036142998337533

Fischler MA, Bolles RC (1981) Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24(6):381–395. https://doi.org/10.1145/358669.358692

Frei, H., Garcke, H. (2016) Segmentation and restoration of images on surfaces by parametric active contours with topology changes. Journal of Mathematical Imaging and Vision 55. https://doi.org/10.1007/s10851-015-0616-6

Giussani A, Scaioni M (2012) Application of tls to support landslides study: survey planning, operational issues and data processing. Int Arch Photogramm Remote Sens Spat Inf Sci 36(2012)

Grilli, E., Menna, F., Remondino, F. (2017) A review of point clouds segmentation and classification algorithms. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W3, 339–344. https://doi.org/10.5194/isprs-archives-XLII-2-W3-339-2017

Haličková J, Mikula K (2016) Level set method for surface reconstruction and its application in surveying. J Surv Eng 142(3):04016007. https://doi.org/10.1061/(ASCE)SU.1943-5428.0000159

Höfle B, Pfeifer N (2007) Correction of laser scanning intensity data: Data and model-driven approaches. ISPRS Journal of Photogrammetry and Remote Sensing 62(6):415–433. https://doi.org/10.1016/j.isprsjprs.2007.05.008

Holst C, Nothnagel A, Blome M, Becker P, Eichborn M, Kuhlmann H (2015) Improved area-based deformation analysis of a radio telescope's main reflector based on terrestrial laser scanning. J Appl Geod 9(1):1–14. https://doi.org/10.1515/jag-2014-0018

Honti, R., Erdélyi, J., Kopáčik, A. (2022) Semi-automated segmentation of geometric shapes from point clouds. Remote Sensing 14(18). https://doi.org/10.3390/rs14184591

Hou TY, Lowengrub JS, Shelley MJ (1994) Removing the stiffness from interfacial flows with surface tension. Journal of Computational Physics 114(2):312–338. https://doi.org/10.1006/jcph.1994.1170

Kimura M (1997) Numerical analysis of moving boundary problems using the boundary tracking method. Japan Journal of Industrial and Applied Mathematics 14(3):373–398. https://doi.org/10.1007/BF03167390

Mikula, K., Urbán, J. (2012) New fast and stable Lagrangean method for image segmentation. In: 2012 5th International Congress on Image and Signal Processing, pp. 688–696. https://doi.org/10.1109/CISP.2012.6469852

Mikula K, Ohlberger M (2011) Inflow-implicit/outflow-explicit scheme for solving advection equations. In: Fořt J, Fürst J, Halama J, Herbin R, Hubert F (eds) Finite Volumes for Complex Applications VI Problems & Perspectives. Springer, Berlin, Heidelberg, pp 683–691

Mikula K, Ševčovič D (2004) A direct method for solving an anisotropic mean curvature flow of plane curves with an external force. Mathematical Methods in the Applied Sciences 27(13):1545–1565. https://doi.org/10.1002/mma.514

Mikula K, Ohlberger M, Urbán J (2014) Inflow-implicit/outflow-explicit finite volume methods for solving advection equations. Applied Numerical Mathematics 85:16–37. https://doi.org/10.1016/j.apnum.2014.06.002

Mikula K, Urbán J, Kollár M, Ambroz M, Jarolímek I, Šibík J, Šibíková M (2021) An automated segmentation of NATURA 2000 habitats from sentinel-2 optical data. Discrete and Continuous Dynamical Systems - S 14(3):1017–1032

Mikula K, Urbán J, Kollár M, Ambroz M, Jarolímek I, Šibík J, Šibíková M (2021) Semi-automatic segmentation of NATURA 2000 habitats in sentinel-2 satellite images by evolving open curves. Discrete and Continuous Dynamical Systems - S 14(3):1033–1046. https://doi.org/10.3934/dcdss.2020231

Osher, S., Fedkiw, R. (2004) The Level Set Methods and Dynamic Implicit Surfaces, vol. 57, p. 273. Springer, USA. https://doi.org/10.1115/1.1760520

Perona P, Malik J (1990) Scale-space and edge detection using anisotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence 12(7):629–639. https://doi.org/10.1109/34.56205

Press, W., Teukolsky, S., Vetterling, W., Flannery, B. (2007) Numerical Recipes 3rd Edition: The Art of Scientific Computing

Previtali M, Barazzetti L, Brumana R, Cuca B, Oreni D, Roncoroni F, Scaioni M (2014) Automatic façade modelling using point cloud data for energy-efficient retrofitting. Appl Geom 6:95–113. https://doi.org/10.1007/s12518-014-0129-9

Remondino F, Rizzi A (2010) Reality-based 3d documentation of natural and cultural heritage sites-techniques, problems, and examples. Appl Geom 2:85–100. https://doi.org/10.1007/s12518-010-0025-x

Sethian JA (1999) Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science, vol 3. Cambridge University Press, USA

Ševčovič D, Mikula K (2001) Evolution of plane curves driven by a nonlinear function of curvature and anisotropy. SIAM Journal on Applied Mathematics 61(5):1473–1501. https://doi.org/10.1137/S0036139999359288

Ševčovič D, Yazaki S (2011) Evolution of plane curves with a curvature adjusted tangential velocity. Japan Journal of Industrial and Applied Mathematics 28(3):413. https://doi.org/10.1007/s13160-011-0046-9

Torr PHS, Zisserman A (2000) Mlesac: A new robust estimator with application to estimating image geometry. Computer Vision and Image Understanding 78(1):138–156. https://doi.org/10.1006/cviu.1999.0832

van Krieken E, Acar E, van Harmelen F (2022) Analyzing differentiable fuzzy logic operators. Artificial Intelligence 302:103602. https://doi.org/10.1016/j.artint.2021.103602

Xu, T., Xu, L., Yang, B., Li, X., Yao, J. (2017) Terrestrial laser scanning intensity correction by piecewise fitting and overlap-driven adjustment. Remote Sensing 9(11). https://doi.org/10.3390/rs9111090

Zhao H-K, Osher S, Merriman B, Kang M (2000) Implicit and nonparametric shape reconstruction from unorganized data using a variational level set method. Computer Vision and Image Understanding 80(3):295–314. https://doi.org/10.1006/cviu.2000.0875

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.