Extraction of the intercellular skeleton from 2D microscope images of early embryogenesis

Paul Bourgine[1], Peter Frolkovič[2], Karol Mikula[2], Nadine Peyriéras[3], and Mariana Remešíková[2]

[1] CREA, Ecole Polytechnique-CNRS, 1 rue Descartes, 75005, Paris, France
paul.bourgine@polytechnique.edu
[2] Department of Mathematics, Slovak University of Technology, Radlinského 11, 81368 Bratislava,
frolkovic@math.sk, mikula@math.sk, remesikova@math.sk
[3] CNRS-DEPSN, Avenue de la Terasse, 91198, Gif-sur-Yvette, France
nadine.peyrieras@inaf.cnrs-gif.fr

Abstract. We suggest an efficient method for automatic detection of the intercellular skeleton in microscope images of early embryogenesis. The method is based on the solution of two advective PDEs. First, we solve numerically the time relaxed eikonal equation in order to obtain the signed distance function to a given set – a set of points representing cell centers or a set of closed curves representing segmented inner borders of cells. The second step is a segmentation process driven by the advective version of subjective surface equation where the velocity field is given by the gradient of the computed distance function. The first equation is discretized by Rouy-Tourin scheme and we suggest a fixing strategy that significantly improves the speed of the computation. The second equation is solved using a classical upwind strategy. We present several test examples and we show a practical application - the intercellular skeleton extracted from a 2D image of a zebrafish embryo.

1 Introduction

The measure of the cell contact surface is an important quantitative characteristic of a living organism, especially during its embryonic development [6]. Together with other characteristics, e.g. the volume of the embryo, the global and local density of cells, the density of cell divisions etc., it provides an insight into the process of the evolution of the organism and allows to detect abnormalities or to compare individuals evolving in different conditions. The intercellular skeleton can be extracted from the miscroscope images of the evolving embryo. Fig. 1 shows an example of suitable image data. These images display significant cell structures (cell nuclei and cell membranes) of a zebrafish embryo at an early stage of its development and they were obtained by a two-photon confocal microscope.

The main goal of our paper is to introduce an efficient and easily implementable method for detecting the intercellular skeleton. Our technique is based

on numerical solution of a pair of advective partial differential equations. The first step is the solution of the time relaxed eikonal equation with a special Dirichlet type condition. The solution of such an equation is the distance function to a given set. This can be a set of points representing cell centers or a set of closed curves representing inner borders of cells obtained by segmentation. In case we deal with the curves, we construct the signed distance function with negative values in the interior part. We discretize the problem using the explicit Rouy-Tourin scheme and we suggest to extend the original scheme by a fixing technique. The idea of fixing is based on the fact that the Rouy-Tourin scheme applied to the time relaxed eikonal equation produces in every point monotonically increasing values approaching the value of the distance function. This means that at some moment, the value will reach some steady state and it will not change anymore. Then the point can be excluded from the calculations. This strategy gradually reduces the computational cost of a single time step, it provides a significant improvement of the efficiency of the method and it brings a natural stopping criterion for the computation. We compare the performance of our algorithm with the computation of the exact distance function and we provide some examples of situations when the numerical solution can be obtained faster. The second step of our procedure is the segmentation using the advective version of the subjective surface equation. For each cell, we construct an initial segmentation function and after, all its level sets are evolved according to the velocity field given by the gradient of the computed signed distance function. By taking one of the level sets of the final form of the evolving function, we obtain the part of the required intercellular skeleton corresponding to one particular cell. The complete skeleton is constructed as the union of the results corresponding to individual cells. Using the distance function corresponding to cell centers, we get a Voronoi type cell skeleton that is already a good approximation of the real one as the cell formations are naturally similar to Voronoi tiling. A very realistic skeleton localization can be obtained if we consider the distance function to the segmented inner boundaries of the cells, assuming that we have a good quality cell segmentation. For pratical purposes, it is even sufficient to perform only a few time steps of the Rouy-Tourin scheme and use a very rough estimate of the distance function in order to obtain a correctly oriented velocity field for the segmentation. This makes the method very efficient without loss of the quality of the resulting skeleton. The advective subjective surface equation is discretized by an explicit upwind approach.

The paper is organized as follows. In Sec. 2, we describe the mathematical models for the two substeps of the procedure. Sec. 3 briefly presents the numerical schemes and it explains the idea of the fixing algorithm intended to reduce the CPU time needed to compute the numerical solution of the time relaxed eikonal euqation. In Sec. 4, we provide a series of numerical experiments and test examples as well as an example of an intercellular skeleton extracted from a 2D microscope image of a zebrafish embryo.

Let us note that we present a method for solving two-dimensional problems, but the extension to three dimensions is rather straightforward.

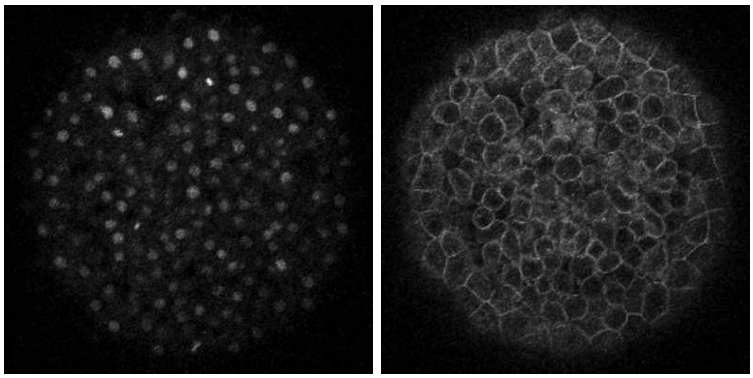

Fig. 1. 2D slices of 3D image of a zebrafish embryo. Left, the cell nuclei. Right, the cell membranes.

2 Mathematical models

The first equation involved in our skeleton extraction strategy is the eikonal equation with time relaxation

$$d_t + |\nabla d| = 1 \tag{1}$$

solved in the domain $\Omega \times [0, T_D]$ where Ω is the image domain and coupled with a Dirichlet type condition

$$d(x,t) = 0, \quad x \in \Omega_0 \subset \Omega. \tag{2}$$

By the problem formulated in this way the solution d approaches, as time is evolving, the distance function to the set Ω_0. In our case, as we have already mentioned, Ω_0 can be a set of points corresponding to approximate cell centers or a set of closed curves representing the segmented inner boundaries of the cells. Since for the segmented cell shapes we are able to recognize their interior due to the final form of the cell segmentation function [1, 2], once we have computed the distance function, the corresponding signed distance function can be constructed straightforwardly. The distance function corresponding to a set of points is always positive.

In the second step, we use the computed signed distance function in the advective part of the subjective surface model [4, 2] and we solve the equation

$$u_t + \nabla g \cdot \nabla u = 0 \tag{3}$$

where $(x,t) \in \Omega \times [0, T_S]$ and $g(x) = d^p(x)$ according to [8] or $g(x) = -1/(1 + Kd^p(x))$ with $K > 0$, $p > 0$ as in [4, 2]. The unknown function u is initialized by a piecewise constant profile localized around the approximate cell center. Then it is evolved by (3) to segment the intercellular borders. Due to the properties of the function d (see Fig. 3 and 6), the border lines of the neighboring cells corresponding to the ridges of the distance function are attached to each other and thus form the intercellular skeleton.

3 Numerical schemes

3.1 Time relaxation method with fixing for computing the distance function

In order to solve the equation (1) with the condition (2) numerically, we use an explicit time discretization with time step τ_D. After, the equation is discretized in space by the Rouy-Tourin scheme [3], cf. also [5, 7]. As it is natural for image processing applications, the space grid elements correspond to the pixels of the image. Considering our space domain rectangular with dimensions $L_x \times L_y$, we get a uniform space grid with square elements V_{ij}, $i = 1 \dots n_x$, $j = 1 \dots n_y$, $n_x = L_x/h_D$, $n_y = L_y/h_D$ where h_D is the length of the side of the pixel. For each volume V_{ij}, let d_{ij} represent the approximate value of the solution d in the center of V_{ij} in time step n. Let us define M_{ij}^{pq}, $p, q \in \{-1, 0, 1\}$, $|p| + |q| = 1$ as

$$M_{ij}^{pq} = \left(\min\left(d_{i+p,j+q}^n - d_{ij}^n, 0\right)\right)^2$$

The Rouy-Tourin scheme for problem (1) then reads as follows

$$d_{ij}^{n+1} = d_{ij}^n + \tau_D - \frac{\tau_D}{h_D}\sqrt{\max\left(M_{ij}^{-1,0}, M_{ij}^{1,0}\right) + \max\left(M_{ij}^{0,-1}, M_{ij}^{0,1}\right)} \qquad (4)$$

This scheme is stable for $\tau_D \leq h_D/2$ and we take advantage of the fact, that it produces monotonically increasing updates that are gradually approaching a steady state. This property allows us to implement (4) in a computationally efficient way. Let us consider the index set $\mathcal{F}^n$ that contains the indices $\{i, j\}$ of the volumes where the steady state has been already reached, i.e. there exists such $n_0 \in N$, $n_0 \leq n$, that $d_{ij}^{n_0} = d_{ij}^{n_0-1}$. The set $\mathcal{F}^0$ is given as follows. At the beginning, we compute exact distances to the set Ω_0 (which is a set of points corresponding to cell centers or a set of curves representing the inner boundaries of cells) in a local (one pixel) neighborhood. Then $\mathcal{F}^0$ consists of the indices of all volumes with these exact values including the set Ω_0. Then the method is given by Algorithm 1.

Algorithm 1 Fixing method for distance function

- **if** $\{i, j\} \in \mathcal{F}^n$ **then** continue
- **else**
- $\quad d_{ij}^{n+1} = d_{ij}^n + \tau_D - \frac{\tau_D}{h_D}\sqrt{\max\left(M_{ij}^{-1,0}, M_{ij}^{1,0}\right) + \max\left(M_{ij}^{0,-1}, M_{ij}^{0,1}\right)}$
- $\quad$ **if** $d_{ij}^{n+1} = d_{ij}^n$ **then** $\mathcal{F}^{n+1} = \mathcal{F}^n \cup \{\{i, j\}\}$

3.2 Advective subjective surface method for detecting the intercellular skeleton

Now we discretize the equation (3). Again, we consider explicit time discretization with time step τ_S and the space grid elements are indetified with the pixels

of the image. The space discretization is based on the upwind principle. If the lentgh of the side of the pixel is denoted by h_S and we define the central differences $D^x_{ij}g = (g_{i+1,j} - g_{i-1,j})/(2h_S)$, $D^y_{ij}g = (g_{i,j+1} - g_{i,j-1})/(2h_S)$, we get the following approximation of (3)

$$u^{n+1}_{ij} = u^n_{ij} - \frac{\tau_S}{h_S}\left(\max\left(D^x_{ij}g, 0\right)(u_{ij} - u_{i-1,j}) + \min\left(D^x_{ij}g, 0\right)(u_{i+1,j} - u_{ij})\right.$$
$$\left. + \max\left(D^y_{ij}g, 0\right)(u_{ij} - u_{i,j-1}) + \min\left(D^y_{ij}g, 0\right)(u_{i,j+1} - u_{ij})\right) \quad (5)$$

As the initial condition we take a shock-like profile localized around the cell center. Due to the properties of the signed distance function computed by the method described in Sec. 3.1, we can see that the advective velocity ∇g drives all level lines of the initial segmentation function to the ridges of the distance function, cf. Fig. 6. These ridges represent the intercellular skeleton.

4 Numerical experiments

4.1 Computation of the signed distance function

Now let us present some computational results obtained by Algorithm 1. We inspected the experimental order of convergence of the suggested method, the CPU time needed for the computation and the effect of the fixing strategy. The results were also compared with the distance function computed analytically.

First, let us make a note about the stopping criteria for the methods. If the fixing technique is not applied, the computation is stopped either when $||d^{n+1} - d^n||_{L_1(\Omega)} \leq \varepsilon_1 << 1$ or when a prescribed number of time steps is performed. If we use the fixing strategy, we stop when $\{i, j\} \in \mathcal{F}^n$ for all $i = 1 \dots n_x$, $j = 1 \dots n_y$, i.e. when all values are already fixed. In practice, the condition $d^{n+1}_{ij} = d^n_{ij}$ is replaced by $|d^{n+1}_{ij} - d^n_{ij}| \leq \varepsilon_2 << 1$. If we want to compare the two methods, we first run the computation with fixing until all points are fixed and then the method with no fixing is prescribed to stop at exactly the same time.

In the first experiment, we computed the distance function to seven given points situated in a square domain $\Omega = [-1, 1] \times [-1, 1]$ and we measured the $L_2(\Omega)$ error with respect to the exact distance function and the EOC. The results and some details of the computations are displayed in Tab. 1, 2, 3 and 4.

Tab. 1 shows the results for the method without any fixing. The value in the seven points was set to 0, and the values in a one pixel neighborhood of the points were set to the values of the exact solution. Tab. 2 shows the results for the method with fixing, the solution was initialized in the same way. In the stopping criterion, we set $\varepsilon_2 = 10^{-5}$. Fig. 2 displays the distance function computed by the method using $n_x = n_y = 320$. Comparing the two tables, we can see the effect of the fixing strategy on the CPU time as well as on the L_2 error and EOC. We can observe that in this case the computation is approximately two times faster and also we get smaller error when we fix the solution by our algorithm. The fact that the L_2 error apparently depends on the value of ε_2 led us to perform

the experiment presented in Tab. 3. We were looking for the optimal choice of ε_2, i.e. the value that would provide the smallest L_2 error. We can see that this value is different for different discretization parameters. Finally, in Tab. 4, we present the EOC for a slightly different implementation of the fixing method. In this case, the solution was initialized with the exact values not only in a one pixel neighborhood, but in a neighborhood whose size was independent of the space step. Again, we were looking for the optimal value of ε_2. We can observe that the EOC is approximately equal to 1.

n_x	τ_D	time steps	$L_2(\Omega)$ - error	CPU	EOC
40	0.025	52	3.447525e-2	0.01	
80	0.0125	90	2.373985e-2	0.05	0.53825
160	0.00625	163	1.533921e-2	0.4	0.63009
320	0.003125	305	9.498908e-3	3.1	0.69139
640	0.0015625	583	5.704456e-3	23.18	0.73567

Table 1. Results of computational tests for the method without fixing. Computation of the distance function to seven given points.

n_x	τ_D	time steps	$L_2(\Omega)$ - error	CPU	EOC
40	0.025	52	3.440187e-2	0.0	
80	0.0125	90	2.359906e-2	0.03	0.54376
160	0.00625	163	1.508116e-2	0.21	0.64598
320	0.003125	305	9.016883e-3	1.73	0.74205
640	0.0015625	583	4.844598e-3	13.08	0.89625

Table 2. Results of computational tests for the fixing method. Computation of the distance function to seven given points.

The next experiment is similar. Instead of seven points, we computed the distance function to four polygons. The results are displayed in Tab. 5 and 6 and in Fig. 3.

The next two tests were realized in order to compare the fixing method with computation of the exact distance function. Again, we used $\varepsilon_2 = 10^{-5}$ for the fixing mehotd. In the first case, a certain number of points was randomly generated in a given 2D domain. After, the distance function to this set of points was computed both numerically by our fixing algorithm and analytically by finding the nearest point to every pixel. The CPU time was measured for different numbers of points and plotted in graphs presented in Fig. 4. We can observe that the computational cost of the analytical computation is increasing with increasing number of points while the cost of the numerical computation is

n_x	τ_D	time steps	$L_2(\Omega)$ - error	ε_2	CPU	EOC
40	0.025	37	1.976552e-2	3.4e-3	0.0	
80	0.0125	73	1.373675e-2	1.3e-3	1.02	0.52494
160	0.00625	145	9.133363e-3	4.0e-4	0.19	0.58882
320	0.003125	287	5.857149e-3	1.3e-4	1.62	0.64095
640	0.0015625	569	3.645805e-3	4.0e-5	12.77	0.68396
1280	0.00078125	1132	2.164545e-3	1.1e-5	100.09	0.75217

Table 3. Results of computational tests for the fixing method. Computation of the distance function to seven given points with determination of optimal ε_2.

n_x	τ_D	time steps	$L_2(\Omega)$ - error	ε_2	CPU	EOC
40	0.025	35	2.648673e-2	6.2e-3	0.0	
80	0.0125	71	1.352430e-2	1.5e-3	0.02	0.96972
160	0.00625	140	6.928000e-3	3.5e-4	0.19	0.96504
320	0.003125	276	3.598868e-3	9.0e-5	1.58	0.94490
640	0.0015625	545	1.795268e-3	2.2e-5	12.29	1.00033

Table 4. Experimental order of convergence for the fixing method with exact values prescribed in a fixed neighborhood of seven given points and with determination of optimal ε_2.

decreasing. For a certain number of points, depending on the size of the image domain, the numerical method becomes more efficient. We can see that in all cases displayed in the figure, this number is at most 1% of the number of image pixels so it is practically meaningful.

Another experiment was performed in order to test the numerical method on data qualitatively similar to segmented cell structures. The result of the cell segmentation is a level set function and the inner borders of the cells are represented by a chosen isoline of this function. For our test purposes, we used images with isolines in form of either randomly placed circles or randomly placed rectangles of a random size (see fig. 5). The maximum image intensity (255) was in the centers of these shapes and then it gradually decreased to 0 with increasing distance from the center. In order to compute the distance function to a certain level set numerically, the location of the level set was detected and the solution was initialized by exact values in its one pixel neighborhood. After, the fixing method was applied. For the exact computation, we used a simple procedure that constructs set of points corresponding to the chosen isoline, finding its crossections with the pixel structure. Then, the nearest of these points was found for each pixel. The CPU time for such a procedure is documented in Tab. 7, column 5. By construction, the points of the isoline are sorted by their coordinates and therefore the computation of the exact distance function can be significantly optimized – we do not have to go through the whole list of points but the nearest point to a pixel can be always found in a certain neighborhood of the nearest point to the previous pixel. The CPU times for this optimized approach are

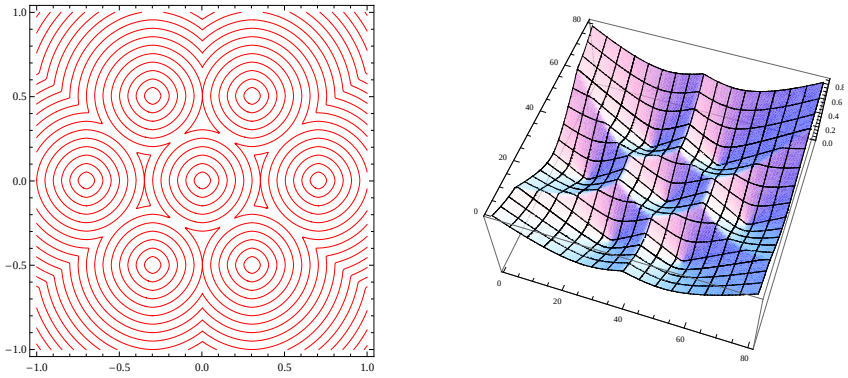

Fig. 2. Distance function to five given points computed by the fast fixing method. Left, the contours of the function. Right, the 3D plot.

n_x	τ_D	time steps	$L_2(\Omega)$ - error	CPU	EOC
40	0.025	36	1.122800e-2	0.01	
80	0.0125	61	8.266972e-3	0.03	0.44167
160	0.00625	108	5.606750e-3	0.26	0.56020
320	0.003125	203	3.599271e-3	2.04	0.63946
640	0.0015625	390	2.228162e-3	15.46	0.69185

Table 5. Results of computational tests for the method without fixing. Computation of the distance function to four polygons.

listed in Tab. 7, column 6. According to Tab. 7, we can observe that in all cases considered here, the numerical solution was faster than the optimized analytical computation. We considered isoline for value $I = 128$, $h_D = 1.0$, $\tau_D = 0.5$, $\varepsilon_2 = 10^{-5}$ and the dimensions of the image domain were 512×512 pixels.

In the last experiment, we computed numerically the distance function to a set of segmented cells of a zebrafish embryo. In fig. 6, we show the computed distance function as well as the vector field given by the gradient of this function. We used $h_D = 1.0$, $\tau_D = 0.5$ and $\varepsilon_2 = 10^{-5}$.

4.2 Extraction of the intercellular skeleton

In the first experiment, we computed the skeleton using the distance function corresponding to approximate cell centers. The result is a Voronoi type skeleton. As we can see in Fig. 7, this skeleton represents a very good approximation of the cell structure and it can provide reliable estimate of some quantities, like the area of the cell contact surface.

If we have a good quality segmentation of the inner boundaries of the cells, we can detect the intercellular skeleton more precisely using the signed distance function to the segmented objects. An example is shown in Fig. 8. In both experiments, we set $h_D = 1.0$, $\tau_D = 0.5$, $\varepsilon_2 = 10^{-5}$, $h_S = 1.0$, $\tau_S = 0.1$.

n_x	τ_D	time steps	$L_2(\Omega)$ - error	CPU	EOC
40	0.025	36	1.121228e-2	0.01	
80	0.0125	61	8.231755e-3	0.01	0.44581
160	0.00625	108	5.539007e-3	0.11	0.57157
320	0.003125	203	3.475747e-3	0.96	0.67230
640	0.0015625	390	2.041495e-3	7.24	0.76770

Table 6. Results of computational tests for the fixing method. Computation of the distance function to four polygons.

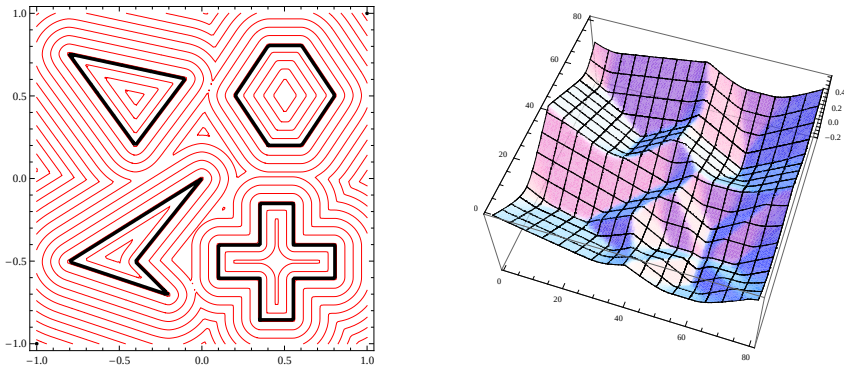

Fig. 3. Distance function to four polygons computed by the RMF. Left, the contours of the function with the shapes indicated. Right, the 3D plot.

Remark 1. In practice, the main determining factor of the quality of the detected skeleton is the correct orientation of the vector field generated by the gradient of the distance function. As it follows from the character of the problem and also from the numerical procedure, the correct orientation of the vector field can be already obtained as soon as the values in all image pixels are nontrivially updated by the numerical scheme (4), i.e. before they are definitely fixed. This allows a significant reduction of the computational time without loss of the quality of the final result. In Fig. 9, the result obtained after 10 steps of the numerical computation of the distance function is shown. At this moment, the values were nontrivially updated in a sufficiently large neighborhood of the segmented cells. Let us note that for the complete fixing of the values in the whole domain, we would need 123 time steps.

References

1. Frolkovič, P., Mikula, K., Peyriéras, N., Sarti, A.: A counting number of cells and cell segmentation using advection-diffusion equations. Kybernetika 43, No. 6(2007), 817–829.

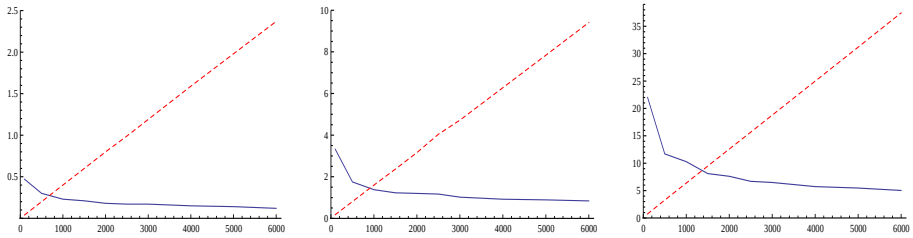

Fig. 4. Comparison of the fixing method (solid line) with computation of exact solution (dashed line)–plot of CPU time depending on the number of randomly distributed points. Left, image domain with 256×256 pixels. Middle, 512×512 pixels. Right, 1024×1024 pixels.

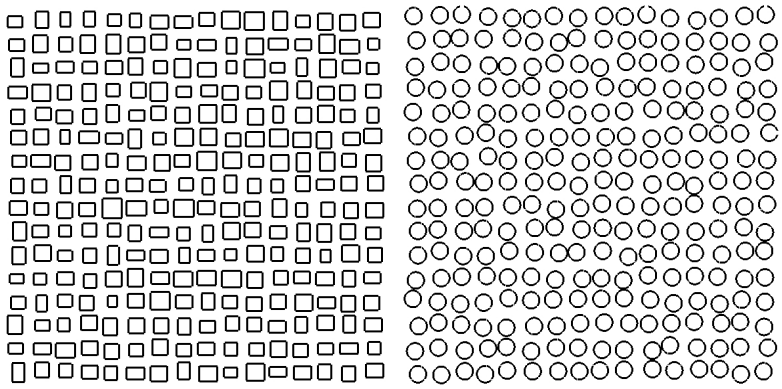

Fig. 5. Isolines ($I = 128$) of test level set functions simulating cell structures.

2. Mikula, K., Peyriéras, N., Remešíková, M., Sarti, A.: 3D embryogenesis image segmentation by the generalized subjective surface method using the finite volume technique. Proceedings of FVCA5 – 5th International Symposium on Finite Volumes for Complex Applications, Hermes Publ., Paris 2008.
3. Rouy, E., Tourin, A.: Viscosity solutions approach to shape-from-shading, SIAM Journal on Numerical Analysis 29 No. 3 (1992), 867–884
4. Sarti, A., Malladi, R., Sethian, J.A.: Subjective Surfaces: A Method for Completing Missing Boundaries. Proc. Nat. Acad. Sci. 12, No. 97 (2000), 6258–6263.
5. Sethian, J.A.: Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Material Science, Cambridge University Press, New York, 1999.
6. Tassy, O., Daian, F., Hudson, C., Bertrandt, V., Lemaire, P.: A quantitative approach to the study of the cell shapes and interactions during early chordate embryogenesis, Currrent Biology, 16 (2006), 345–358.
7. Zhao, H.-K., Fast sweeping method for eikonal equations, Mathematics of Computation, Vol. 74 (2005) 603–627
8. Zhao, H.-K., Osher, S., Fedkiw, R.: Fast surface reconstruction using the level set method, Proc. IEEE workshop on variational and level set methods - VLSM'01, Vancouer (2001) 194–201.

type of level set	time steps	$L_2(\Omega)$ - rel. error	CPU num.	CPU exact	CPU exact opt.
256 rectangles	52	3.509519e-2	0.91	35.56	2.87
1024 rectangles	33	5.813329e-2	0.63	75.82	2.88
256 circles	54	2.441320e-2	0.91	34.55	3.00
1024 circles	34	4.153412e-2	0.64	68.41	3.07

Table 7. Comparison of the numerical and analytical computation of the distance function to the cell-like structures.

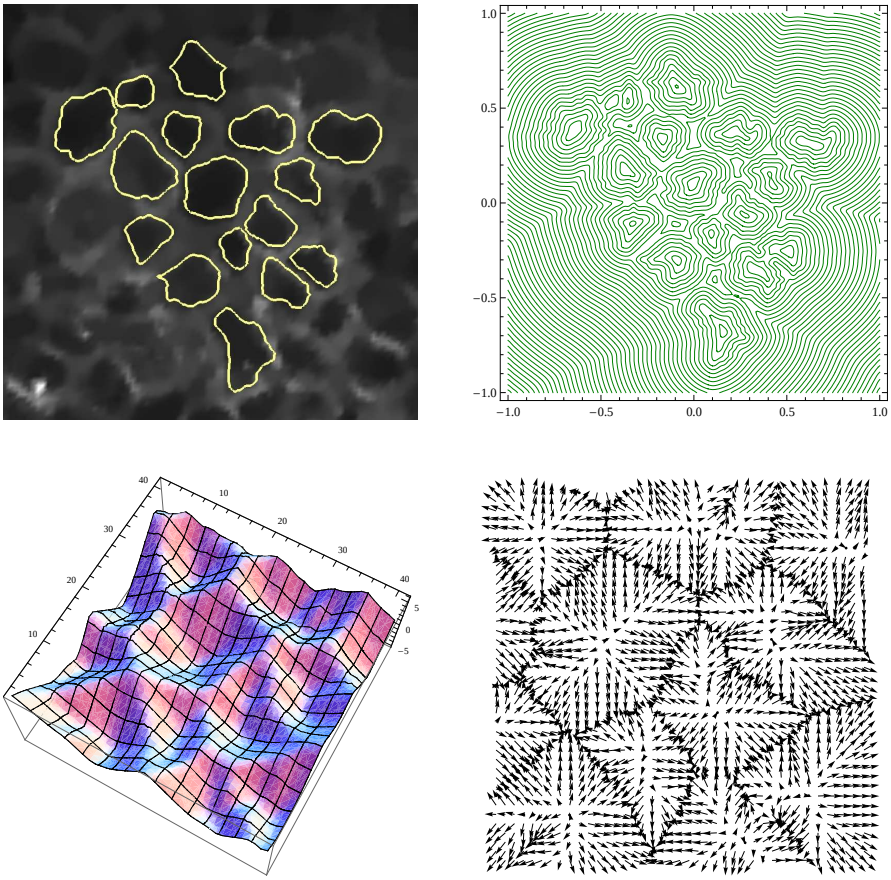

Fig. 6. Signed distance function to segmented cells. Top left, segmentation of inner boundaries of cells. Top right, contour plot of the signed distance function. Bottom left, detail of the 3D plot. Bottom right, detail of the vector field given by ∇g, $g(x) = d(x)$, with recognizable position of the intercellular skeleton.

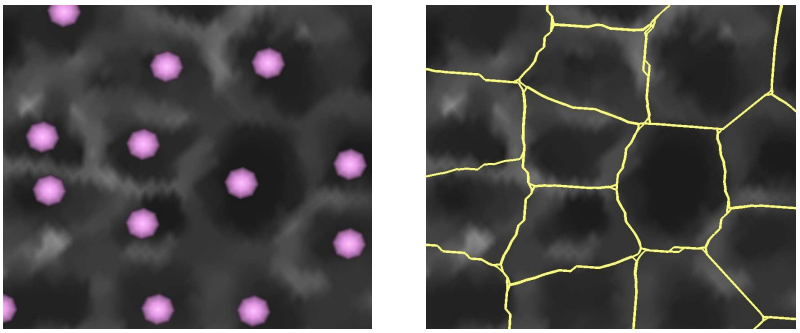

Fig. 7. Voronoi type skeleton. On the left, the approximate cell centers. On the right, the corresponding Voronoi type skeleton.

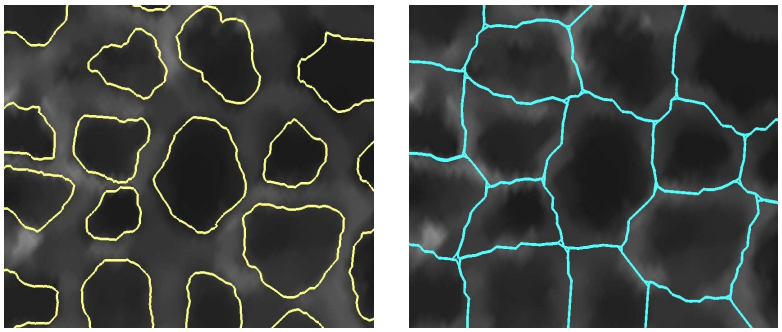

Fig. 8. Real skeleton detection. On the left, the cell segmentation. On the right, the corresponding intercellular skeleton.

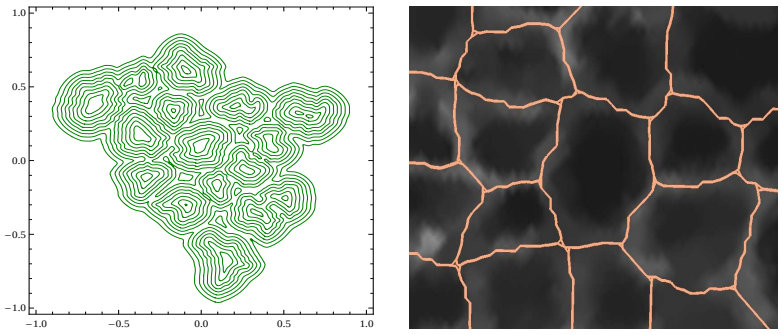

Fig. 9. Left, the result after 10 steps of computation of the distance function. Right, the skeleton obtained using the corresponding vector field.