A High-Performance Parallel Thinning Approach Using a Non-cubic Grid Structure

David Brunner, Guido Brunnett
Chemnitz University of Technology, Germany
{brunner, brunnett}@informatik.tu-chemnitz.de

Robin Strand
Uppsala University, Sweden
robin@cb.uu.se

Abstract

In the past years the so-called body-centered cubic grid (bcc) has been examined and proved to be superior over Cartesian lattices for certain applications. Our work deals with parallel thinning on these bcc grids. We introduce conditions which are sufficient for retaining topology and suggest additional conditions to influence the shape of the resulting skeleton. We further developed an algorithm to extract curve skeletons out of 3d objects in parallel which we also present here.

We show in our results that the developed thinning approach on bcc grids is extremely efficient.

1 Introduction

Recent years have shown that alternative representation schemes of 3D objects afford new applications. One important and well researched scheme is the medial axis of an object, often referred to as skeleton. For objects in 3D two kinds of medial axes are distinguished: surface skeletons and curve skeletons respective. For medial axis transformations (or synonymous called skeletonization) numerous applications exist. In medical field it is applied for instance to the virtual navigation and virtual endoscopy respectively [26]. In traditional computer graphics, skeletons are used to specify animation [5]. The skeleton bones control the behavior of the polygonal representation. In the field of geometric modeling skeletons are utilized for decomposing polygonal meshes into components [8], surface reconstruction [1] and mesh repair [16]. Other applications are feature tracking [25] or matching of 3D objects [7]. A very detailed survey of common applications and properties of the curve-skeleton is given by Cornea et al. [10]. Most of these applications require a previous transformation of the polygonal surface mesh into a 3D sampling lattice. This discretization step is extensively researched in [27] especially on non-cubic lattices.

Usually, this definition for a medial axis is used: Let $X \subset \mathbb{R}^3$ be a 3D object. A sphere of radius r centered at $x \in X$ is defined as $S_r(x) = \{y \in \mathbb{R}^3 : \|x, y\| \leq r\}$. A sphere $S_r(x) \subset X$ is maximal if it is not completely included in any other sphere included in X. The medial axis is the set of the centers of all maximal spheres included in X.

In continuous space the medial axis transformation is extremely time-consuming. Therefore, other, mostly approximative methods have prevailed. Following Cornea et al. [10],

skeletonization algorithms are categorized with respect to the underlying implementation into thinning and boundary propagation (e.g. [2]), distance field based (e.g. [26]), geometric (e.g. [6]) and general-field functions (e.g. [9]).

Considering approximative approaches only thinning algorithms preserve the objects topology. This condition is fulfilled, if the 3D object and its skeleton have the same number of connected components, tunnels and cavities [15]. In the stricter sense a curve skeleton cannot preserve topology since it cannot have cavities. Thus, the definition is relaxed [10]: the curve skeleton preserves the topology of the original object if it has the same number of connected components and at least as many loops as tunnels and cavities in the original object. For our examination we assume no cavities are included.

Aside from the topology, other properties of the skeleton are more or less important depending on the particular application. With our approach we meet the following properties. (1) Homotopy: topology preserving, (2) Thinness: curve skeletons are one-dimensional except at joints to ensure connectivity, (3) Centeredness: The curve skeleton lies on the medial axis, (4) Robustness: the skeleton of an object and its duplicate with added noise are the same, (5) Efficiency: the processing is fast. A sixth property is the ability to reconstruct the original object out of the skeleton. We do not meet this property because thinness and reconstruction are conflicting properties. But we discuss which criteria are suitable for reconstruction tasks.

Almost all previous contributions on thinning have been based on the Cartesian lattice although it is not well suited to defining a correct discrete topology [14], because they are de-facto standard for regular representations of volumetric data. Cartesian lattices are used to define volume objects consisting of cubical voxels (volume elements). This paper will extend thinning algorithms to optimal regular lattices, specifically to body-centered cubic lattices (bcc) which introduces 14-sided voxels. These lattices have recently emerged in the field of volume graphics and show great promise in storing volumetric data more efficiently [17]. Also these lattices are superior over Cartesian lattices due to their efficient distribution of sample points. And for instance they are used to speed up traditional volume rendering algorithms [24]. Other positive features are caused by the advantageous neighborhood. Examining small voxel neighborhoods, there are only 14 neighbors which have to be explored per voxel during the thinning process. Furthermore, no topological paradoxes such as objects being both connected and disconnected [15] can occur since a volume element in bcc lattices has just face-neighbours (Fig. 1). Hamitouche et al. state in [14] that bcc lattices induces the simplest discrete topology definition.

A first attempt for a thinning algorithm on bcc lattices to extract surface skeletons was introduced by Strand [18]. In this paper we present a thinning approach for surface and curve skeletons, whereas the main focus lies on the curve skeletons because they are relevant for the most of the mentioned applications. After basic definitions in section 2, we describe in section 3 a thinning approach on bcc lattices. Further, we show how this approach can be parallelized to increase processing speed. The motivation is the current development like dual core processors or GPU-based processing. We also look ahead to Cell processors for multiple processing cores. In the following section we suggest additional criteria to vary the properties of the emerged curve skeletons according to the

requirements of the particular application. The concrete implementation of the suggested approach is introduced in section 5. The results conclude this paper.

2 Basic definitions and notations

We use a particular point lattice, the so-called bcc grid, as the mathematical model for describing a tessellation of the Euclidean 3D space $\mathbb{R}^3$ into volumetric elements. The coordinates of a grid point $p \in \mathbb{Z}^3$ are denoted by $p^i, i \in 1,2,3$. Then the bcc grid $\mathcal{B}$ is defined as the subset of $\mathbb{Z}^3$ that consists of all points having either only even or only odd coordinates:

$$\mathcal{B} = \{p : p^1 \equiv p^2 \equiv p^3 \pmod 2\}.$$

The volumetric elements used to tessellate $\mathbb{E}^3$ are defined as the Voronoi cells of the lattice points. For each $p \in \mathcal{B}$ we associate a Voronoi cell

$$v(p) = \{e \in \mathbb{E}^3 : d_e(p,e) \leq d_e(q,e) \text{ for } q \in \mathcal{B} \text{ and } p \neq q\}.$$

in which d_e is Euclidean distance. These cells are truncated octahedrons (Fig. 1(a)) and are used instead of grid points in the figures.

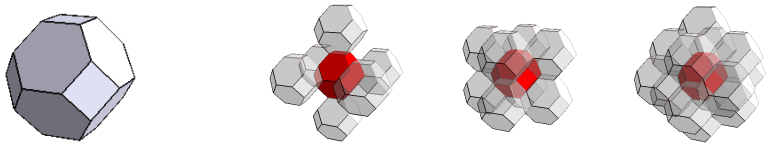
In the lattice we consider three sets of directions:

$$\begin{aligned} D_6(p) &= \{d \in \mathbb{Z}^3 : |d^1| + |d^2| + |d^3| = 2\}, \\ D_8(p) &= \{d \in \mathbb{Z}^3 : |d^1| + |d^2| + |d^3| = 3\}, \\ D_{14}(p) &= D_6(p) \cup D_8(p). \end{aligned}$$

Based on these directions three kinds of *neighborhoods* are defined (Fig. 1). For $p \in \mathcal{B}$ and $\alpha = 6, 8, 14$ then:

$$N_\alpha(p) = \{q \in \mathcal{B} : p - q \in D_\alpha\}.$$

Note, that the point p itself does not belong to the neighborhood.



(a) (b) (c) (d)

Figure 1: A Voronoi cell of the bcc lattice (a) and its N_6- (b), N_8- (c) and $N_{14}-$neighborhood (d).

A bcc-discretized object is defined by the set of bcc grid points $O \subset \mathcal{B}$. Using the notion of a binary image it is common to say that each point of O belongs to the foreground while the points of $\overline{O} = \mathcal{B} \setminus O$ form the background.

The number of foreground neighbors for $p \in O$ is $|N_{14}(p) \cap O|$. To describe important special cases in the thinning process we call $p \in O$

an *isolated point* iff $|N_{14}(p) \cap O| = 0$,
an *end point* iff $|N_{14}(p) \cap O| = 1$,
a *curve point* iff $|N_{14}(p) \cap O| = 2$.

Since we use a directional thinning method we define $p \in O$ to be a *directed border point* of direction $d \in D_{14}$ if $p - d$ belongs to the background:

$$p - d \in N_{14}(p) \cap \overline{O}.$$

A *sequel point* f of $p \in O$ in direction d is the point $f = p + d$. We call a point pair (f, q) with $f, q \in N_{14}(p)$ that satisfies $f - p = p - q$ opposite points (see Fig. 2 (a), (b)).

A point set $P \subset Q \subset \mathcal{B}$ is a connected component of Q, if for any two points $p, q \in P$, a sequence of points $p_1, ..., p_m \in P$ exists such that $p_1 = p$, $p_m = q$ and $p_{i+1} \in N_{14}(p_i)$ for $1 \leq i < m$. A connected component $P \subset Q \subset \mathcal{B}$ is maximal if there is no point r in Q such $P \cup \{r\}$ is a connected component of Q. $C(Q)$ denotes the number of maximal connected components of Q. We distinguish the number of maximal connected foreground components of $Q \subset \mathcal{B}$:

$$C(Q \cap O)$$

and the number of maximal connected background components:

$$C(Q \cap \overline{O}).$$

3 Parallel Thinning on body-centered cubic lattices

The term thinning is used for a process during which cells are removed from an object that is represented as a cell complex. Since the cells are uniquely associated with points in discrete space $\mathbb{Z}^3$ this process can also be formulated in terms of point removal.

In this section we will present the basic principles of a thinning method that produces a one dimensional skeleton from a given discrete object $O \subset \mathcal{B}$. We will formulate conditions that ensure topological equivalence of the skeleton with the original object both for sequential and parallel removal of points.

It is common to say that a point is simple if its removal does not change the topology of O.

In the literature different criteria for topological invariant point removal have been suggested (see [3, 4, 28]). Note, that some of the criteria refer to a relaxed definition of topological invariance that only takes the number of connected components into account.

In [3] Bertrand and Malandain proved that conditions for the simplicity of a point can be formulated that inspect only the local neighborhood of the point. Since they worked with cubical voxels this definition uses the 26-neighborhood. We adapt this characterization to the bcc grid. We say a point $p \in O \subset \mathcal{B}$ is simple if

Condition 1: $C(N_{14}(p) \cap O) = 1$ and $C(N_{14}(p) \cap \overline{O}) = 1$.

This condition restricts the choice of removable points to border points. Furthermore it ensures that the deletion of p neither disconnects the local foreground nor the local background. This is proven especially for bcc grids in a companion paper [21].

Since we intend to create 1D curve-like skeletons we further restrict the removal of points to those that satisfy

Condition 2: $|N_{14}(p) \cap O| > 1$.

Obviously this condition preserves end points. Without this condition branches of the skeleton would be removed completely.

In a thinning process the test for simple points is often the most expensive operation. Therefore the parallelization of this step is desirable. However, this parallelization requires a different formulation of the thinning conditions. The example in figure 3(a) shows an object in which each point is considered to be simple.

Tsao and Fu proposed in [28] a method for cubic grids which parallel processes all border voxels which are simple with respect to a prescribed direction. For such a directional thinning approach additional criteria have been formulated that guarantee the preservation of topology. In the following we will introduce the notion of checking rings that adapts the well known concept of checking planes to the bcc grid. Let $O_d \subseteq O$ denote the set of all border points in direction d:

$$O_d = \{p \in O : p - d \in \overline{O}, d \in D_8\}.$$

Further, we consider a special subset of $N_{14}(p)$ that is defined by the intersection of a *grid plane* P with $N_{14}(p)$. To span a grid plane we use a point p and vectors $d_1 \in D_6$ and $d_2 \in D_8$:

$$P(p, d_1, d_2) = \{g \in \mathcal{B} : g = p + \lambda d_1 + \mu d_2; \lambda, \mu \in \mathbb{Z}\}$$

The subsets

$$R(p, d_1, d_2) = P(p, d_1, d_2) \cap N_{14}(p)$$

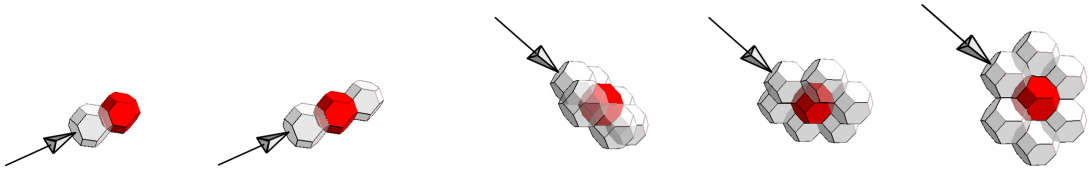
are called *checking rings*.

Note, that for $d_1, d_2 \in D_6$ no connected ring can be derived, but with $d_1, d_2 \in D_8$ provided that $d_1 \neq d_2$ the same rings as specified above can be defined. Remember that $d_1 \in D_6$. For a fixed $d_2 \in D_8$ the vectors d_1 and $-d_1$ define the same checking ring. So for each $d_2 \in D_8$ three different checking rings exist (see Fig. 2 (c) - (e)). In the same way, for a fixed $d_1 \in D_6$ and the vectors d_2 and $-d_2$ the same three checking rings are defined. That means for direction d_2 and $-d_2$ the same checking rings are obtained.

With these definitions the next two conditions can be formulated. A point $p \in O_d$ is parallel removable with respect of a certain direction $d_2 \in D_8$ if in addition to condition 1 and 2 the following conditions are satisfied:

Condition 3: $C(R(p, d1, d2) \cap O) = 1$ for any $d_1 \in D_6$,

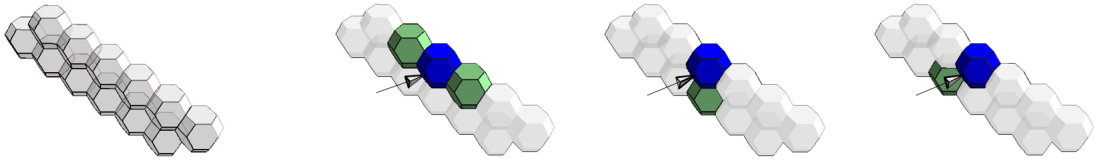
Condition 4: $p + d_2 \in N_{14}(p) \cap O$.



(a) (b) (c) (d) (e)

Figure 2: According to the indicated thinning direction a border point and the determined point (a) and additionally its opposite point (b) are shown. The illustrations (c)-(e) show three checking rings depending on a certain direction.

The third condition is called checking ring condition. It prevents the removal of whole regions due to the parallel processing. For explanation take a look at Fig. 3. For the object (a) each point is simple. The parallel processing of these border points leads to the removal of all points. Let us apply condition 3 for an arbitrary point and a particular direction. The illustrations (b)-(d) show the three checking rings intersected with the object. It is obvious that the checking ring in Fig. 3 (b) is disconnected i.e. the number of foreground components is 2. The checking rings condition prevents the removal.

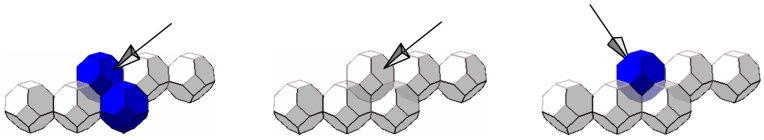


(a) (b) (c) (d)

Figure 3: For the shown object each point is simple and for an arbitrary point (a). In (b)-(d) the foreground points of the three checking rings are emphasized.

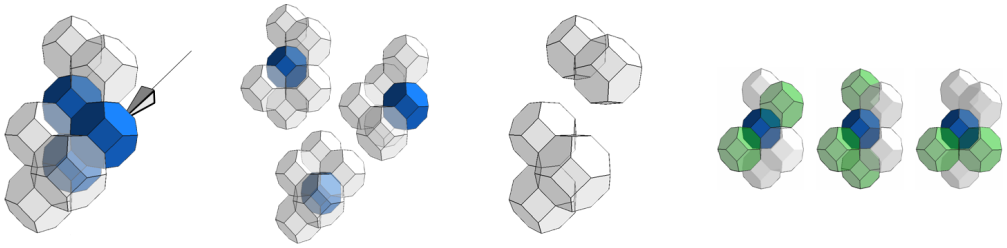
Even with this restriction it is possible that an object may become disconnected. Hence the fourth condition is introduced for special cases like that in Fig. 4. Considering just the first three conditions in case (a), two points would be removed and the topology would change. Now we also consider the condition 4 keeping the thinning direction fixed. None of the points has a sequel point and therefore no point would be removed. Fig. 4 (b) illustrates this situation. In case (c) the thinning direction is different to the cases (a) and (b). Only one point meets all four conditions.

The case that is shown in Fig. 3 is just for a good illustration and cannot occur under consideration of condition 4. Anyway, condition 4 alone is not sufficient to prevent a topological change as you can comprehend in Fig. 5. At least one of the three highlighted border points in Fig. 5 (a) is necessary to connect the both foreground components in Fig. 5 (c). The condition 3 is not fulfilled for example for the topmost border point as Fig. 5 (d) illustrates.



(a) (b) (c)

Figure 4: For case (a) the highlighted points meet conditions 1-3 and are removed. Considering condition 4 no point will removed (b). When thinning direction changes one point fulfil all four conditions (c).



(a) (b) (c) (d)

Figure 5: In (a) the highlighted points meet condition 4 and because they each are simple (b) they are removed in a parallel process (c). Considering condition 3 for the topmost highlighted point, the foreground component of two of the three checking rings is not connected (d).

4 Additional thinning attributes

Since different directions are used the method described above is sensitive to the order in which the different directions are processed and the resulting skeletons may not exactly be centered within the object. Non-directional thinning methods (for instance [12]) do not have this disadvantage. We decided to use additional distance information to achieve this effect. A common way to determine the distance of any object point to the background is a distance transformation. For a fast distance transformation non-Euclidean distances are used as approximation. We apply a distance transformation based on Chamfer distances which is defined for two points p, q by the sum of local distances of the shortest digital path between p and q. For bcc grids local distances denotes the Euclidean distances between 6-neighbored and 8-neighbored points respectively. For voxel sets, thinning algorithms that involve the distance transformation are presented in [11, 18, 23]. Distance transformations on bcc grids are examined in [19, 20].

Since for each foreground point the approximated distance to the background is known, we can define *shells* S_i that contain only points which have a rounded distance i to the background (see fig. 6):

$$S_i = p : dist(p, \overline{O}) = i, p \in O.$$

We are utilizing these sets in the thinning process by iterating through the distances

beginning with the smallest. This is almost equivalent to the idea of thinning out an object layer by layer but is indented in a stricter sense according to different thinning directions. Using the defined shells we achieve that for all eight directions only points with a specific distance to the background will be considered for removal. Let c be a specific thinning distance, p an object point and $dist(p, \overline{O})$ its shortest distance to the background. Than the following is checked:

Condition 5: $p \in S_i : 1 \leq i \leq dist(p, \overline{O})$.

If p is element of a shell S_i, thus the condition holds, the point p is examined in the way described in section 3.

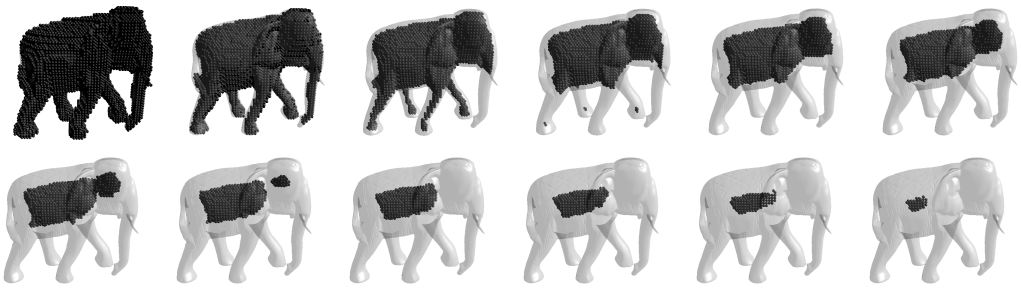


Figure 6: All shells S_1 to S_{12} of an object are shown (resolution: $50 \times 50 \times 25$).

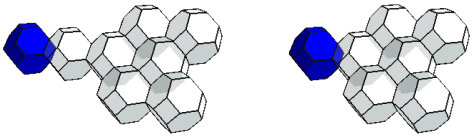
Using all the mentioned conditions curve skeletons with a lot of branches are generated (see Fig. 8). This property may be useful for some applications particularly where reconstruction is required. Though a complete reconstruction of the original object is not possible, it is clear that a widely ramified curve skeleton will generate a better reconstruction [13]. For other applications like virtual navigation or animation using inverse-kinematics only few branches which represent some larger convex regions on the object are desired.

In order to remove smaller branches some pruning algorithms exist and are used as post-process (e.g. [22]). We integrate a pruning mechanism in the thinning process based on an additional condition. Candidates for a pruning are all points classified as end points. However, we apply the pruning only to branches of length 1 to retain the main skeleton curves. Therefore, we require that the single neighbor of an end point must not be a curve point in order to be removable. Or in other words, if a end point is removed its neighbor must not become a end point itself. Let $p \in O$ be a end point and $q \in O$ its unique neighbor. Then p is removable if the following condition (which is independent of the previous ones) holds:

Condition 6: $|N_{14}(q) \cap O_f| > 2$.

We call points that meet this condition *stubble points*. In Fig. 7 an example is shown. Note, that the examined neighborhood is extended over the usual 14-neighborhood, which is a disadvantage for the processing speed.

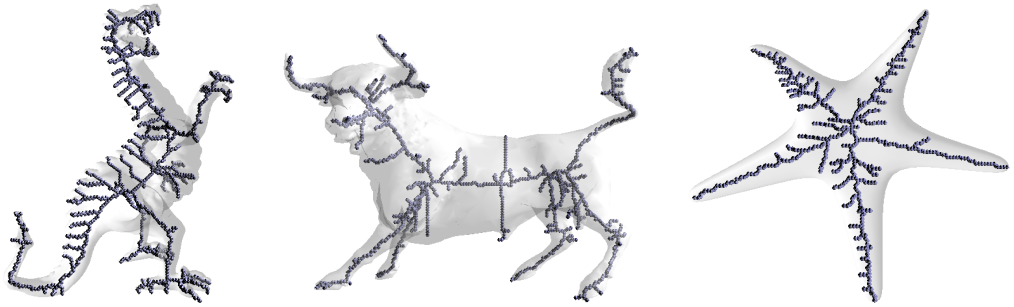
If a point satisfies condition 6 it is removed in any case. The other conditions need not to be checked. Nevertheless, the most important property, the topology retaining,



(a) (b)

Figure 7: A stubble point (a) can be removed without changing the shape of the resulting skeleton to much. The end point in (b) is not classified as stumble point and must not be removed.

is ensured: Stubble points are always simple points, whereas the unique foreground neighbor of a stubble point cannot be simple because there exist always at least two connected foreground components. Comparing the thinning results for some objects with and without involving condition 6 (Fig. 8, Fig. 12), it is obvious that removing stubble points has a great effect on skeleton branching.



(a) (b) (c)

Figure 8: Thinning results without pruning according to condition 6.

5 Thinning algorithm

The ambition for the algorithm design was the high efficiency of the time-consuming parts by means of parallelization. Thus, we outsource the most time-consuming parts into a subfunction which may processed in parallel (Fig. 9).

Before starting the thinning process a distance transform is necessary. The intended purpose of the distance transform is to build sets of points with equal distance to the object border. In practice an array of lists is used. The index to access the array refers to the distance. Each list contains references to points of a certain distance. Fig. 10 illustrates the data structure.

Remember, that we want to remove points layer-wise starting at the outside moving inwards. We achieve this using the data structure described above. Hence, for each distance d (from 1 to m) the point analysis is limited to the points which have a distance to the background less or equal to the given distance d. To coordinate the parallel processing of these points the point set is subdivided into disjoint subsets. The unprocessed subsets

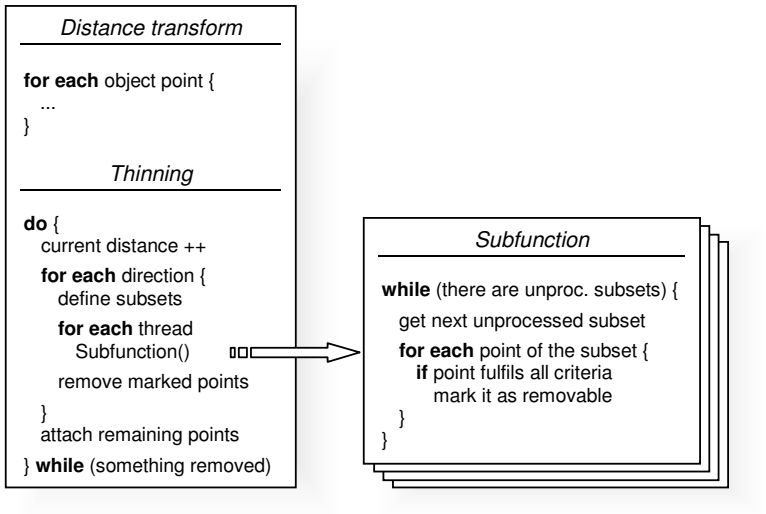


Figure 9: Block diagram of the thinning process including the preprocessing distance transform.

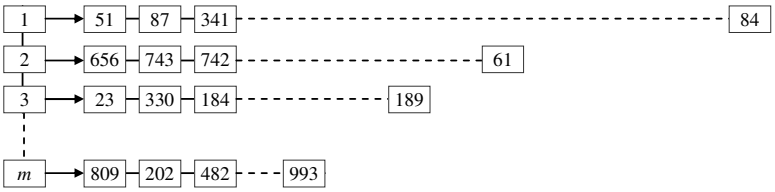


Figure 10: The data structure that organizes the object points into equal-distance-sets.

are evenly distributed to the parallel working subfunctions.

The subfunction itself analyzes each point of a subset for fulfilling all necessary and optional criteria respective. Such points are marked for removal. The removal of all marked points may be performed parallel similar to the previous process. Since this pass needs approximately 1% of the whole computational time, parallelisation is irrelevant. After that step the list l_d (related to the distance d) contains only references to points that still belong to the foreground. Having iterated over all directions the list l_d of remaining foreground points is attached to the next list within the array, $l_d + 1$. This proceeding results from condition 5. To give an example, imagine the first iteration. All points of distance 1 to the background are analysed over all eight directions. After that, some points still remain since they met not all removal conditions. That is not to say they are not removable in a later step. On that account these remaining points are attached to the list containing points with distance 2. Then the current distance is incremented and all points of distance 2 or 1 are analysed.

The thinning process terminates if no more points are removable.

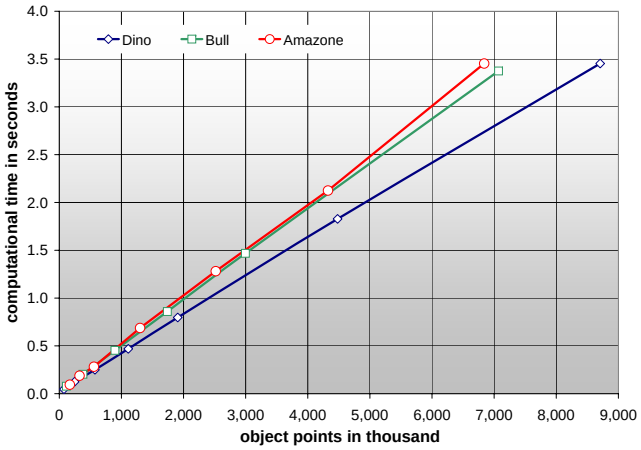


Figure 11: Computational time for various objects in different resolutions.

6 Conclusion and Results

In this article, a parallel working thinning method that is adapted to bcc grids is introduced. We also invent new criteria to hold topological requirements and to influence the shape of the resulting skeleton respectively. Fig. 12 indicates typical results that can be achieved with the presented method including the number of object points and the computational time for thinning (P4, 3 GHz, 2 GB RAM).

We implement the algorithm introduced in section 5 on an Intel dual-core processor. Compared to recent results we achieve skeletons much faster: In [10] a thinning algorithm is implemented that needs more than 100 seconds to extract the skeleton of around 850,000 object points. For that task we need less than one second. Our parallel implementation of the thinning process is up to 90% faster on a dual-core processor which reduces the calculation time to approximately a half second for such objects. On the average we obtain a speedup of factor 1.86 for the multi-threaded version compared to the single-threaded on the dual-core processor. Thus, the suggested approach is optimally suitable for high resolution objects. To get an impression we show in Fig. 11 the computational time for the first three objects in Fig. 12 for different resolutions.

Future work deals with the implementation of an even faster thinning using the GPU on "state of the art" graphics hardware. Furthermore we study conditions for non-directional thinning.

References

[1] Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust. In *SMA '01: Proceedings of the sixth ACM symposium on Solid modeling and applications*, pages 249–266, New York, NY, USA, 2001. ACM Press.

[2] Gilles Bertrand and Zouina Aktouf. A three-dimensional thinning algorithm using subfields. In *Proc. Vision Geometry III*, volume 2356, pages 113–124. SPIE, 1994.

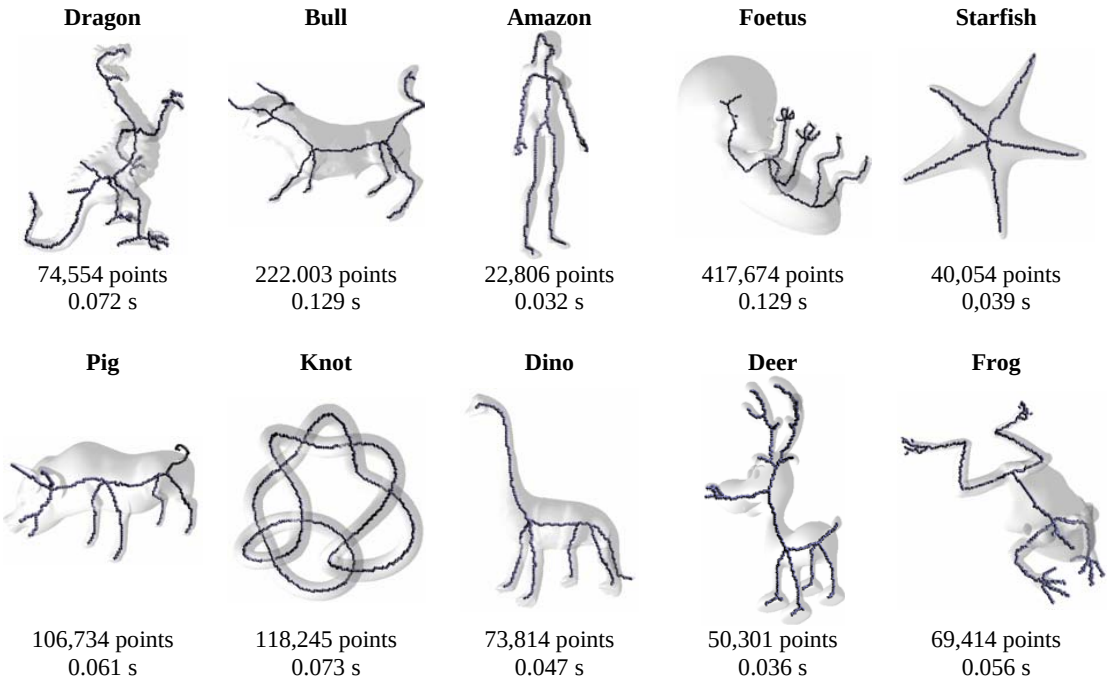


Figure 12: Thinning results for different models.

[3] Gilles Bertrand and Grégoire Malandain. A new topological classification of points in 3d images. In *ECCV '92: Proceedings of the Second European Conference on Computer Vision*, pages 710–714, London, UK, 1992. Springer-Verlag.

[4] Francisco Nivando Bezerra and Neucimar Jerônimo Leite. Some comments on thinning algorithms for 3-d images. Technical Report IC-98-22, Instituto de Computação, Universidade Estadual de Campinas, May 1998.

[5] Jules Bloomenthal. Medial-based vertex deformation. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 147–151, New York, NY, USA, 2002. ACM Press.

[6] Jonathan W. Brandt and V. Ralph Algazi. Continuous skeleton computation by voronoi diagram. *CVGIP: Image Underst.*, 55(3):329–338, 1992.

[7] Angela Brennecke and Tobias Isenberg. 3d shape matching using skeleton graphs. In *Proc. Simulation and Visualization*, Magdeburg, Germany, 2004.

[8] David Brunner and Guido Brunnett. Mesh segmentation using the object skeleton graph. In *Proc. of the 7th IASTED International Conference on Computer Graphics and Imaging*, pages 48–55. ACTA Press, 2004.

[9] Jen-Hi Chuang, Chi-Hao Tsai, and Min-Chi Ko. Skeletonization of three-dimensional object using generalized potential field. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1241–1251, 2000.

[10] Nicu D. Cornea, Deborah Silver, and Patrick Min. Curve-skeleton applications. In *IEEE Visualization*, page 13, 2005.

[11] G. Sanniti di Baja G. Borgefors, I. Nyström and S. Svensson. Simplification of 3d skeletons using distance information. In *Vision Geometry IX, volume 4117 of Proceedings of SPIE*, pages 300–309, San Diego, USA, 2000.

[12] I. Nyström G. Borgefors and G. Sanniti di Baja. Computing skeletons in three dimensions. In *Pattern Recognition*, volume 32(7), pages 1225–1236, 1999.

[13] Nikhil Gagvani and Deborah Silver. Parameter-controlled volume thinning. *CVGIP: Graph. Models Image Process.*, 61(3):149–164, 1999.

[14] Chafiaâ; Hamitouche, Luis Ibánez, and Christian Roux. Discrete topology of A_n^* optimal sampling grids. interest in image processing and visualization. *J. Math. Imaging Vis.*, 23(3):401–417, 2005.

[15] T. Yung Kong and A. Rosenfeld. Digital topology: introduction and survey. *Comput. Vision Graph. Image Process.*, 48(3):357–393, 1989.

[16] Frederic Fol Leymarie. *Three-Dimensional Shape Representation via Shock Flows*. PhD thesis, Brown University, Division of Engineering, Providence, RI, USA, May 2003.

[17] Neophytos Neophytou and Klaus Mueller. Space-time points: 4d splatting on efficient grids. In *VVS '02: Proceedings of the 2002 IEEE symposium on Volume visualization and graphics*, pages 97–106, Piscataway, NJ, USA, 2002. IEEE Press.

[18] Robin Strand. Surface skeletons in grids with non-cubic voxels. In *ICPR (1)*, pages 548–551, 2004.

[19] Robin Strand. The Euclidean distance transform applied to the fcc and bcc grids. In Jorge S. Marques, Nicolas Pérez de la Blanca, and Pedro Pina, editors, *Pattern Recognition and Image Analysis, Second Iberian Conference, IbPRIA 2005, Estoril, Portugal, June 7-9, 2005, Proceedings, Part I*, volume 3522 of *Lecture Notes in Computer Science*, pages 243–250. Springer, 2005.

[20] Robin Strand and Gunilla Borgefors. Distance transforms for three-dimensional grids with non-cubic voxels. *Comput. Vis. Image Underst.*, 100(3):294–311, 2005.

[21] Robin Strand and David Brunner. Simple points on the body-centered cubic grid. Technical Report 42, Centre for Image Analysis, Uppsala University, Uppsala, Sweden, 2006.

[22] Stina Svensson and Gabriella Sanniti di Baja. Simplifying curve skeletons in volume images. *Comput. Vis. Image Underst.*, 90(3):242–257, 2003.

[23] Stina Svensson, Ingela Nyström, and Gunilla Borgefors. Fully reversible skeletonization for volume images based on anchor-points from the d^{26} distance transform. In *Proc. 11th SCIA*, pages 601–608, Kangerlussuaq, Greenland, June 1999.

[24] Thomas Theussl, Torsten Möller, and Meister Eduard Gröller. Optimal regular volume sampling. In *VIS '01: Proceedings of the conference on Visualization '01*, pages 91–98, Washington, DC, USA, 2001. IEEE Computer Society.

[25] Benjamin Vrolijk, Freek Reinders, and Frits H. Post. Feature tracking with skeleton graphs. In *Data Visualization: The State of the Art*, pages 37–52, 2003.

[26] Ming Wan, Frank Dachille, and Arie Kaufman. Distance-field based skeletons for virtual navigation. In *VIS '01: Proceedings of the conference on Visualization '01*, pages 239–246, Washington, DC, USA, 2001. IEEE Computer Society.

[27] Haris Widjaya, Torsten Möller, and Alireza Entezari. Voxelization in common sampling lattices. In *PG '03: Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, page 497, Washington, DC, USA, 2003. IEEE Computer Society.

[28] K.S. Fu Y.F. Tsao. A parallel thinning algorithm for 3d pictures. *Computer Vision, Graphics and Image Processing*, pages 315–331, 1981.