Journal for Geometry and Graphics
Volume 18 (2014), No. 1, 105–114.

Polygonization of Line Skeletons by Using Equipotential Surfaces – A Practical Description

Franz Gruber, Günter Wallner

Department of Geometry, University of Applied Arts Vienna
Oskar Kokoschka Platz 2, A-1010 Vienna, Austria
email: guenter.wallner@uni-ak.ac.at

Abstract. Following the common concepts of polygonization of implicit surfaces we provide a modified discrete physically based method to derive polygonal meshes from line skeletons. A given line skeleton is considered to be electrically charged by placing a set of uniformly distributed point charges on it. Furthermore, we define an implicit iso-surface, representing the equipotential surface of this set of point charges, which encloses the line skeleton. Finally, with the help of a dynamic particle system a polygonal mesh is derived from this surface. The shape of the tubular cross section and the general cylindricality of the mesh can be controlled with several parameters. Moreover, the algorithm allows for sharp-edged borders.

Key Words: equipotential surface, line skeleton, organic structures, modeling
MSC 2010: 68U05, 00A66

1. Introduction

Surfaces around branching points of line skeletons with more than two branches are usually hard to model. Early methods (e.g., [7, 8]) which approximate the geometry with primitives like spheres, cylinders or cones do not properly reflect the *saddle* surfaces occurring in organic structures around such points. In special cases, explicit analytical formulas can be given to construct a blend surface between quadrics (e.g., [13]). For the general case implicit surfaces turned out to be well suitable for modeling of smooth organic structures (e.g., [6, 4]) independent from the complexity of the input line skeleton.

In this paper we present a discrete physically based method to convert a line skeleton into a tubular 2-manifold polygonal mesh by arranging particles on an equipotential surface induced by the electrically charged line skeleton. The resulting particle cloud is later used for triangulation of the surface. However, triangulation is not the focus of this paper. We show that the algorithm is stable, easy to implement, and is suitable to create a wide variety of complex organic meshes as results will show.

ISSN 1433-8157/$ 2.50

The remainder of this paper is structured as follows. Some related work is reviewed in Section 2. In Section 3 the algorithm and its parameters are described in detail. Results obtained with the algorithm are shown in Section 4 before the paper is concluded in Section 5.

2. Related work

Smooth branching structures are frequently required in computer graphics for the modeling of organic shapes like trees or bushes. Bloomenthal [1] was among the first to improve the realism of branching structures by using generalized cylinders for limbs and by using free-form surfaces to connect branching limbs. However, as pointed out by Galbraith et al. [4] his method can not be easily extended to arbitrary branching structures. Galbraith et al. on the other hand introduced a hierarchical implicit modeling system, called Blob-Tree, which combines several techniques to allow for smooth and non-smooth blending at the joints. Tobler et al. [12] used a generalized subdivision scheme, which in contrast to standard subdivision allows different subdivision rules at different levels, and combined it with a rule-based mesh growing procedure. By alternating between subdivision and mesh growing, complex geometry (including structures such as forking trunks) can be generated. Recently, Zhongke et al. [15] used so-called ball b-spline curves to generate stems of a tree. However, as the authors point out, the individual branches are only put together without any blending operations. A good overview of tree modeling is given in the survey of Soner and Day [11] and the more recent paper of Boudon et al. [2].

Two works which are similar to ours, are the papers of Jin et al. [6] and de Figueiredo et al. [3]. Jin et al. [6] generated a convolution surface based on line-segment skeletons with a variable kernel modulated by a polynomial. In contrast to the discrete approach presented in this paper, they calculated an analytical solution for a specific case of the field function. However, their approach does not allow for sharp boundaries. De Figueiredo et al. [3] described two discrete physically based methods to approximate implicit surfaces with a polygonal mesh. In order to sample the implicit surface, the first method uses a particle system whose dynamics are driven by the potential function. Their second approach is based on a mass-spring system — based on a Freudenthal triangulation — which seeks equilibrium on the implicit surface. There is a huge amount of literature on implicit surface sampling, see, for example, [9, 14].

3. Algorithm

In this section we first give an overview of the algorithm and then describe the individual steps in detail.

3.1. Overview

The input of the algorithm is a line skeleton defined as an undirected graph $G = (V, E)$ with a set of vertices V and a set of edges E. In this paper edges are represented as straight lines. However, the following description is similarly applicable if edges are represented as arbitrary curves like, for example, b-splines. In a first step, a set of positive charges $C = \{c_1, \ldots, c_n\}$ with charge values $c_i^q > 0$ are evenly placed at static positions[1] $c_i^{pos} \in \mathbb{R}^3$ on E.

[1] We will use the superscript *pos* to refer to the position of different elements.

These charges induce an electrostatic field $\mathcal{E} \backsim 1/r^{\alpha}, \alpha \in \mathbb{R}, \alpha \geq 2$. In case of Coulomb's law $\alpha = 2$, but we allow for larger values to control the cylindricality of the resulting surface. The equipotential surface of C corresponding to potential value λ can then be written implicitly as

$$S_\lambda = \left\{ \vec{r} = (x, y, z) \in \mathbb{R}^3 \;\middle|\; \sum_{i=1}^{n} \frac{c_i^q}{\|\vec{r} - c_i^{pos}\|^{\alpha-1}} - \lambda = 0 \right\} \tag{1}$$

While λ influences the diameter of the tubular cross section globally (henceforth referred to as *global diameter*), the individual charges c_i^q influence the diameter locally. From S_λ a polygonal mesh M is derived. For this purpose, S_λ is roughly approximated by geometric primitives (spheres and cylinders) on which an initial particle cloud — consisting of positive unit charges — is defined. Afterward, the particles are iteratively moved along the field lines of $\mathcal{E}$ towards S_λ. The particle cloud is later used for triangulation.

3.2. Initial placement of charges

In a first step positive charges are placed at the positions of the vertices V and as evenly as possible along their connecting edges as shown in Figure 1. At this point it should be noted that if we refer to the charges of an edge $e = (v_0, v_1)$, this does not include the charges located directly at v_0 and v_1 but only the charges between them. The intended distance between two neighboring charges is specified by the parameter d_c. Clearly, the positions and values of the charges influence the shape of S_λ and hence also the shape around the leaf vertices. A way to allow for sharped edged surface boundaries (v_0 and v_3 in Figure 1) is to place additional charges on the extension of the edge of the leaf vertex. This ensures the cylindricity of S_λ around the leaf vertex. Otherwise the shape of S_λ will be spherical at endings, as it is the case for v_5 in Figure 1.

Generally, the choice of the charge values for each charge is arbitrary but for practical reasons we use — in the following — the same charge value for all charges on the same edge. That way each edge e is associated with a certain charge value e^q.

e^q is a measure of the average diameter e^d of the tubular surface around e. It would be convenient for the user to directly define e^d for each edge but analytically determining the suitable charge values to match these diameters in advance is a hard task because changing a single charge value influences the shape of S_λ globally. This bears comparison with calibration of a complex interdependent system.

For a smooth blend surface around a vertex v whose incident edges have different charge

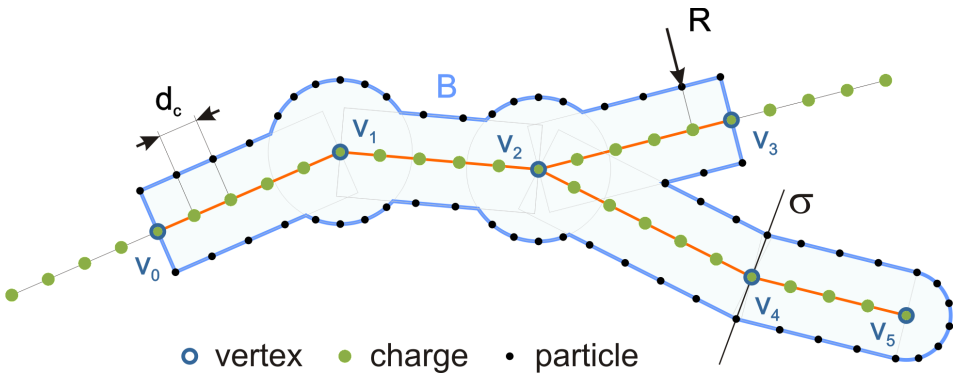

Figure 1: Initial placement of charges and particles (illustrated in 2D for reasons of simplicity)

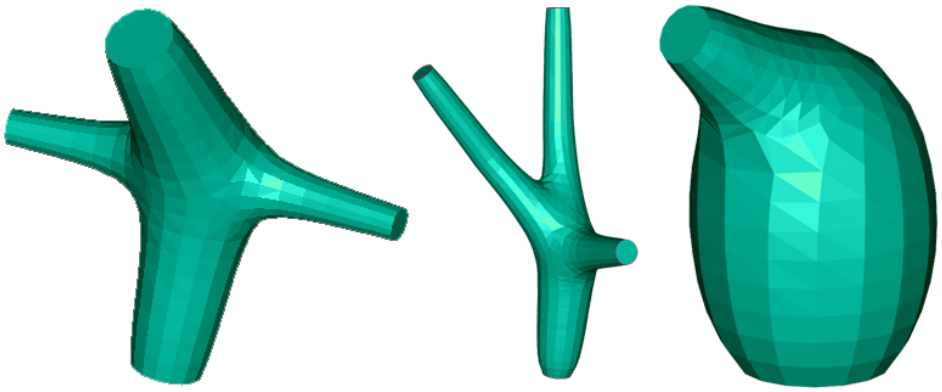

Figure 2: Examples to illustrate the influence of the charge values to the blend surface around branching points.

values, the charge value of the charge at v is the arithmetic mean of the charge values of all incident edges. Figure 2 gives examples for blending surfaces around such branching vertices. Each case in Figure 2 only uses two distinct charge values.

3.3. Initial placement of particles

Since the particles are later iteratively moved toward S_λ it is reasonable to initially place them evenly on a simple approximation of S_λ. The better the initial placement of the particles, the better the resulting distribution on S_λ. For this reason the particles are placed on spheres and cylinders around the vertices and edges of G. The initial radius R of the cylinders is a user-definable parameter. At this point we should note that R is the radius used for placement of the particles and does not correspond to the final radius of the cross-section, which is influenced locally by the different charge values c_i^q and globally by λ as well as α. The reason for using a constant radius is to simplify the placement of the particles. The particles located on spheres are used to evenly cover the area of the blend surface between adjacent cylinders. To ensure an *organic* blend surface, the radius of the sphere has to be greater than R. For the examples in this paper we defined the sphere radius as $1.5R$.

For the placement of particles several cases are distinguished. For charges between vertices and in case of charges at leaf vertices (v_0, v_3, v_5 in Figure 1), n_{circ} particles are placed circularly around their edge. If the ending at a leaf vertex should be closed (v_5 in Figure 1) additional particles are placed on a half-sphere around such a vertex. In case of a charge at vertex v with two incident edges e_a and e_b which enclose a rather obtuse angle $\angle(e_a, e_b) > \gamma$ (as it is the case for v_4 in Figure 1) the symmetry plane σ of e_a and e_b is calculated. In practice, a value of $\gamma = 160°$ turned out to be reasonable. The particles are then placed on a circle lying in σ with center v. For any other charge at vertex w, n_{sphere} particles are placed as evenly as possible on a sphere ($r = 1.5R$) centered at w (see v_1 and v_2 in Figure 1). However, for the purpose of triangulation it is beneficial that the initial particle cloud is topologically similar to the resulting 2d-manifold and therefore all particles which are inside the Boolean union B of all cylinders and spheres (cf. Figure 1) are removed. In other words, only particles on the surface of B will be used. Figure 3 (left) shows the initial placement of particles for a simple 3D example. Furthermore, to speed up calculation time later in the algorithm a set of nearest charges N_p is initialized for every particle.

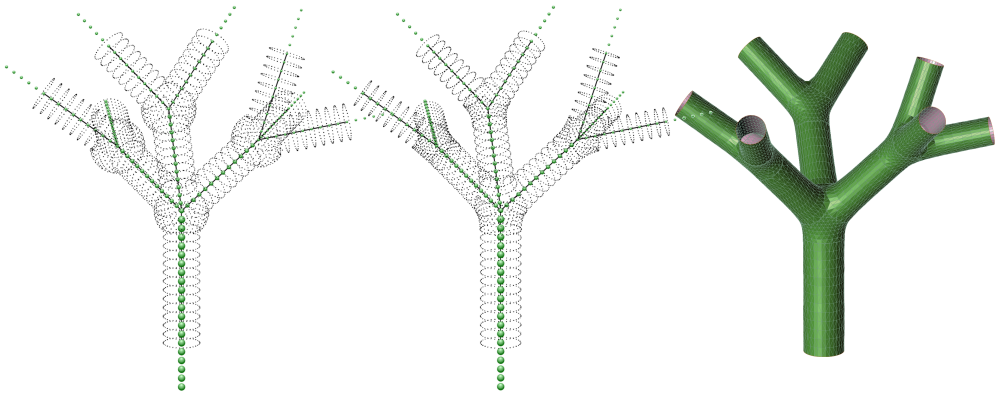

Figure 3: An example with different tube diameters. Charges are shown as green spheres on a line skeleton of a tree, whose radius corresponds to their charge value. Particles are depicted as black dots. Left: Initial placement of the particles. Middle: Arrangement of the particles after the algorithm has terminated. Right: Triangulation of the particle cloud.

3.4. Estimation of λ

The implicit surface S_λ as given in Eq. (1) is defined by a set of charges, the field exponent α, and a specific iso-value λ. In general, $\lambda \in [0, \infty]$ can be chosen arbitrarily but is not necessarily an intuitive parameter. An inappropriate value for λ can result in an iso-surface which does not reflect the branching structure properly.

Since the values of the charges located on an edge correspond to the local diameter of the cross section of the tubular mesh they can be used to estimate a reasonable value for λ. For that purpose we place four test particles evenly on a circle around each edge. Each circle is centered at the midpoint of its edge and its radius equals the charge value e^q associated with the edge (see Section 3.2). This yields a set of test particles T. λ is then given by the average potential of all test particles, mathematically

$$\lambda = \frac{1}{4|E|} \sum_{t \in T} \sum_{i=1}^{n} \frac{c_i^q}{\|t^{\,pos} - c_i^{pos}\|^{\alpha-1}} \,. \tag{2}$$

Figure 4 illustrates the final particle cloud for different values of λ for a simple line skeleton. As can be seen from the figure, varying λ does not change the characteristic of the blend surface around branching points but rather the global diameter of the tube system. In contrast to λ, a variation of the field exponent α changes the shape of the blend surfaces with respect to smoothness and roundness. In general, the higher the value of α the more local the influence of a point charge and therefore the *sharper* the blending. In other words, α allows the user to change the characteristic of the surface between organic and cylindrically looking. Figure 5 illustrates this circumstance.

3.5. Arranging particles on S_λ

Once all charges are defined and all particles have been placed at their initial position, the particles are iteratively moved along the field lines of $\mathcal{E}$ towards S_λ. For this purpose, each particle is considered to be a positive electrical unit charge. This induces a repulsive electrostatic force (in case of $\alpha = 2$ the Coulomb force) between a particle p and the charges C.[2]

[2]For reasons of computational efficiency, repulsive forces between particles are not taken into account.

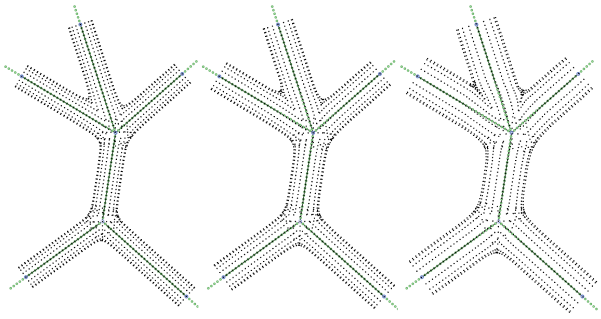

Figure 4: The parameter λ influences the global diameter of the resulting particle cloud. From left to right: estimated λ, 0.5λ, 0.2λ; $\alpha = 4$ in each case.

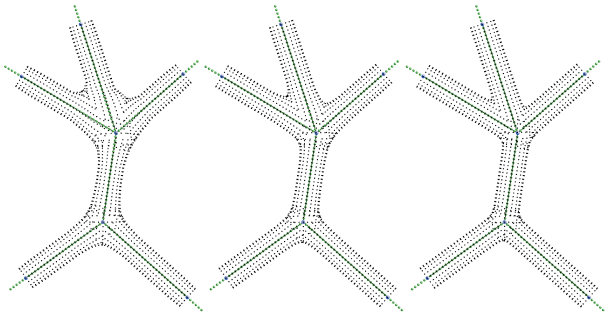

Figure 5: The exponent α of the field function influences the shape of the resulting particle cloud. Lower values of α yield smoother blend surfaces around branching points, whereas higher values increase the cylindricality and sharpness. From left to right: $\alpha = 2$, $\alpha = 3$ and $\alpha = 5$.

However, for an efficient approximation of the force only the set of nearest charges N_p of p is considered (cf. Section 3.3). The force direction is therefore given by

$$forceDir\,(p) = \sum_{c_k \in N_p} \frac{c_k^q}{\|\vec{r_k}\|^\alpha}\, \vec{r_k}^{(0)}, \tag{3}$$

where $\vec{r_k}^{(0)}$ denotes the unit vector of $\vec{r_k} = p^{pos} - c_k^{pos}$. Similarly, the electric potential of p can be approximated by

$$potential\,(p) = \sum_{c_k \in N_p} \frac{c_k^q}{\|\vec{r_k}\|^{\alpha-1}}. \tag{4}$$

If $potential\,(p) > \lambda$ then p is moved a small distance δ in force direction, otherwise p is moved a small distance δ in the opposite direction. This happens iteratively until $|potential\,(p) - \lambda| \leq \delta$. Once this condition is fulfilled for every particle the algorithm terminates.

Listing 1 summarizes the steps described in Section 3.2 to Section 3.5, and Table 1 provides an overview of the various parameters discussed. Figure 3 (middle) shows the final position of the particles for the simple 3D example.

```
setCharges();          // Section 3.2
setParticles();        // Section 3.3
estimate λ;            // Section 3.4

do {                   // Section 3.5
  for(i=0;i<nParticles;i++) {
    Vector3D dir = forceDir(p_i);
    normalize(dir);
    float dp = potential(p_i) - λ;
    p_i^pos += sgn(dp)*δ*dir;
  }
} while (breakCondition == false);
```

Listing 1: Pseudocode to derive the particle cloud representing the implicit surface.

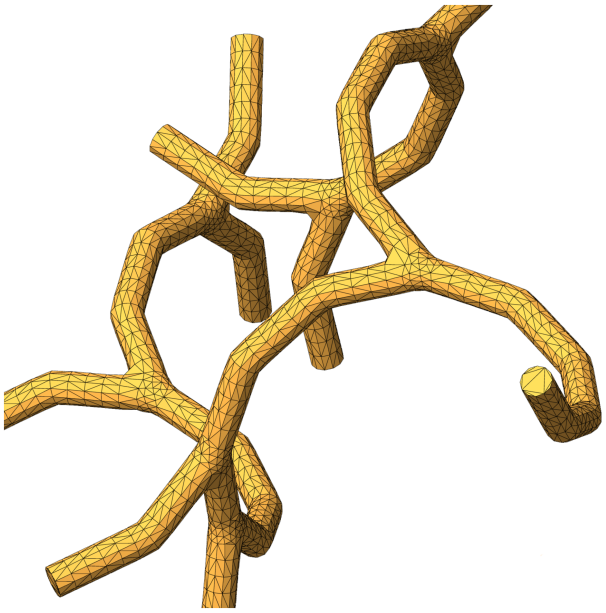

Figure 6: A branching system based on a line skeleton generated with a L-system. The triangulation is also shown (4068 particles, 8132 triangles, orthographic projection, flat shaded).

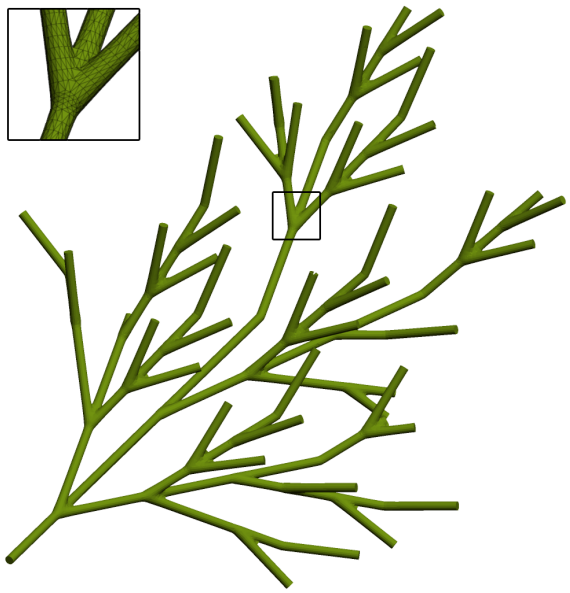

Figure 7: A more complex branch generated by a L-system and a close-up of a *3-way* branch (24369 particles, 48700 triangles, orthographic projection, flat shaded).

Table 1: Parameters of the algorithm

Parameter	**Description**
n_{circ}	Number of particles placed circularly around a charge
n_{sphere}	Number of particles placed spherically around a charge
R	Initial radius of the tubular cross section
d_c	Distance between the individual charges
λ	Potential value of the resulting equipotential surface
α	Exponent of the field function
c_i^q	Individual charge values to control radius locally

3.6. Triangulation

The particle cloud as arranged by the above algorithm is used to derive the polygonal mesh M. However, triangulation of point clouds is a complex topic and is out of scope of this paper. Triangulation of unordered point clouds can, for example, be performed with the open-source Point Cloud Library[3].

4. Results

Figure 6 through Figure 10 show results obtained with the described algorithm. All examples in these figures were produced by using the same charge value for all charges.

[3] www.pointclouds.org

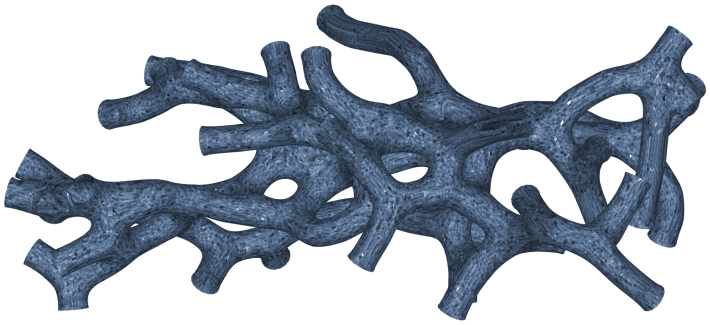

Figure 8: A textured organic structure based on Voronoi-Paths [5].

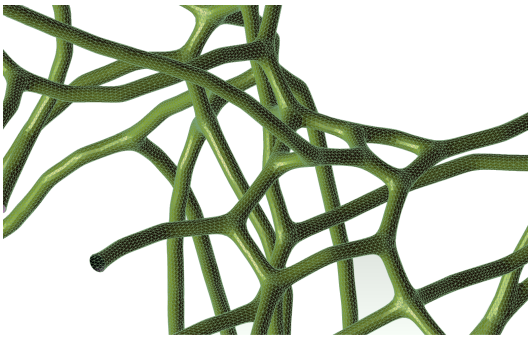

Figure 9: Part of a surface based on a line skeleton generated by the Voronoi-Path algorithm.

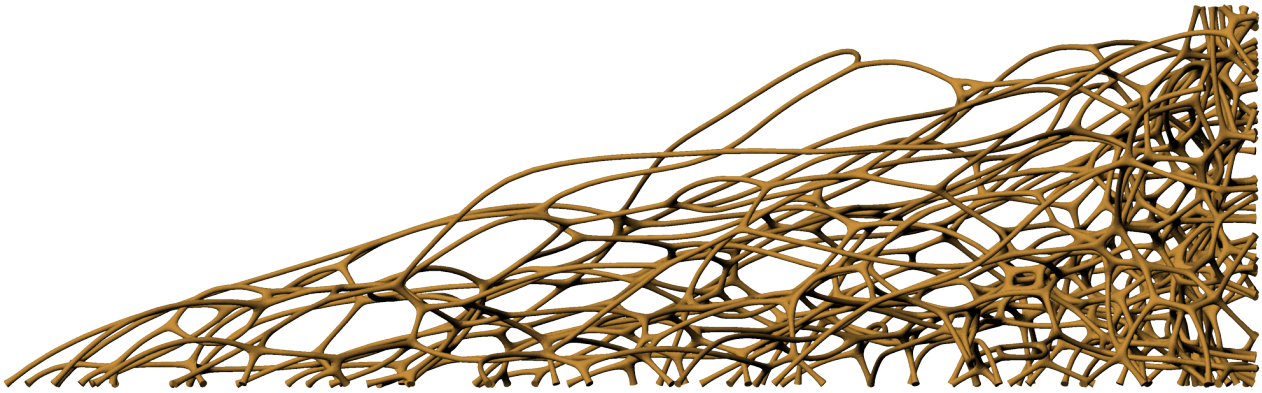

Figure 10: A complex organic structure. The triangulation of the point cloud has been smoothed in a post-processing step (235175 particles, 471374 triangles, orthographic projection, smooth shaded).

The line-skeletons for Figures 6 and 7 have been generated with *L-systems*[4]. In contrast, the line-skeletons for Figures 8, 9 and 10 have been generated with the *Voronoi-Path algorithm* described in [5].

For Figures 6 to 8, the triangulation was performed directly on the point cloud as calculated and no manual post-processing of the triangulation was conducted. For the examples in Figures 9 and 10 the triangulation has been smoothed.

5. Conclusions

Although the task of generating organic looking shapes seems quite complex the algorithm presented above provides a relatively simple solution that delivers satisfying results. The algorithm has proven to be stable even for large line skeletons. Generally, there are no restrictions for the input skeleton. By choosing a proper initial placement which topologically corresponds to the intended 2-manifold polygonal mesh, repulsion between particles can be omitted. However, line skeletons with quite acute angles between adjacent edges can cause clusters of particles in the resulting particle cloud which makes triangulation more difficult.

[4]L-systems were invented by Aristid Lindenmayer as a mathematical theory to model the growth of plants. A good introduction to L-systems can be found in [10].

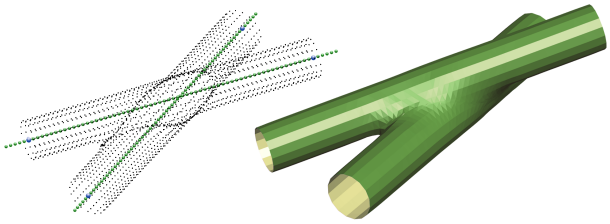

Figure 11: If two edges of the line skeleton are near to each other, the iso-surface is the fusion of the initial cylindrical particle clouds.

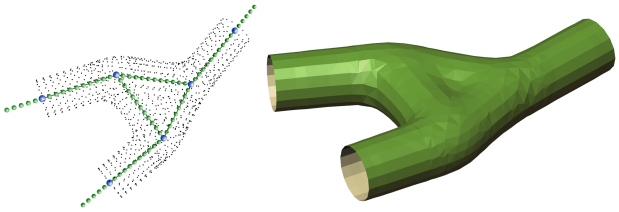

Figure 12: If vertices are close to each other then the connecting cylinders will fuse into a single iso-surface with characteristic *dimples*.

In such a case repulsive forces between particles should be taken into account to ensure a rather uniform distribution of the particles.

The essential parameters to control the shape of the resulting surface are the individual charge values c_i^q, the iso-parameter λ and the field exponent α. The values c_i^q influence to local diameter and allow to design bulges or to fuse surface parts together (see Figures 11 and 12 for examples). In contrast, λ influences the global diameter of the tubular cross section and α controls the smoothness of the blend surfaces around branching points. Together, all three parameters allow to generate a wide variety of organic shapes.

The current implementation uses a quite large number of particles because particles are also initially placed in small distances between the vertices of G. This ensures a good approximation of the equipotential surface and allows for better post-processing of the triangulated mesh. Alternatively, the number of particles could be reduced considerably by only calculating areas around branching points with the proposed algorithm and by approximating edges with cylinders or truncated cones. Finally, it can be noted that the input needs not necessarily be restricted to line skeletons but could also be extended with charges placed on surfaces.

References

[1] J. Bloomenthal: *Modeling the mighty maple*. Proc. 12th annual conference on Computer graphics and interactive techniques, SIGGRAPH '85, ACM, New York 1985, pp. 305–311.

[2] F. Boudon, A. Meyer, C. Godin: *Survey on computer representations of trees for realistic and efficient rendering*. Tech. Rep. RR-LIRIS-2006-003, 2006.

[3] L.H. de Figueiredo, J.d.M. Gomes, D. Terzopoulos, L. Velho: *Physically-based methods for polygonization of implicit surfaces*. Proc. conference on Graphics interface '92, Morgan Kaufmann Publishers Inc., San Francisco 1992, pp. 250–257.

[4] C. Galbraith, P. MacMurchy, B. Wyvill: *Blobtree trees*. Proc. Computer Graphics International, IEEE Computer Society, Washington DC, 2004, pp. 78–85.

[5] F. Gruber, G. Wallner: *Algorithms for generation of irregular space frame structures*. J. Geometry Graphics **15**, 169–179 (2011).

[6] X. Jin, C.-L. Tai, J. Feng, Q. Peng: *Convolution surfaces for line skeletons with polynomial weight distributions*. J. Graph. Tools **6**, 17–28 (2002).

[7] Y. Kawaguchi: *A morphological study of the form of nature*. Proc. 9th annual conference on Computer graphics and interactive techniques, SIGGRAPH '82, ACM, New York 1982, pp. 223–232.

[8] N. Max: *Cone-spheres*. Proc. 17th annual conference on Computer graphics and interactive techniques, SIGGRAPH '90, ACM, New York 1990, pp. 59–62.

[9] M.D. Meyer, P. Georgel, R.T. Whitaker: *Robust particle systems for curvature dependent sampling of implicit surfaces*. Proc. International Conference on Shape Modeling and Applications 2005, IEEE Computer Society, Washington DC, 2005, pp. 124–133.

[10] P. Prusinkiewicz, A. Lindenmayer: *The algorithmic beauty of plants*. Springer-Verlag, New York 1996.

[11] S.I. Sen, A.M. Day: *Modelling trees and their interaction with the environment: A survey*. Computers & Graphics **29**(5), 805–817 (2005).

[12] R.F. Tobler, S. Maierhofer, A. Wilkie: *Mesh-based parametrized l-systems and generalized subdivision for generating complex geometry*. Int. J. Shape Model. **8**(2), 173–191 (2002).

[13] J. Wallner, H. Pottmann: *Rational blending surfaces between quadrics*. Comput. Aided Geom. Design **14**, 407–419 (1997).

[14] A.P. Witkin, P.S. Heckbert: *Using particles to sample and control implicit surfaces*. Proc. 21st annual conference on Computer graphics and interactive techniques, SIGGRAPH '94, ACM, New York 1994, pp. 269–277.

[15] W. Zhongke, Z. Mingquan, W. Xingce: *Interactive modeling of 3d tree with ball b-spline curves*. Int. J. Virtual Reality **8**, 101–107 (2009).

Received August 2, 2012; final form March 26, 2014