DISCRETE AND CONTINUOUS
DYNAMICAL SYSTEMS SERIES S

doi:10.3934/dcdss.2020351

3D IMAGE SEGMENTATION SUPPORTED BY A POINT CLOUD

Balázs Kósa[1], Karol Mikula[1] and Markjoe Olunna Uba[1*]

[1]Department of Mathematics and Descriptive Geometry
Faculty of Civil Engineering, Slovak University of Technology
Radlinského 11, 810 05 Bratislava, Slovakia
*Department of Mathematics
University of Nigeria
410001 Nsukka, Nigeria

Antonia Weberling[1], Neophytos Christodoulou* and
Magdalena Zernicka-Goetz[1]

[1]Mammalian Embryo and Stem Cell Group
University of Cambridge, Department of Physiology, Development and Neuroscience
Downing Street, Cambridge CB2 3EG, UK
*Department of Biological Sciences
University of Cyprus
University Avenue 1, Nicosia 2109, Cyprus

Abstract. Here, we report a novel method of 3D image segmentation, using surface reconstruction from 3D point cloud data and 3D digital image information. For this task, we apply a mathematical model and numerical method based on the level set algorithm. This method solves surface reconstruction by the application of advection equation with a curvature term, which gives the evolution of an initial condition to the final state. This is done by defining the advective velocity in the level set equation as the weighted sum of distance function and edge detector function gradients. The distance function to the shape, represented by the point cloud, is computed using the fast sweeping method. The edge detector function is applied to the presmoothed 3D image. A crucial point for efficiency is the construction of an initial condition by a simple tagging algorithm, which allows us also to highly speed up the numerical scheme when solving PDEs. For the numerical discretization, we use a semi-implicit co-volume scheme in the curvature part and implicit upwind scheme in the advective part. The method was tested on representative examples and applied to real data representing 3D biological microscopic images of developing mammalian embryo.

1. **Introduction.** Image segmentation is widely used to divide images into fractions of units with related values to allow easier representation, edge detection or to subtract a part of the image. Such segments contain data, which are related and have similar traits. An image simplified in this manner is easier to analyze and the data it represents is clear to understand. One of the methods used for image segmentation is the subjective surface method [13, 2, 3, 6]. However, the application of

2010 *Mathematics Subject Classification.* Primary: 65M06, 65Y20, 53A05, 68U10; Secondary: 65D17.

Key words and phrases. Point cloud, level set methods, advection equation, reconstruction, image segmentation.

this method to data with strong noise or the segmentation of structures composed of cells is complicated. Consequently, we add point cloud data given by the user to the image to support the segmentation.

The main goal of processing point cloud data is to create a computerized three-dimensional (3D) model, representing the real object, from which the point cloud was created, as accurately as possible. This process can be very complex and time consuming. The prime cause for this difficulty is that the point cloud data has no information about ordering or connectivity. Other reasons, which can make the process difficult, are the unknown topology of the scanned object and the presence of noise in the data.

Various methods already exist to process such data sets. The most used approaches by commercial software are nonuniform B-spline surfaces, triangulated surfaces, and the substitution of point cloud by shapes defined by mathematical equations representing geometric objects as cuboid, cylinder, cone or sphere. Some software packages are available for different tasks, nevertheless, it is difficult to select the optimal one for all requirements. These software packages mainly use point cloud data obtained by 3D laser scanning or photogrammetric methods. Here, we analyzed data sets that included both point cloud data as well as 3D image intensity information that we want to utilize in the reconstruction process.

Another approach for the processing of point cloud data is given by application of the level set method [19], which has a wide range of applications in image processing, computer graphics, material science and physics [14, 11]. In image processing, the level set method is used for image segmentation. Our data sets contain 3D images, in which image intensity is given in every voxel. These images can represent biological data for which one may want to segment different continuous tissues consisting of separate cells. Such tissues do not have clear boundaries and as a result, automatic segmentation algorithms used so far may not give any useful result. To distinguish the tissues from the other parts of the whole image we can use manually identified point cloud data. Thus, the segment that we aim to identify, can be seen as a volume, the boundary of which is determined by the point cloud data related to the image information. In our algorithm, we enrich the level set motion by the information coming from 3D digital images, allowing the image segmentation combining 3D images and the point cloud data. Such a semi-automatic approach for segmentation of image objects without clearly observable boundaries has not been used in any previous articles.

The basic form of the algorithm consists in the solution of a partial differential equation (PDE) representing the level set motion. The solution of the PDE is calculated on a rectangular computational domain, which we choose according to the point cloud and image data. The point cloud data set must be a subset of the computational domain. The solution by the level set method results in the evolution of the level set function as a deformation of an initial guess. The reconstructed surface is then created as an isosurface of the level set function at the numerical steady state.

2. **Mathematical model.**

Our method is based on solution of the following advection equation with the curvature term,

$$u_t + v \cdot \nabla u - \delta \left|\nabla u\right| \nabla \cdot \left(\frac{\nabla u}{|\nabla u|}\right) = 0, \ (x,t) \in \Omega \times [0,T], \tag{1}$$

where $u(x,t)$ is an unknown function, v denotes the advective velocity, the parameter $\delta \in [0,1]$ determines influence of the curvature to the result, Ω is the computational domain, and $[0,T]$ is a time interval. This equation is coupled with homogeneous Neumann boundary conditions and an initial condition. The initial condition is given by help of the distance function to a surface containing the point cloud data Ω_0, see [7] section 2.2. Then, this initial condition is driven to reach the final segmentation result through the advective velocity v, smoothed by the curvature term.

The advective velocity v in (1) contains information on both the given point cloud data and 3D image intensity. For the point cloud data, the distance function d is calculated using the fast sweeping method [18]. To utilize the image information, we introduce the edge detector function $g^0 = g\left(\left|\nabla G_\sigma * I^0\right|\right)$, similarly as it is used in [13, 2, 10, 8, 9, 17] for the classical image segmentation. Here, I^0 denotes the 3D image intensity, G_σ represents a smoothing kernel applied to the image for presmoothing the gradients of the intensity, and g is a function defined as

$$g(s) = \frac{1}{1+Ks^2}, \tag{2}$$

where $K \geq 0$ is an empirically chosen parameter [12]. Then, the advective velocity is given by a combination of the distance function and the edge detector function gradients, i.e.

$$v = \theta(-\nabla d) + \rho\left(\frac{-\nabla g^0}{|\nabla g^0|_\varepsilon}\right). \tag{3}$$

In the model, the first term $-\nabla d$ drives the level sets of the solution for function $u(x,t)$ to the points of the cloud and the second term $\left(-\nabla g^0 / \left|\nabla g^0\right|_\varepsilon\right)$ drives them to the image edges in the neighborhood of the point cloud.

The term $\left|\nabla g^0\right|_\varepsilon$ is calculated according to the Evans-Spruck ε-regularization [4] as $\sqrt{\varepsilon^2 + |\nabla g^0|^2}$ where $\varepsilon > 0$. The parameters $\theta, \rho \in [0,1]$ determine the influence of two mentioned gradients on the surface reconstruction process. The edge detector gradient is normalized and regularized in order to get comparable speed of advection coming from the distance function and the edge detector function terms.

3. **Numerical discretization.** First of all, we apply a backward difference in time with a uniform time step τ and the semi-implicit time discretization in nonlinear curvature term of (1) to get

$$\frac{u^n - u^{n-1}}{\tau} + v \cdot \nabla u^n - \delta\left|\nabla u^{n-1}\right| \nabla \cdot \left(\frac{\nabla u^n}{|\nabla u^{n-1}|}\right) = 0. \tag{4}$$

The spatial discretization is performed on a uniform voxel grid where the voxels can be described as cubes with edge size h. The voxels are denoted by the indexes $p = (i,j,k)$ and at each voxel center the value of the image intensity is given by $I^0_{i,j,k}$, the value of computed distance function is denoted by $d_{i,j,k}$, and the computed value of unknown function u^n at time step n in voxel $p = (i,j,k)$ is denoted by both u^n_p and $u^n_{i,j,k}$. For the calculation of the norm of gradient $\left|\nabla u^{n-1}\right|$, on the voxel faces and in the voxel, and the calculation of $\left|\nabla G_\sigma * I^0\right|$ to determine the values of g^0 at voxels, we use the 3D tetrahedral finite element grid $\mathcal{T}_h$ illustrated in Figure 1 [2].

The tetrahedral finite elements in $\mathcal{T}_h$ are created by the following approach. Every voxel is divided into six pyramid shaped elements with the base surface given

by the voxel's faces and the common vertex by the voxel center. Each of these pyramids is joined with the neighboring pyramids, with which they have a common base surface. These newly formed octahedrons are then split into four tetrahedrons.

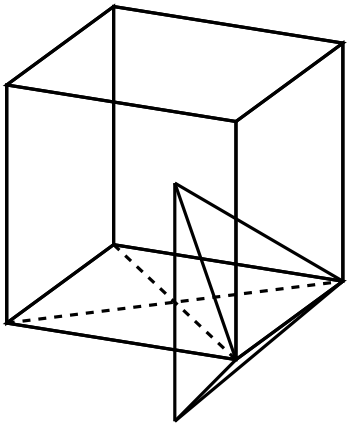

FIGURE 1. The voxel grid cell with a tetrahedral finite element.

For tetrahedral grid $\mathcal{T}_h$, we construct a co-volume mesh, which consists of cells $p = (i, j, k)$ corresponding again to the voxels at our grid, see also [2]. We denote the set of all neighboring cells q of p by C_p. The centers of these cells $q \in C_p$ are all connected to center of p by a common edge σ_{pq} of four tetrahedrons with the length $h_{pq} = h$. Each cell p is bounded by a plane for every $q \in C_p$, which is perpendicular to σ_{pq} and denoted by e_{pq}. The set of tetrahedrons T, which have σ_{pq} as an edge, is denoted by ε_{pq}. For every $T \in \varepsilon_{pq}$, c_{pq}^T is the area of the intersection of e_{pq} and T. Let N_p be the set of tetrahedrons that have p as a vertex. Let u_h be a piecewise linear function on $\mathcal{T}_h$. We use the notation $u_p = u_h(x_p)$ where x_p denotes the coordinates of the center of cell p.

The spatial discretization of (4) is derived by using the following form of the equation:

$$\frac{u^n - u^{n-1}}{\tau} + v \cdot \nabla u^n = \delta \left|\nabla u^{n-1}\right| \nabla \cdot \left(\frac{\nabla u^n}{\left|\nabla u^{n-1}\right|}\right). \tag{5}$$

We begin by integration of (5) over every p:

$$\int_p \frac{u^n - u^{n-1}}{\tau} dx + \int_p v \cdot \nabla u^n dx = \int_p \delta \left|\nabla u^{n-1}\right| \nabla \cdot \left(\frac{\nabla u^n}{\left|\nabla u^{n-1}\right|}\right) dx. \tag{6}$$

For the first part of left hand side in (6), we get the approximation in the form

$$\int_p \frac{u^n - u^{n-1}}{\tau} dx = m(p) \frac{u_p^n - u_p^{n-1}}{\tau}, \tag{7}$$

where $m(p)$ is a measure in $\mathbb{R}^3$ of the cell p. The second part of the left hand side can be written in an equivalent form by

$$v \cdot \nabla u = \nabla \cdot (uv) - u \nabla \cdot v,$$

$$\int_p v \cdot \nabla u^n dx = \int_p \nabla \cdot (u^n v)\, dx - \int_p u^n \nabla \cdot v\, dx.$$

Considering u^n constant on p in the second term on the right hand side and by using the divergence theorem, we get

$$\begin{aligned}\int_p \nabla\cdot(u^n v)dx - \int_p u^n \nabla\cdot v\,dx = \int_{\partial p} u^n v\cdot\bar{n}\,d\sigma - u_p^n\int_{\partial p} v\cdot\bar{n}\,d\sigma = \\ \sum_{q\in C_p} u_{pq}^n \int_{e_{pq}} v\cdot\bar{n}\,d\sigma - \sum_{q\in C_p} u_p^n \int_{e_{pq}} v\cdot\bar{n}\,d\sigma\end{aligned} \tag{8}$$

where u_{pq}^n is an approximate value of the level-set function on the face e_{pq} and $\bar{n}$ is the outer normal to p. We approximate $\int_{e_{pq}} v\cdot\bar{n}\,d\sigma$ in (8) by $v_{pq} = h^2 v\left(x_{pq}\right)\cdot\bar{n}$, where x_{pq} is a center of e_{pq}, and get

$$\int_p v\cdot\nabla u^n dx = \sum_{q\in C_p} v_{pq}\left(u_{pq}^n - u_p^n\right). \tag{9}$$

In order to use the upwind approach, the set C_p is divided into $C_p = C_p^{in}\cup C_p^{out}$, where $C_p^{in} = \{q\in C_p,\, v_{pq} < 0\}$, which consists of the inflow boundaries, and $C_p^{out} = \{q\in C_p,\, v_{pq} > 0\}$ consisting of the outflow boundaries. By the upwind approach, we set the values u_{pq}^n to u_q^n if $q\in C_p^{in}$ and to u_p^n if $q\in C_p^{out}$. After these definitions, we can rewrite (9) into

$$\int_p v\cdot\nabla u^n dx = \sum_{q\in C_p} min\left(v_{pq},0\right)\left(u_q^n - u_p^n\right). \tag{10}$$

For discretization of the right hand side of (6), we consider the term in front of divergence constant on p and then we use the divergence theorem to get

$$\int_p \delta\left|\nabla u^{n-1}\right|\nabla\cdot\left(\frac{\nabla u^n}{\left|\nabla u^{n-1}\right|}\right)dx = \delta\left|\nabla u_p^{n-1}\right|\sum_{q\in C_p}\int_{e_{pq}}\frac{1}{\left|\nabla u^{n-1}\right|}\frac{\partial u^n}{\partial\bar{n}}d\sigma. \tag{11}$$

The integral $\int_{e_{pq}}\frac{1}{\left|\nabla u^{n-1}\right|}\frac{\partial u^n}{\partial\bar{n}}d\sigma$ and $\left|\nabla u_p^{n-1}\right|$ in (11) is approximated numerically using piecewise linear reconstruction of u^{n-1} on the tetrahedral grid $\mathcal{T}_h$, thus we get

$$\int_p \delta\left|\nabla u^{n-1}\right|\nabla\cdot\left(\frac{\nabla u^n}{\left|\nabla u^{n-1}\right|}\right)dx = \delta\left|\nabla u_p^{n-1}\right|\sum_{q\in C_p}\left(\sum_{T\in\varepsilon_{pq}} c_{pq}^T\frac{1}{\left|\nabla u_T^{n-1}\right|}\right)\frac{u_q^n - u_p^n}{h}$$

where

$$\left|\nabla u_p^{n-1}\right| = M_p^{n-1} = \sum_{T\in N_p}\frac{m\left(T\cap p\right)}{m\left(p\right)}\left|\nabla u_T^{n-1}\right| \tag{12}$$

and $\left|\nabla u_T^{n-1}\right|$ denotes the gradient of u_h^{n-1} on T. Then the discretized form of the equation (5) is given by

$$\begin{aligned}m\left(p\right)\frac{u_p^n - u_p^{n-1}}{\tau} + \sum_{q\in C_p} min\left(v_{pq},0\right)\left(u_q^n - u_p^n\right) = \\ \delta\, M_p^{n-1}\sum_{q\in C_p}\left(\sum_{T\in\varepsilon_{pq}} c_{pq}^T\frac{1}{\left|\nabla u_T^{n-1}\right|}\right)\frac{u_q^n - u_p^n}{h}.\end{aligned} \tag{13}$$

After rearranging (13) and applying the Evans-Spruck ε-regularization $\left|\nabla u_T^{n-1}\right|_\varepsilon = \sqrt{\varepsilon^2 + \left|\nabla u_T^{n-1}\right|}$, the equation has the following form

$$\begin{aligned} u_p^n + \frac{\tau}{m\left(p\right)} \Bigg(& \sum_{q \in C_p} min\left(v_{pq}, 0\right)\left(u_q^n - u_p^n\right) \\ -\delta\, M_p^{n-1} & \sum_{q \in C_p} \left(\sum_{T \in \varepsilon_{pq}} c_{pq}^T \frac{1}{\left|\nabla u_T^{n-1}\right|_\varepsilon} \right) \frac{u_q^n - u_p^n}{h} \Bigg) = u_p^{n-1}. \end{aligned} \tag{14}$$

If we define the coefficients

$$a_{pq}^{n-1} = \frac{\tau}{m\left(p\right)} \left(min\left(v_{pq}, 0\right) - \delta\, M_p^{n-1} \frac{1}{h} \sum_{T \in \varepsilon_{pq}} c_{pq}^T \frac{1}{\left|\nabla u_T^{n-1}\right|_\varepsilon} \right), \tag{15}$$

we get from (14) a system of linear equations with strictly diagonally dominant M-matrix in the form

$$u_p^n + \sum_{q \in N_p} a_{pq}^{n-1} \left(u_q^n - u_p^n\right) = u_p^{n-1}, \tag{16}$$

which by addition of the homogeneous Neumann boundary conditions and initial values u_p^0 is solved in every time step n. Since a_{pq}^{n-1} are non-positive coefficients a unique solution exists $(u_1^n, ..., u_M^n)$ of (16) where M is a number of unknowns, for any $\tau > 0$, $\varepsilon > 0$, and for every $n = 1, ..., N$.

For the numerical solution of (16), we need the values of matrix coefficients a_{pq}. We apply a "finite-difference notation", the cell p is denoted by the index triplet (i, j, k). The value u_p^n is associated with $u_{i,j,k}^n$. At each cell p, the set N_p consists of 24 tetrahedrons on which we compute $|\nabla u_T^n|_\varepsilon$ denoted by $Gu_{i,j,k}^{n,l}$, $l = 1, ..., 24$. To derive the computation of $Gu_{i,j,k}^{n,l}$, we introduce a similar notation as used in [9].

First, we define the index sets P^1, P^2 and P^3 as

$$\begin{aligned} P^1 &= \{(r,s,t)\,; r,s,t \in \{-1,0,1\}\,; |r| + |s| + |t| = 1\} \\ P^2 &= \{(r,s,t)\,; r,s,t \in \{-1,1\}\} \\ P^3 &= \{(r,s,t)\,; r,s,t \in \{-1,0,1\}\,; |r| + |s| + |t| = 2\} \end{aligned}$$

Considering any $(r,s,t) \in P^1$ let $x_{i,j,k}^{r,s,t}$ be the points where the edges σ_{pq} intersect the planes e_{pq}, for every $q \in C_p$. Every vertex of a cubic element is represented by $z_{i,j,k}^{r,s,t}$, for $(r,s,t) \in P^2$. The midpoints between each neighboring vertices of cubic elements are denoted by $y_{i,j,k}^{r,s,t}$, for $(r,s,t) \in P^3$. The explained notation is presented in Figure 2.

The values u^{n-1} approximated at $x_{i,j,k}^{r,s,t}$, $z_{i,j,k}^{r,s,t}$ and $y_{i,j,k}^{r,s,t}$ are denoted by $u_{i,j,k}^{r,s,t}$, leaving out the time index. In $x_{i,j,k}^{r,s,t}$, $(r,s,t) \in P^1$, the values are calculated as the average value for two neighboring cells,

$$u_{i,j,k}^{r,s,t} = \frac{1}{2}\left(u_{i,j,k}^{n-1} + u_{i+r,j+s,k+t}^{n-1}\right), \text{for } (r,s,t) \in P^1.$$

In $z_{i,j,k}^{r,s,t}$, $(r,s,t) \in P^2$, as the average value of the 8 voxels which have $z_{i,j,k}^{r,s,t}$ as a common vertex,

$$u_{i,j,k}^{r,s,t} = \frac{1}{8}(u_{i,j,k}^{n-1} + u_{i+r,j,k}^{n-1} + u_{i,j+s,k}^{n-1} + u_{i,j,k+t}^{n-1} + u_{i+r,j+s,k}^{n-1} +$$

$$u^{n-1}_{i+r,j,k+t} + u^{n-1}_{i,j+s,k+t} + u^{n-1}_{i+r,j+s,k+t}), \text{for } (r,s,t) \in P^2.$$

In $y^{r,s,t}_{i,j,k}$, $(r,s,t) \in P^3$, as the averages of values $u^{r,s,t}_{i,j,k}$ in points $z^{r,s,t}_{i,j,k}$,

$$u^{r,s,t}_{i,j,k} = \frac{1}{2}\left(u^{r+\xi(r),s+\xi(s),t+\xi(t)}_{i,j,k} + u^{r-\xi(r),s-\xi(s),t-\xi(t)}_{i,j,k}\right), \text{for } (r,s,t) \in P^3,$$

where

$$\xi(x) = \begin{cases} 0 & \text{if } x \in \{-1,1\} \\ 1 & \text{if } x = 0 \end{cases}.$$

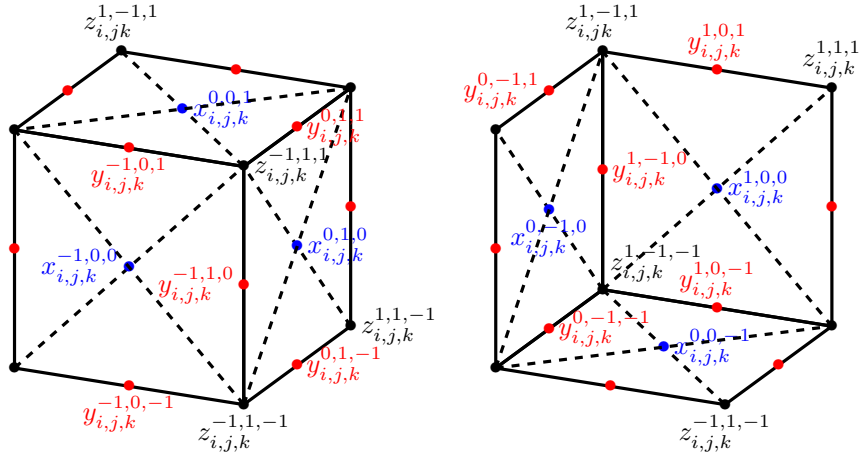

FIGURE 2. Notation for the additional points of a grid cell used for calculation of $Gu^{n,l}_{i,j,k}$, $l = 1, ..., 24$

Norm of gradient $\left|\nabla u^{n-1}_T\right|_\varepsilon$ is computed by using values in two neighboring voxel centers, in vertices of voxels corresponding to tetrahedral edge, $z^{r,s,t}_{i,j,k}$, and center of that edge, $y^{r,s,t}_{i,j,k}$, and center of the face, $x^{r,s,t}_{i,j,k}$. Approximation of partial derivatives included in gradient computation for one tetrahedra is illustrated in Figure 3. By red color and dashed pattern we indicate the line which is used for approximating $\frac{\partial u_h}{\partial x}$, by green dotted line approximation of $\frac{\partial u_h}{\partial y}$, and blue dash-dotted line approximation of $\frac{\partial u_h}{\partial z}$. For other tetrahedrons it is done similarly. Thereby, we get the values $Gu^{n-1,l}_{i,j,k}$ $l = 1, ..., 24$.

Besides $\left|\nabla u^{n-1}_T\right|_\varepsilon$, given using the above explanation and M^{n-1}_p given by (12), we need to determine also the values v_{pq} in (15). Taking into account the definition (3), we get

$$v_{pq} = h^2\left(\theta\left(-\nabla d\right) + \rho\left(\frac{-\nabla g^0}{|\nabla g^0|_\varepsilon}\right)\right)\Big|_{x_{pq}} \cdot \bar{n}. \tag{17}$$

On the six voxel faces we can write

$$v^t_{i,j,k} = -\theta h\left(d_{i,j,k+1} - d_{i,j,k}\right) - \rho h\left(\left(g^0_{i,j,k+1} - g^0_{i,j,k}\right) / \left|\nabla g^0_{i,j,k}\right|_\varepsilon\right)$$

$$v^b_{i,j,k} = \quad \theta h\left(d_{i,j,k} - d_{i,j,k-1}\right) + \rho h\left(\left(g^0_{i,j,k} - g^0_{i,j,k-1}\right) / \left|\nabla g^0_{i,j,k}\right|_\varepsilon\right)$$

$$v^n_{i,j,k} = -\theta h\left(d_{i+1,j,k} - d_{i,j,k}\right) - \rho h\left(\left(g^0_{i+1,j,k} - g^0_{i,j,k}\right) / \left|\nabla g^0_{i,j,k}\right|_\varepsilon\right)$$

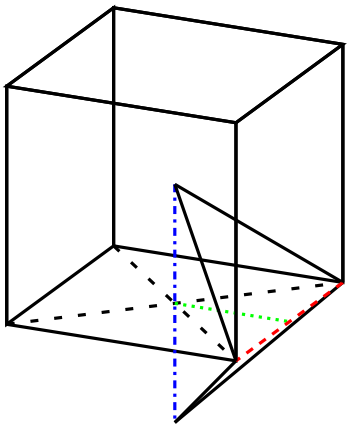

FIGURE 3. Tetrahedral finite element with marked edges for approximation of partial derivatives.

$$
\begin{aligned}
v^s_{i,j,k} &= \quad \theta h \left(d_{i,j,k} - d_{i-1,j,k}\right) + \rho h \left(\left(g^0_{i,j,k} - g^0_{i-1,j,k}\right) / \left|\nabla g^0_{i,j,k}\right|_\varepsilon\right) \\
v^e_{i,j,k} &= -\theta h \left(d_{i,j+1,k} - d_{i,j,k}\right) - \rho h \left(\left(g^0_{i,j+1,k} - g^0_{i,j,k}\right) / \left|\nabla g^0_{i,j,k}\right|_\varepsilon\right) \\
v^w_{i,j,k} &= \quad \theta h \left(d_{i,j,k} - d_{i,j-1,k}\right) + \rho h \left(\left(g^0_{i,j,k} - g^0_{i,j-1,k}\right) / \left|\nabla g^0_{i,j,k}\right|_\varepsilon\right).
\end{aligned}
$$

To obtain these values, we need to evaluate the discrete values of $g^0_{i,j,k} = g\left(\left|\nabla G_\sigma * I^0_{i,j,k}\right|\right)$ for every voxel. To achieve this, we solve the convolution $I^\sigma = G_\sigma * I^0$ through solving the linear heat equation by the implicit scheme with a time step related to σ, where also the homogeneous Neumann boundary conditions are applied.

Now, we can calculate $\left|\nabla I^\sigma_{i,j,k}\right|$ analogously to (12) and finally we get

$$
g^0_{i,j,k} = g\left(\frac{1}{24}\sum_{l=1}^{24} GI^{\sigma,l}_{i,j,k}\right). \tag{18}
$$

With the discrete values of g^0_p, we compute norm of regularized gradient $\left|\nabla g^0_p\right|_\varepsilon = \sqrt{\varepsilon^2 + \left|\nabla g^0_p\right|^2}$ where $\left|\nabla g^0_p\right|$ is given according to the "magic formula" as described in [5]:

$$
\left|\nabla g^0_p\right|^2 = \frac{1}{m\left(p\right)} \sum_{e_{pq} \in C_p} m\left(e_{pq}\right) \frac{1}{d_{pq}} \left(g^0_{pq} - g^0_p\right)^2 \tag{19}
$$

where d_{pq} is the distance between the center of cell p and the center of e_{pq}. With $m\left(p\right) = h^3$, $m\left(e_{pq}\right) = h^2$, $d_{pq} = \frac{h}{2}$ and using $g^0_{pq} = \frac{g^0_q + g^0_p}{2}$, we get:

$$
\left|\nabla g^0_p\right| = \sqrt{\frac{1}{2}\sum_{q \in C_p} \left(\frac{g^0_q - g^0_p}{h}\right)^2}. \tag{20}
$$

So we get

$$\left|\nabla g^0_{i,j,k}\right| = \sqrt{\frac{1}{2}\sum_{(r,s,t)\in P^1}\left(\frac{g^0_{i+r,j+s,k+t} - g^0_{i,j,k}}{h}\right)^2} \tag{21}$$

By using above considerations, we define

$$b_{i,j,k} = \frac{\tau}{h^3}\left(min\left(v^b_{i,j,k},0\right) - \delta\, M^{n-1}_{i,j,k}\frac{h}{4}\sum_{l=1}^{4}\left(\varepsilon^2 + \left(Gu^{n-1,l}_{i,j,k}\right)^2\right)^{-\frac{1}{2}}\right)$$

$$t_{i,j,k} = \frac{\tau}{h^3}\left(min\left(v^t_{i,j,k},0\right) - \delta\, M^{n-1}_{i,j,k}\frac{h}{4}\sum_{l=5}^{8}\left(\varepsilon^2 + \left(Gu^{n-1,l}_{i,j,k}\right)^2\right)^{-\frac{1}{2}}\right)$$

$$n_{i,j,k} = \frac{\tau}{h^3}\left(min\left(v^n_{i,j,k},0\right) - \delta\, M^{n-1}_{i,j,k}\frac{h}{4}\sum_{l=9}^{12}\left(\varepsilon^2 + \left(Gu^{n-1,l}_{i,j,k}\right)^2\right)^{-\frac{1}{2}}\right)$$

$$s_{i,j,k} = \frac{\tau}{h^3}\left(min\left(v^s_{i,j,k},0\right) - \delta\, M^{n-1}_{i,j,k}\frac{h}{4}\sum_{l=13}^{16}\left(\varepsilon^2 + \left(Gu^{n-1,l}_{i,j,k}\right)^2\right)^{-\frac{1}{2}}\right)$$

$$e_{i,j,k} = \frac{\tau}{h^3}\left(min\left(v^e_{i,j,k},0\right) - \delta\, M^{n-1}_{i,j,k}\frac{h}{4}\sum_{l=17}^{20}\left(\varepsilon^2 + \left(Gu^{n-1,l}_{i,j,k}\right)^2\right)^{-\frac{1}{2}}\right)$$

$$w_{i,j,k} = \frac{\tau}{h^3}\left(min\left(v^w_{i,j,k},0\right) - \delta\, M^{n-1}_{i,j,k}\frac{h}{4}\sum_{l=21}^{24}\left(\varepsilon^2 + \left(Gu^{n-1,l}_{i,j,k}\right)^2\right)^{-\frac{1}{2}}\right)$$

where $M^{n-1}_{i,j,k}$ is given by

$$M^{n-1}_{i,j,k} = \sqrt{\varepsilon^2 + \left(\frac{1}{24}\sum_{l=1}^{24} Gu^{n-1,l}_{i,j,k}\right)^2}$$

and the diagonal coefficient is defined as

$$c_{i,j,k} = 1 - b_{i,j,k} - t_{i,j,k} - n_{i,j,k} - s_{i,j,k} - e_{i,j,k} - w_{i,j,k}.$$

With these notations, we rewrite (16) into the form of a system of linear equations

$$\begin{aligned} &c_{i,j,k}u^n_{i,j,k} + b_{i,j,k}u^n_{i,j,k-1} + t_{i,j,k}u^n_{i,j,k+1} + n_{i,j,k}u^n_{i+1,j,k} \\ &+s_{i,j,k}u^n_{i-1,j,k} + e_{i,j,k}u^n_{i,j+1,k} + w_{i,j,k}u^n_{i,j-1,k} = u^{n-1}_{i,j,k}, \end{aligned} \tag{22}$$

for all (i,j,k). This system is solved at every time step by the SOR (Successive Over Relaxation) iterative method. We stop the calculation when the residuum between two consecutive time steps is lower than a chosen tolerance.

For the calculation, we need discrete initial values for the level set function $u\,(x,0)$ in the zero time step. This initial condition is determined by a simple tagging algorithm using distance function to the point cloud. With further modification of this tagging algorithm, we are able to construct a band around the area between the initial surface and point cloud data for computation acceleration [7]. For finding the surface, which we want to reconstruct, it is sufficient to update the solution values on grid cells contained only in such band.

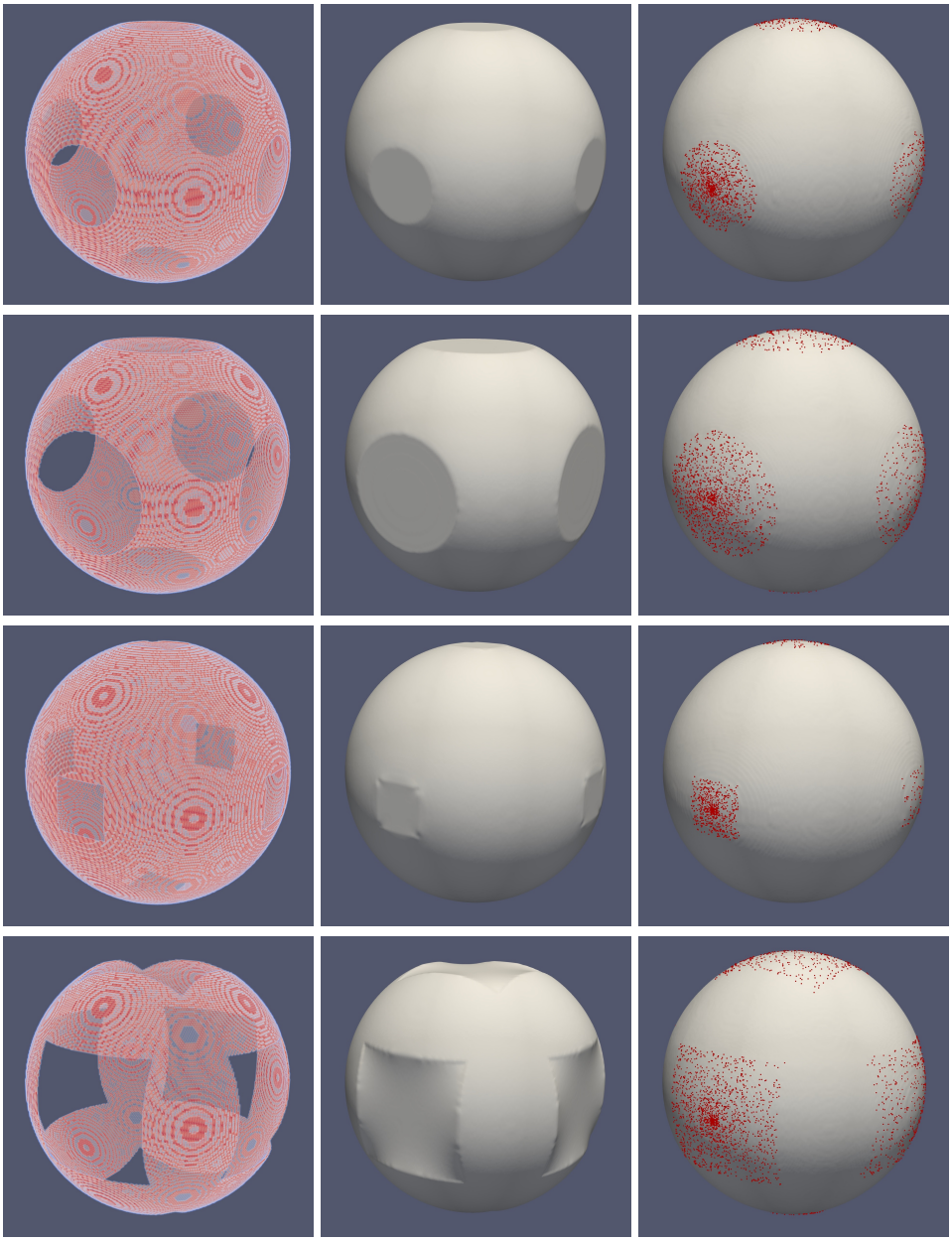

FIGURE 4. In the first column there are 3D images of a sphere with 6 symmetrically placed holes. The experiment was executed on spheres with holes of different shapes and sizes. The second column shows the result of the mathematical model (1) with $\theta = 0$ and $\rho = 1$ for the advective velocity (3). In the third column we show the result after we added points to the missing parts of the sphere and set $\theta = 1$, $\rho = 1$ in (3).

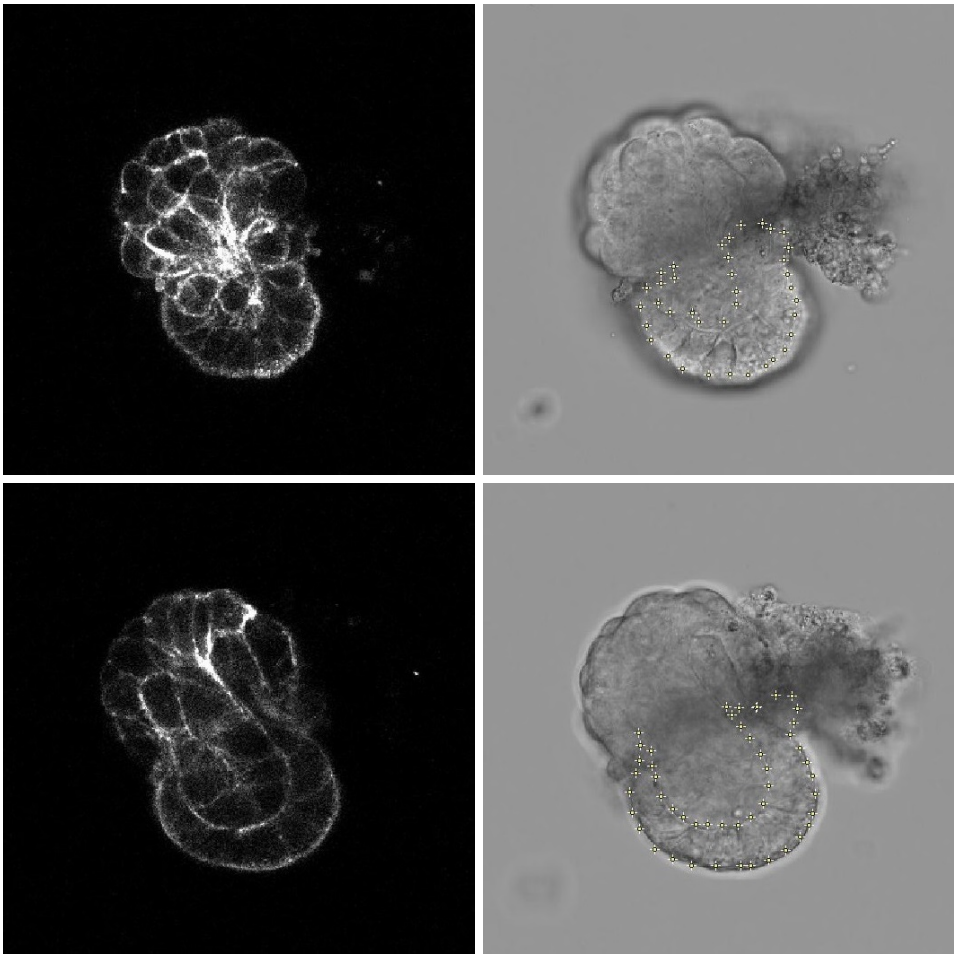

FIGURE 5. Sections of the embryo together with the marked points for visceral endoderm (VE).

3.1. **Numerical experiments.** In the numerical experiments, we first set $\theta = 0$ and $\rho = 1$ for the advective velocity (3). We prepared 3D images of a sphere, with the radius $r = \frac{\sqrt{3}}{2}$, on which we symmetrically placed 6 holes. We executed the experiments with holes of different shapes and sizes. The 3D images can be seen in the first column of the Figure 4. In the second column, the results we obtained are represented. As expected, the missing parts were completed by the minimal surfaces.

For the second experiment, we added points on the sphere randomly distributed in the areas of the holes and set $\theta = 1$ to see the effects with combined information of point cloud data and image intensity. In the third column of the Figure 4, we see the added point cloud together with the results of the reconstruction. In all cases the missing parts of the sphere were accurately recreated.

In both cases, the calculations were executed on a grid with 260^3 voxels which have edge size $h = 0.01$ and we set the parameter δ for the curvature term in (1) to 0.05.

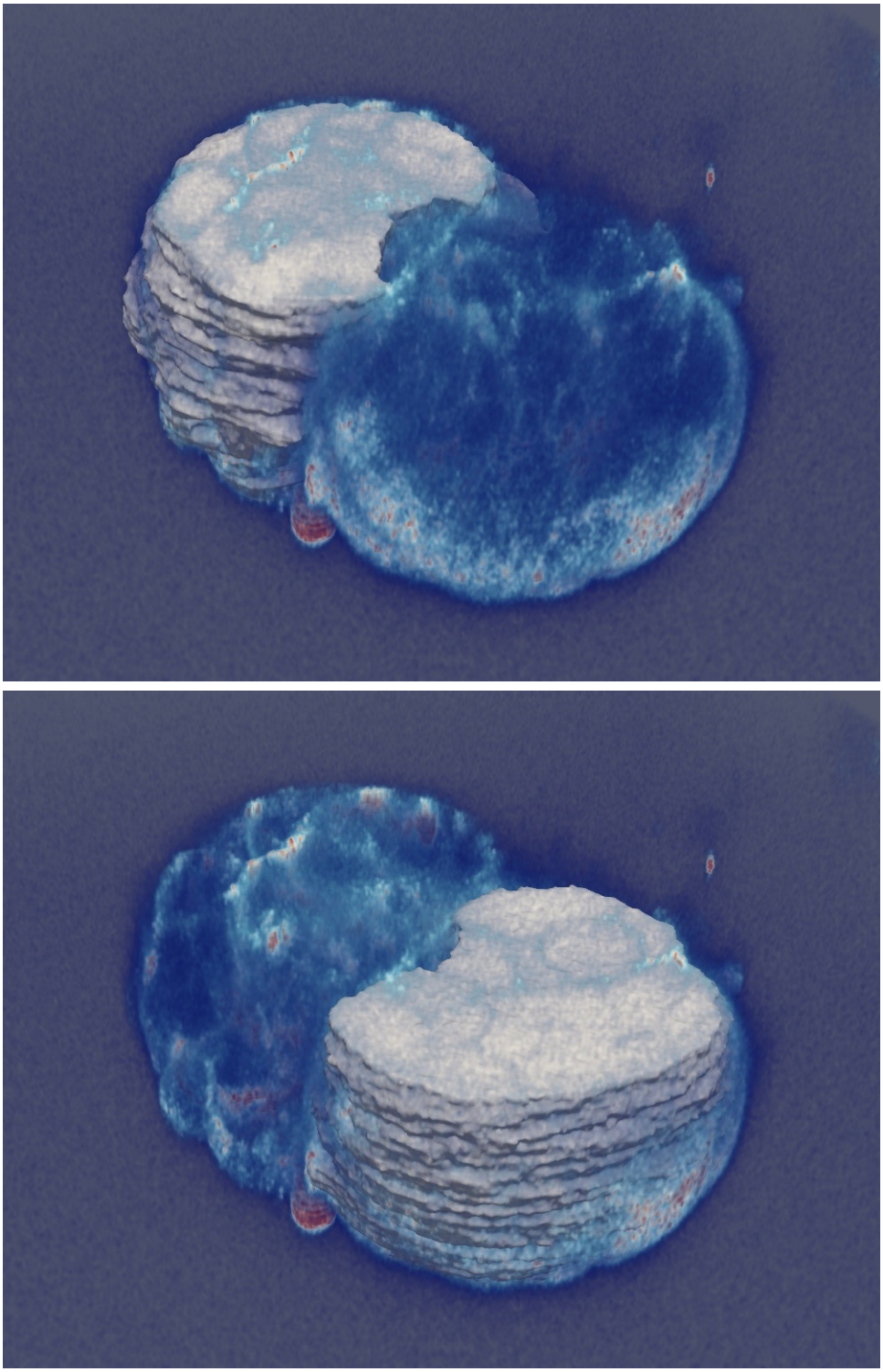

FIGURE 6. 3D image of embryo structure, visualized with the reconstructed ExE part of the embryo in the upper picture and with VE part in the lower one.

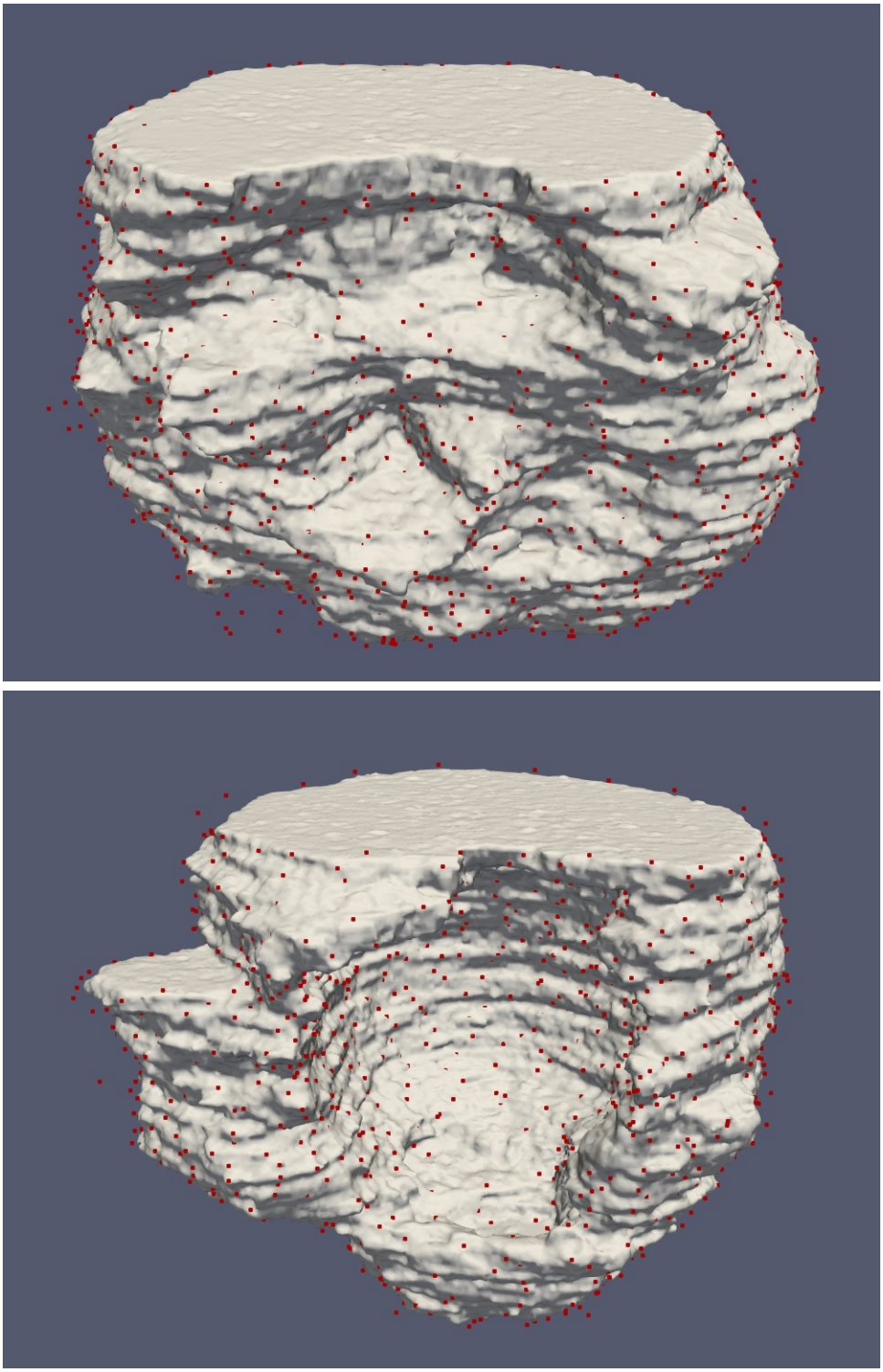

FIGURE 7. Reconstructed surface of extraembryonic ectoderm (ExE) and visceral endoderm (VE), visualized with point cloud.

To test our method on biological samples, we processed 3D microscopic images of mouse embryos at early peri-implantation stages provided by the group of Magdalena Zernicka-Goetz, University of Cambridge, in course of an ImageInLife EC funded project. At the time of implantation, the embryo is composed of three distinct tissues, the epiblast, which will give rise to the organism, is enveloped by two extraembryonic lineages, the visceral endoderm (VE) and the extra-embryonic ectoderm (ExE). Both provide essential signaling cues mediating proper embryonic development eventually giving rise to yolk sac and placenta respectively [1, 15, 16].

To gain further information on the 3D shape of both extra-embryonic tissues, we decided to reconstruct both ExE and VE taking advantage of our new model (1). For this, the embryo was scanned in 58 sections along the z-axis in 1μm steps. In Figure 5, we can see section 20 (top row) and 40 (bottom row). Figure 6 depicts the entire embryo through volume rendering of the 3D image.

As both tissues could not be segmented automatically by using 3D image intensity information, the tissue borders were identified in 2D slices by marking through single points. This manual process took around 2 hours for each tissue in the whole 3D image. These points represent a basis for 3D point cloud data entering the model equation (1). The idea of using point cloud for segmentation instead of image information, was first explored in [3], and it is used in the current work as well, but combined with the 3D image intensity information. We can see the examples of manually identified points in the second column of Figure 5 for VE part. We applied linear interpolation between neighboring points in every 2D slice and combined all the points of all sections to create a 3D point cloud data set. We applied our algorithm to solve (1) and present the results in Figure 7 visualized with the supporting points.

In this example the, dimension of the 3D image was 512x512x58 voxels. The calculations were performed on a computer with an Intel(R) Core(TM) i7-5820 CPU 3.30GHz processor and 128 GB RAM. We measured the CPU times for both cases. For ExE part the calculation lasted 1017 seconds and for the VE part 1420 seconds. We also applied parallelization to our implementation with the utilization of the OpenMP library. For parallel calculations we used 6 threads on the mentioned computer. After this the calculation of the ExE part took 448 seconds and for the VE part 599 seconds.

Acknowledgments. This work was supported by the grants APVV-15-0522 and EC project ImageInLife - Marie Sklodowska-Curie grant agreement No. 721537.

REFERENCES

[1] N.Christodoulou, C. Kyprianou, A. Weberling, R. Wang and G. Cui, et al., Sequential formation and resolution of multiple rosettes drive embryo remodeling after implantation, *Nature Cell Biology*, **20** (2018), 1278–1289.

[2] S. Corsaro, K. Mikula, A. Sarti and F. Sgallari, Semi-implicit covolume method in 3D image segmentation, *SIAM J. Sci. Comput.*, **28** (2006), 2248–2265.

[3] S. Dyballa, T. Savy, P. Germann, K. Mikula and M.Remešíková, et al., Distribution of neurosensory progenitor pools during inner ear morphogenesis unveiled by cell lineage reconstruction, *eLife*, **6** (2017).

[4] L. C. Evans and J. Spruck, Motion of level sets by mean curvature. I, *J. Differential Geom.*, **33** (1991), 635–681.

[5] R. Eymard, A. Handlovičvá and K. Mikula, Study of a finite volume scheme for the regularised mean curvature flow level set equation, *IMA J. Numer. Anal.*, **31** (2011), 813–846.

[6] E. Faure, et al., A workflow to process 3D+time microscopy images of developing organisms and reconstruct their cell lineage, *Nature Communications*, **7** (2016).

[7] B. Kósa, J. Haličková-Brehovská and K. Mikula, New efficient numerical method for 3D point cloud surface reconstruction by using level set methods, *Proceedings of Equadiff 2017 Conference*, 2017, 387–396.
[8] K. Mikula, N. Peyriéras, M. Remešíková and A. Sarti, 3D embryogenesis image segmentation by the generalized subjective surface method using the finite volume technique, in *Finite Volumes for Complex Applications V*, ISTE, London, 2008, 585—592.
[9] K. Mikula and M. Remešíková, Finite volume schemes for the generalized subjective surface equation in image segmentation, *Kybernetika*, **45** (2009), 646–656.
[10] K. Mikula and A. Sarti, Parallel co-volume subjective surface method for 3D medical image segmentation, in *Deformable Models*, Topics in Biomedical Engineering. International Book Series, Springer, NY, 2007, 123–160.
[11] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, Applied Mathematical Sciences, 153, Springer-Verlag, New York, 2003.
[12] P. Perona and J. Malik, Scale-space and edge detection using anisotropic diffusion, *IEEE Trans. Pattern Anal. Machine Intelligence*, **12** (1990), 629–639.
[13] A. Sarti, R. Malladi and J. A. Sethian, Subjective surfaces: A method for completing missing boundaries, *Proc. Natl. Acad. Sci. USA*, **97** (2000), 6258–6263.
[14] J. A. Sethian, *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Material Science*, Cambridge Monographs on Applied and Computational Mathematics, 3, Cambridge University Press, Cambridge, 1999.
[15] M. N. Shahbazi, A. Scialdone, N. Skorupska, A. Weberling and G. Recher, et al., Pluripotent state transitions coordinate morphogenesis in mouse and human embryos, *Nature*, **552** (2017), 239–243.
[16] M. N. Shahbazi and M. Zernicka-Goetz, Deconstructing and reconstructing the mouse and human early embryo, *Nature Cell Biology*, **20** (2018), 878–887.
[17] C. Zanella, M. Campana, B. Rizzi, C. Melani and G. Sanguinetti, et al., Cells segmentation from 3-D confocal images of early zebrafish embryogenesis, *IEEE Trans. Image Process.*, **19** (2010), 770–781.
[18] H. Zhao, A fast sweeping method for Eikonal equations, *Math. Comp.*, **74** (2005), 603–627.
[19] H. Zhao, S. Osher, B. Merriman and M. Kang, Implicit and nonparametric shape reconstruction from unorganized data using a variational level set method, *Comput. Vision Image Understanding*, **80** (2000), 295–319.

Received January 2019; revised October 2019.

E-mail address: kosa@math.sk
E-mail address: mikula@math.sk
E-mail address: markjoeuba@gmail.com
E-mail address: absw2@cam.ac.uk
E-mail address: nchris06@ucy.ac.cy
E-mail address: mz205@cam.ac.uk