Eurographics Conference on Visualization (EuroVis) 2016
E. Bertini, N. Elmqvist, and T. Wischgoll
(Guest Editors)

Short Paper

Fast 3D Thinning of Medical Image Data based on Local Neighborhood Lookups

Tobias Post[1], Christina Gillmann[1], Thomas Wischgoll[2] and Hans Hagen[1]

[1] Computer Graphics and HCI Group, University of Kaiserslautern, Germany
[2] Advanced Visual Data Analysis Group, Wright State University, U.S.A.

Abstract

Three-dimensional thinning is an important task in medical image processing when performing quantitative analysis on structures, such as bones and vessels. For researchers of this domain a fast, robust and easy to access implementation is required. The Insight Segmentation and Registration Toolkit (ITK) is often used in medical image processing and visualization as it offers a wide range of ready to use algorithms. Unfortunately, its thinning implementation is computationally expensive and can introduce errors in the thinning process. This paper presents an implementation that is ready to use for thinning of medical image data. The implemented algorithm evaluates a moving local neighborhood window to find deletable voxels in the medical image. To reduce the computational effort, all possible combinations of a local neighborhood are stored in a precomputed lookup table. To show the effectiveness of this approach, the presented implementation is compared to the performance of the ITK library.

Keywords: Medical Image Processing, 3D Thinning, Lookup Table

1. Introduction

An important task in medical image processing is the analysis of bones and vessels. To get insight into their geometric and topological properties, centerlines are used [KQ03]. The computation of these centerlines is often based on the thinning of the original structures where voxels are deleted successively until a skeleton remains [LLS92]. Unfortunately, the properties of skeletons are not clearly defined as they depend on the purpose of a skeleton [CSM07].

Contrary to other fields, the medical area claims restrictive requirements for a skeleton [PB13]. First, skeletons of medical structures are required to have a one pixel thickness to be analyzed. Additionally, medical datasets are discrete 3D images that are segmented, thereby forming a binary mask. Furthermore, skeletons of medical structures need to capture the geometry of the examined object as well as possible, meaning, that the thinning should be performed equally in all directions and the length and shape of objects need to be preserved. Additionally, the topological properties of objects have to be retained. This means, that continuous objects need to be presented as continuous skeletons and branching or disconnected objects also require an appropriate skeleton representation.

In medical image processing, various algorithms are available that perform thinning with more or less concern to the mentioned requirements (Section 2). As thinning does not demand a user input, an easy to access implementation is desirable. Libraries such as ITK [JMIC13] are often used in the medical field as they offer various image processing methods including thinning. Unfortunately, this implementations is slow and can delete voxels that belong to the skeleton as they are not suitable for 3D datasets.

To solve this problem, this paper presents a fast and robust implementation for a 3D thinning in Section 3. Therefore, the algorithm presented by Lee et al. [LKC94] is refined as it results in the correctly thinned objects preserving the required properties in medicine. Five criteria are used to examine a moving local neighborhood and determine whether a voxel in the object can be deleted or not. This approach can be slow if all properties need to be checked. In contrast to that, the presented approach evaluates all possible local neighborhood configurations with the given criteria and stores them in a lookup table. During the runtime of the algorithm it is not required to recheck every voxel's surrounding which results in faster thinning computation as comparisons show. The used lookup table is open access available to be included in an arbitrary framework.

Therefore this paper contributes:

- a fast and robust implementation of 3D thinning using lookup tables
- open access to the local neighborhood lookup table

Section 4 will compare the presented approach with the ITK implementation. At last, Section 5 will conclude this paper and give future directions.

DOI: 10.2312/eurovisshort.20161159

2. Related Work

The following section will give an overview on thinning algorithm classes and presents relevant examples and their properties.

Reviews of thinning approaches are provided by Lam et. al [LLS95] (2D) and Saed et. al [STRA10] (3D). They stated, that thinning methods can be divided into iterative and non-iterative approaches. Iterative approaches start with an object that is thinned by iteratively deleting border voxels using a specific scheme until the skeleton remains. This category itself can be divided into parallel and sequential approaches where sequential approaches check for each voxel separately whether it can be deleted or not. Parallel thinning algorithms decide in each iteration for which can be deleted. These algorithms often differentiate between the direction from where voxels are deleted and alter their iterations based on them. Depending on the amount of different iterations an algorithm performs the number of cycles for a parallel thinning algorithm can be determined.

Thinning can be performed for grey scale images as shown in [MD99, CBB13]. Although these algorithms can handle arbitrary image data, their computational effort increases. Additionally, in medical image processing thinning is applied to a preprocessed image where the object to be thinned is already determined via a segmentation step. This results in a binary image that does not benefit from the flexibility of a grey-scale image thinning. Therefore, this paper addresses binary images as the medical domain does not require a grey scale thinning approach.

Thinning methods for binary images are available as sequential [Hil69] or parallel approaches [AdB89, AW02]. Their common drawback is the thinning result computed without concerning the local neighborhood of a voxel. They result in skeletons that do not have a thickness of one voxel. As mentioned before, this is an important requirement in medical image processing. To solve this problem, this paper presents a thinning algorithm based on local neighborhood evaluations, that outputs a skeleton with a thickness of one voxel.

Various methods [BNB99, EM93, MS96, MBPL99] use a moving local neighborhood window to evaluate if a voxel can be deleted or not. Unfortunately, they do not preserve the geometric and topological properties of the examined medical object. Therefore, this paper presents a thinning method based on local neighborhood evaluation that results in a skeleton suitable for medical image processing.

Lee et. al [LKC94] presented a parallel, topology and geometry preserving algorithm that results in a skeleton with a thickness of one voxel. Therefore, this algorithm is able to fulfill the mentioned requirements for skeletonization of medical image data. The approach is implemented in different libraries [Rel, Hom07]. The underlying algorithm determines deletable voxels by checking different properties. Although the approach is based on a lazy evaluation (if one property does not hold, the remaining do not need to be checked) this approach can be computationally expensive. Therefore, this paper presents a fast implementation that is able to identify deletable voxels correctly, fast and with an equal time consumption in all possible cases by using a lookup table.

3. Methods

According to the requirements in medical image processing, the following section describes a fast and robust implementation to achieve a 3D thinning that is suitable for medical image data. The approach performs a parallel thinning based on a moving local neighborhood determining deletable voxels in six subcycles. As an input the algorithm requires a discrete, two or three-dimensional binary image.

3.1. Local Neighborhood Evaluation

To determine removable voxels of an image object each voxels local neighborhood is evaluated. This evaluation has to ensure, that none of the formulated requirements for medical image processing is violated. To achieve this, Lee et al. [LKC94] proved that the following five properties are sufficient to identify deletable voxels:

1. The current voxel has to be set
2. Concerning the current direction of thinning: the voxel in front of the current voxel is not allowed to be set
3. The total number of neighbors must be higher than 1
4. The euler characteristic needs to be 0 [GW06]
5. The considered voxel has to be a simple point [GB90]

The properties are computed in the listed order until a criteria is not fulfilled. This can result in different runtimes depending on the criteria that causes the computation to stop. In the worst case all criteria need to be checked, what slows the computation. As a consequence, the runtime of the thinning procedure does not depend on the size of the input image. Instead it depends on the presence of slow cases. Therefore, the aim is to achieve a constant runtime for all possible local neighborhood settings.

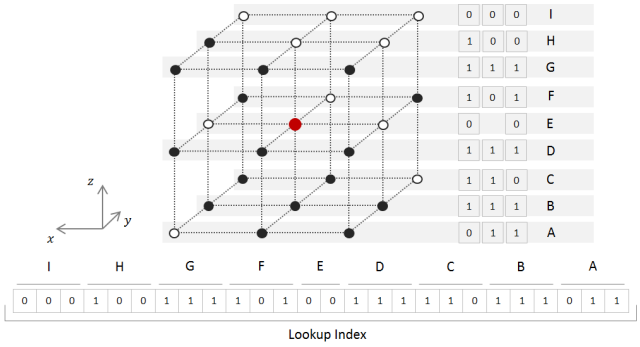

Figure 1: *Example local neighborhood for a voxel and the resulting lookup index used to determine whether this voxel is deletable.*

As the output of the local neighborhood evaluation remains the same in any state of the thinning algorithm the use of a lookup table is suitable. With this approach the evaluation time of a voxel can be equalized for all cases to a constant amount. A lookup table that solves this problem can be realized in different manners: First, it is possible to use a separate lookup table for each direction, what would result in 6 tables each containing 2^{26} entries. As the direction can be encoded in the lookup table their number can be reduced accordingly. Another option would be the identification of symmetric

and rotated cases of the local neighborhood. The resulting lookup table would consist of a minimal set of possible local neighborhoods and the algorithm would need to identify the matching case in the lookup table. This would not lead to an acceleration of the algorithm run as the algorithm would need more computational steps to determine the correct local neighborhood setting.

Based on these simplifications a lookup table that contains 2^{26} entries, storing all possible settings of the local neighborhood for a voxel is sufficient. This results in a 8MiB file, that can be generated in 11 seconds or loaded in less then one second.

In a precomputational step, all settings are evaluated based on the mentioned criteria and stored in the corresponding entry of the lookup table. To access an entry of the lookup table, the algorithm reads all values in the local neighborhood and strings them together to a lookup index according to a fixed scheme as shown in Figure 1. During the runtime of the algorithm there is no need to recheck any of the defined criteria.

The presented lookup table can be used as a basis for a thinning algorithm that requires a constant, low evaluation time for each local neighborhood evaluation.

3.2. Thinning Algorithm

The following pseudocode sketches the optimized thinning procedure and shows how the lookup table is used to successively remove voxels until the skeleton remains:

function THINNING(*Image*)
  repeat
    $modified \leftarrow false$
    for $d \in \{up, down, right, left, forward, backward\}$ **do**
      $Candidates \leftarrow \emptyset$
      for $v \in Voxels$ **do**
        if $\begin{pmatrix} Image(v) = 1 \wedge \\ Image(v-d) = 0 \wedge \\ LookupTable(v) = 1 \end{pmatrix}$ **then**
          $Candidates \leftarrow Candidates \cup \{v\}$
      for $c \in Candidates$ **do**
        if $LookupTable(c) = 1$ **then**
          $Image(c) \leftarrow 0$
          $modified \leftarrow true$
  until $\neg modified$

As an initial step, the input image is enlarged in each direction by adding a voxel with the values of 0. Still, the thinning algorithm works on the original image voxels. The benefit from the duplicated border lies in the avoidance of special border cases. In the original image, the algorithm needs to check each voxel if it belongs to the border what can be computational expensive. This problem can be avoid by the enlarged border.

After that, the thinning can be started. Each iteration consists of 6 cycles that alter the direction d of the thinning (up, down, right, left, forward, backward). This ensures, that the remaining skeleton lies as close to the center of the thinned object as possible.

In each iteration all voxels of the input image are evaluated by the moving local neighborhood. If their corresponding entry in the lookup table ($LookupTable(v)$) identifies the current voxel as deletable it is stored in a list of *Candidates*. This list holds all voxels that fulfill the five deletion criteria and thus they are candidates to be removed.

After all voxels are evaluated and the list of *Candidates* is filled, the algorithm rechecks the lookup table for all *Candidates* sequentially. If the checked voxel is still deletable it is finally deleted in the image. This can change the local neighborhood of voxels in the list of *Candidates* what makes the recheck necessary. The recheck of a voxel needs to be followed directly by its deletion, as simultaneous deletion could lead to topological or geometrical changes of the skeleton.

After the recheck and final deletions, the next cycle starts from a different direction, working on the manipulated image. This is repeated until the image cannot be modified any longer from any of the thinning directions.

As Lee et al. showed, this results is a skeleton that has a thickness of one, preserves the geometric as well as the topological properties of the thinned object.

4. Results and Discussion

The standard implementation given by the ITK 4.9 version [Hom07] can be applied to a 3D dataset, but outputs incorrect results as it performs an independent 2D thinning for each layer. Although each of these layers is thinned correctly, the connection of the thinned object through the layers is not checked what can result in wrong thinning results. To demonstrate the improvements of the presented implementation, the approach is compared to the ITK implementation of Lee et al.'s approach [Hom07] that can be found in the itk journal. To obtain comparable results, the thinning procedure was uncoupled from the ITK workaround, that dictates the used datastructures. This ensures both implementations to work with the same underlying datastructures. Experiments showed, that the average evaluation of local neighborhood settings is five times faster with the presented approach in comparison to the ITK journal implementation. As the frequency of different local neighborhood settings, the number of iterations and the rechecking phase influence the computation time for different datasets.

Table 4 compares the computation time of different datasets between the ITK Journal implementation and the presented approach. The size of the tested datsets range from 512x512x199 to 513x513x513. The used datasets have a voxel ratio from 0.2118 to 15.15 what describes the percentage of the image voxels that are set. Both implementations require the same amount of iterations ranging from 7 to 38 in the presented cases. The time savings from our approach range from 61 % to 69 % which is about one third in each case. The presented approach is up to double as fast as the ITK implementation performed on a single core computer with 2.3GHz as further experiments showed.

The tested datasets and their thinning results are shown in Figure 2. The resulting thinned object is always equal for both implementations. The examples show, that the presented approach results in a one pixel wide skeleton, that captures the geometric and topological properties of the original objects. Additionally, it can be observed

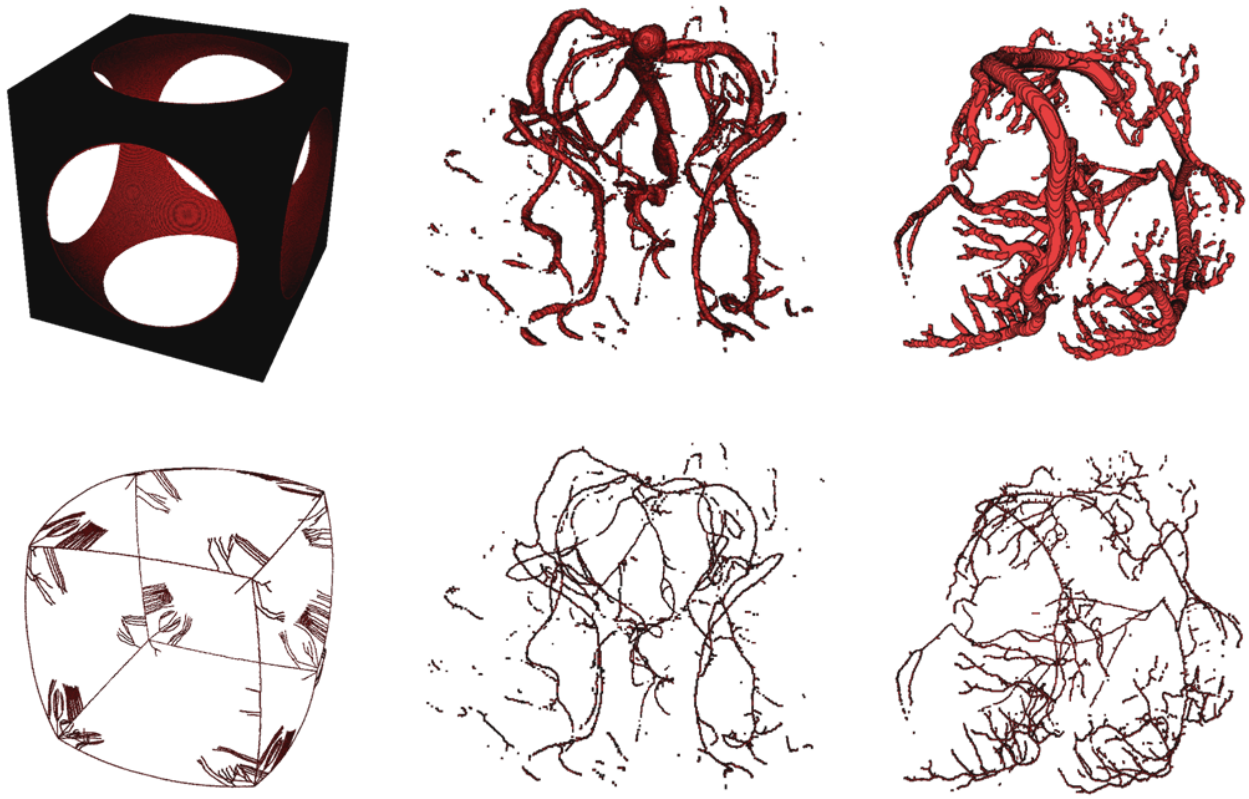

Figure 2: *Example datasets (top) and their thinning result (bottom). Left: Synthetic Cube Dataset with holes. Middle: Aneurysm [Dat]. Right: Vascular system of a pig dataset.*

Dataset	Cube	Aneurysm	Vessels
Size	513x513x513	512x512x512	512x512x199
Voxels Original	15.15 %	0.2118%	0.4997%
Voxels Thinned	0.0168 %	0.0049%	0.0164%
Iteration	38	10	7
Time ITK	85.1	18.8	4.982
Time LUT	52.2	11.6	3.289

Table 1: *Example datasets, their size, number of thinning iterations and compared runtimes of the presented approach with the ITK implementation.*

that the skeleton remains in the center of the objects. Like in the original ITK Journal implementation for 3D thinning, the runtime is linearly dependent on the size when scaling an input image.

With the presented approach all possible settings of the local neighborhood can be evaluated by an equally optimal time consumption. The offered lookup table can be found in the supplementary material to be reused for any dataset that holds binary values. Since, this approach is based on the work of Lee et al. who proved that the number of iterations is minimal, our approach output optimal thinning results.

5. Conclusion

This paper presents a local neighborhood-based thinning implementation utilizing a lookup table. Therefore, the algorithm is optimized to address the special requirements in medical image processing. The obtained implementation is able to generate a one pixel wide skeleton that preserves the geometry and topology of the examined objects. As the lookup table approach of the moving local neighborhood reduces the computational effort of finding deletable voxels to a constant cost, the underlying algorithm is accelerated. The lookup table is provided to be reused in an arbitrary framework (see Supplementary material).

As future work, a framework for arbitrary lookup tables is planned that allows programmers to realize thinning algorithms suitable in their given use case. Furthermore, the extension of the lookup table based approach to grey scale images is considered.

Acknowledgements

This research was funded by the German Research Foundation (DFG) within the IRTG 2057 “Physical Modeling for Virtual Manufacturing Systems and Processes”. We would like to thank Ghassan Kassab’s research team for providing their datasets used in this study.

References

[AdB89] ARCELLI C., DI BAJA G. S.: A one-pass two-operation process to detect the skeletal pixels on the 4-distance transform. *IEEE Trans. Pattern Anal. Mach. Intell. 11*, 4 (1989), 411–414. 2

[AW02] AHMED M., WARD R. K.: A rotation invariant rule-based thinning algorithm for character recognition. 1672–1678. 2

[BNB99] BORGEFORS G., NYSTRÃŰM I., BAJA G. S. D.: Computing skeletons in three dimensions. *Pattern Recognition 32*, 7 (1999), 1225 – 1236. 2

[CBB13] COUPRIE M., BEZERRA N., BERTRAND G.: A parallel thinning algorithm for grayscale images. In *Discrete Geometry for Computer Imagery*, Gonzalez-Diaz R., Jimenez M.-J., Medrano B., (Eds.), vol. 7749. Springer Berlin Heidelberg, 2013, pp. 71–82. 2

[CSM07] CORNEA N. D., SILVER D., MIN P.: Curve-skeleton properties, applications, and algorithms. *IEEE Transactions on Visualization and Computer Graphics 13*, 3 (May 2007), 530–548. 1

[Dat] The volume vis database. http://www.volvis.org/. Accessed: 2016-02-19. 4

[EM93] ECKHARDT U., MADERLECHNER G.: Invariant thinning. *IJPRAI 7*, 5 (1993), 1115–1144. 2

[GB90] GONG W., BERTRAND G.: A simple parallel 3d thinning algorithm. In *Pattern Recognition, 1990. Proceedings., 10th International Conference on* (Jun 1990), vol. i, pp. 188–190 vol.1. 2

[GW06] GONZALEZ R. C., WOODS R. E.: *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006. 2

[Hil69] HILITCH C. J.: Linear skeletons from square cupboards. In *Machine Intelligence 4*, Meltzer B., Michie D., (Eds.). Edinburgh University Press, 1969, p. 403. 2

[Hom07] HOMANN H.: Implementation of a 3d thinning algorithm. http://hdl.handle.net/1926/1292, 10 2007. 2, 3

[JMIC13] JOHNSON H. J., MCCORMICK M., IBÁÑEZ L., CONSORTIUM T. I. S.: *The ITK Software Guide*, third ed. Kitware, Inc., 2013. *In press*. URL: http://www.itk.org/ItkSoftwareGuide.pdf. 1

[KQ03] KIRBAS C., QUEK F.: Vessel extraction techniques and algorithms: a survey. In *Bioinformatics and Bioengineering, 2003. Proceedings. Third IEEE Symposium on* (2003), pp. 238–245. 1

[LKC94] LEE T.-C., KASHYAP R. L., CHU C.-N.: Building skeleton models via 3-d medial surface/axis thinning algorithms. *CVGIP: Graph. Models Image Process. 56*, 6 (1994), 462–478. 1, 2

[LLS92] LAM L., LEE S.-W., SUEN C. Y.: Thinning methodologies-a comprehensive survey. *IEEE Trans. Pattern Anal. Mach. Intell. 14*, 9 (1992), 869–885. 1

[LLS95] LAM L., LEE S.-W., SUEN C. Y.: Document image analysis. IEEE Computer Society Press, 1995, ch. Thinning Methodologies&Mdash;a Comprehensive Survey, pp. 61–77. 2

[MBPL99] MANZANERA A., BERNARD T. M., PRÊTEUX F., LONGUET B.: A unified mathematical framework for a compact and fully parallel n-d skeletonization procedure. In *Proc. SPIE, Vol. 3811, Vision Geometry VIII* (July 1999), pp. 57–68. 2

[MD99] MERSA S. S., DARWISH A. M.: A new parallel thinning algorithm for gray scale images. In *IEEE Nonlinear Signal and Image Proc. Conf* (1999), pp. 409–413. 2

[MS96] MA C., SONKA M.: A fully parallel 3d thinning algorithm and its applications. *Computer Vision and Image Understanding 64*, 3 (1996), 420 – 433. 2

[PB13] PREIM B., BOTHA C. P.: *Visual Computing for Medicine: Theory, Algorithms, and Applications*, 2 ed. Morgan Kaufmann Publishers Inc., 2013. 1

[Rel] Matlab central - file exchange. http://www.mathworks.com/matlabcentral/fileexchange/43400-skeleton3d. Accessed: 2016-02-12. 2

[STRA10] SAEED K., TABEDZKI M., RYBNIK M., ADAMSKI M.: K3m: A universal algorithm for image skeletonization and a review of thinning techniques. *Applied Mathematics and Computer Science 20*, 2 (2010), 317–335. 2