On Skeletonization in 4D Images

Pieter P. Jonker and Obbe Vermeij

Pattern Recognition Section, Faculty of Applied Physics,
Delft University of Technology, P.O. Box 5046,
2600 GA Delft, The Netherlands
e-mail: pieter@ph.tn.tudelft.nl

Abstract. In this paper we extended the 3D skeletonization method based on Euler number count & cluster count to 4D. As the Euler sum in 4D is always 0, we solved this by embedding 4D objects in a 5D space to calculate the 5D Euler number. The 5D Euler number changes due to 3 events: 1D tunnels, 2D tunnels and 3D tunnels. As the latter does not occur in 4D objects, together with a foreground and a background cluster count, the number of equations is equal to the number of unknown variables, and a breakpoint change can be detected. However, this also indicates that making a skeleton in this way is limited to the 4^{th} dimension.

1 Introduction

Our aim is to provide insight in topology preserving thinning or skeletonization in 4 dimensional binary images. The motivation for this research was found first of all in the use of skeletons from 3D images from confocal microscopes, CT, NMR or ultrasound sensor systems or robot range sensors. The motivation for thinning in images with a dimension of 4 can be found in problems like finding the trace of a 3D object moving in time. Generally, thinning "images" with dimensions higher than 3 can be used for finding the shortest or safest non colliding path of an object in an N dimensional space. These problems are frequently encountered in Printed Circuit Board and VLSI mask routing problems (a 3D problem: x, y, z), planning of mutually collision free routes for multiple autonomous vehicles (a 4-D problem: x, y, ø, t), or the planning of a simultaneously collision free path for a multi robot system (e.g a 6 axis robot combined with a 4 axis yields a 10-D search image).

Robot path finding itself, is a consequence of the robot vision problem: If a robot's mission to grasp an object is guided by the objects in its field of view, it should avoid to collide with the objects it does not need. Usually, path planning solutions are based on a (heuristic) graph search procedure (such as A*, or uniform cost) applied on an equidistantly sampled grid; i.e. an image. Uniform cost search in an equidistantly sampled space is then completely comparable to a constrained distance transform, followed by a steepest descent from a start to a goal point. As speed is a main issue in robotics, in the sense that planning a path should take about the same time as the robot move, hardware and fast software

implementations have been investigated and developed by the authors [2, 3], however they are still not fast enough. By way of example, fig. 1 shows how a skeleton without endpixel conditions but anchored to a start and goal point can be used to find the safest shortest path from start to goal. If the letter T is a top-view of a corridor, then the safest path in the sense of maximum distance to the walls is given by the anchor skeleton. See [4, 6] for the structuring elements of the anchor skeleton.

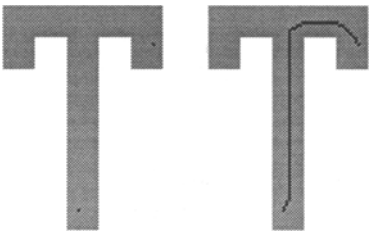


Fig.1. Using the anchor skeleton for path finding

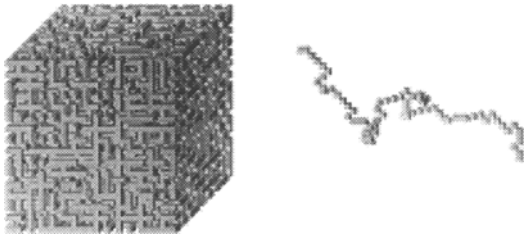


Fig.2. 3D maze solved by an anchor skeleton operation

Fig.3. Skeletonization in 3D applied to path finding

To find a path through a 3D maze the same approach, using a 3D anchor skeleton can be used. See fig. 2. A solution to a 3D planning problem for an Automated Guided Vehicle (AGV) that should be planned in x, y and the vehicle's rotation ø

can be solved by first converting the problem into a binary image of the appropriate dimension (the configuration space) and then taking the anchor skeleton. Fig. 3a shows the configuration space (x, y, ø), fig. 3b shows the 3D skeleton of the background anchored to start and goal point. If more than a single path is found, a constrained distance transform, over the path branches may be used to select the shortest skeleton branch as the shortest safest path. This collision free path is shown in world space (x, y) in fig. 3c. In this solution topology preserving thinning is used as a fast pre-processing phase to reduce the number of voxels.

Fig. 4 shows the result of a 4 dimensional anchor skeleton operation to find a mutually collision free path for two AGVs in a 2D world with a configuration space in 4D (x_a, y_a, x_b, y_b). Figs. 4a through 4c show the two AGVs as a small and a big circles traveling over a number of path points. The big circle's aim is to travel from upper left to lower left. The small circle's aim is to travel from lower right to upper left. Note that both circles avoid eachother, as it is a mutually collision free path.

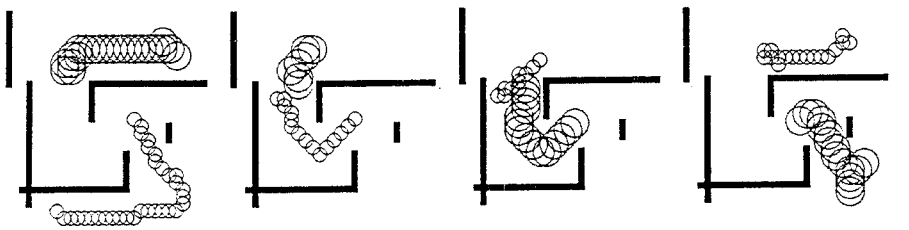


Fig.4. Skeletonization in 4D applied to path finding for 2 AGVs

As thinning is a implemented as a cellular logic operation or Hit-or-Miss aproach in terms of mathematica morphology [4, 5, 6], massively parallel implementation is within reach and hence, possibly, the required speed for real time robot path planning. That is: less than a second. Consequently, our focus is on the search for generic topology preserving conditions for thinning in high dimensional images, in a form that facilitates parallel implementation. Section 2 of this paper is a short introduction in skeletonization. Section 3 holds the functional comparison of three 3D algorithms based on different principles, but with identical result. With this knowledge, in section 4, a 4D algorithm is developed based on change detection using the Euler sum and cluster counting. Conclusions can be found in section 5.

2 Skeletonization in 2D and 3D

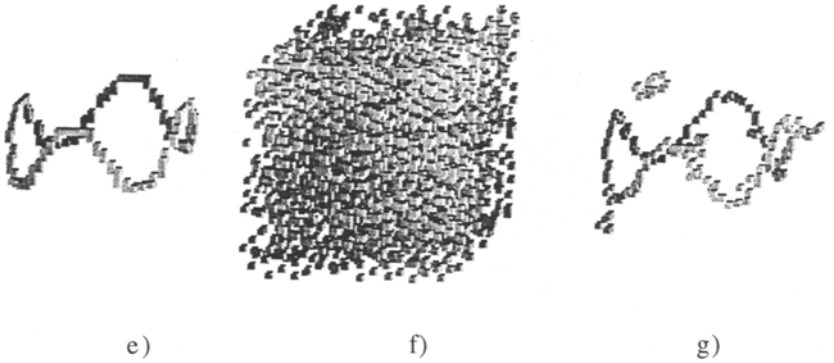
In [4, 5, 6] it was shown that skeletonization can be written as a Hit-or-Miss transform using as a set of masks with size 3^N as structuring element, with N the dimension of the image. Skeletonization can also be seen as a reduction of the intrinsic dimensions of the object in the image. In 1D: Lines are eroded to points,

points are eroded to void. In 2D: Surfaces are eroded to curves, curves to points, points to void. In 3D: Volumes to surfaces, surfaces to curves, curves to points, points to void. And finally for 4D: Hypervolumes to volumes, to surfaces, to curves, to points, to void. At any intrinsic dimension one can stop the skeletonization procedure by using the correct end point criteria within the topology check procedure, e.g. surface-end or curve-end criteria. For path planning one is interested in reduction to a curve. The points on the curve are the path points.

For 2D and 3D, storing the masks in an AND/OR array (PLD) makes hardware possible that allows a neighbourhood check in a single clockcycle. The feasibility of this approach for 4D is uncertain as the number of masks (being around 250 for 3D) is likely to run out of hand in 4D and higher. This biased the search for solutions in 4D and higher towards a more algorithmic approach than the pure Hit-or-Miss approach. Note however, that the mask sets implement an algorithm. The method with which the masks had been generated can also be used as topology check during the image scan, however, at the penalty of a loss of speed. Extending this mask method to 4D appeared to have the problem that, apart from the large numbers of masks it generated, the intersection of foreground and background masks in 4D was not trivial. And hence the need arouse for a second algorithm to verify the mask approach. Reason to focus first on skeletonization in 4D based on the Euler number and on cluster counting. I.e. the methods of Lobregt / Toriwaki [7, 9] and Malandain [8]. The three 3D thinning algorithms had been compared and proven to be identical by exhaustive verification [5].

A way to verify the correct operation of a skeleton is to compare it with a second algorithm. However, this will not yield a "ground truth" as both algorithms can be wrong. A method we developed to verify 4D skeletons is to generate closed structures such as hollow balls and closed space curves, contaminate them with foreground noise, and check whether the skeletonization algorithm is able to find the structure again within the noise, without breaking the topology. See fig. 5. In which: a) e) originals, b) f) noise contaminated, c) g) skeletons found

a) b) c)



e) f) g)

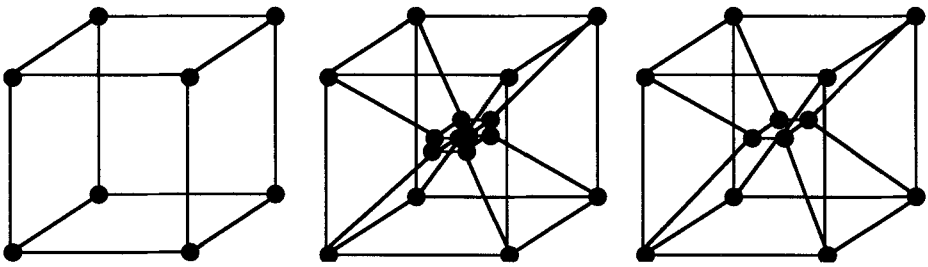
Fig.5. Verifying algorithms by thinning noise contaminated images

3 Methods in 3D

Before we start with Euler counting in 4D we will explain the procedure in 3D. In the application of the Euler sum for topology checks of objects, one uses the fact that the Euler number (N) can be calculated using the nodes (n), edges (e), and faces (f), that constitute a closed net surface around the object With:

$$N = n-e+f \qquad (1)$$

The Euler number of fig. 6a is: N = 8 - 12 + 6 = 2. For fig. 6b, in which top and bottom face are squeezed inward: N = 16 - 28 + 14 = 2. In fig. 6c, when the squeezing resulted in a single face in the middle of the object: N = 12 - 24 + 13 = 1, and when the squeezing resulted in a break-through (a tunnel): N = 12 - 24 + 12 = 0. Fig. 6c shows that combining objects does not change the Euler number, but that each tunnel leaves two faces out. Consequently the Euler number N = -2. For fig. 6d, the Euler numbers are N = 0 for one tunnel and N = - 4 if the left and right faces on the cube are squeezed inwards to force a second tunnel through the centre. This squeezing breaks 4 faces. Objects with intersecting tunnels can be continuously transformed into objects with tunnels that do not intersect. See fig. 7, that makes clear that the second tunnel that intersects the first are actually two extra tunnels from a topological viewpoint.



a) b) c)

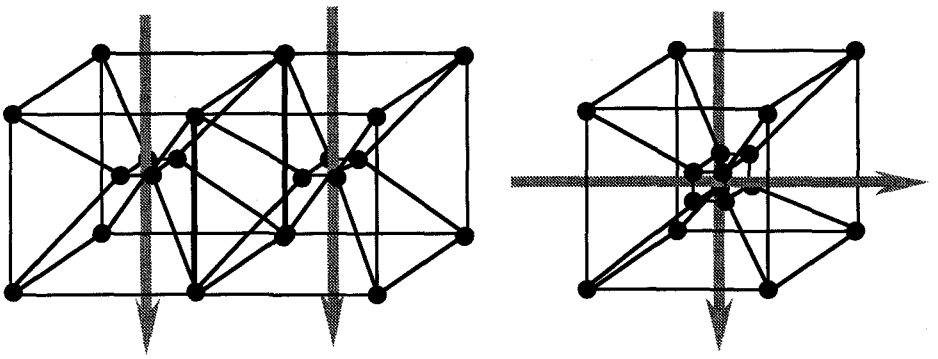


Fig.6. Euler number for various net surfaces

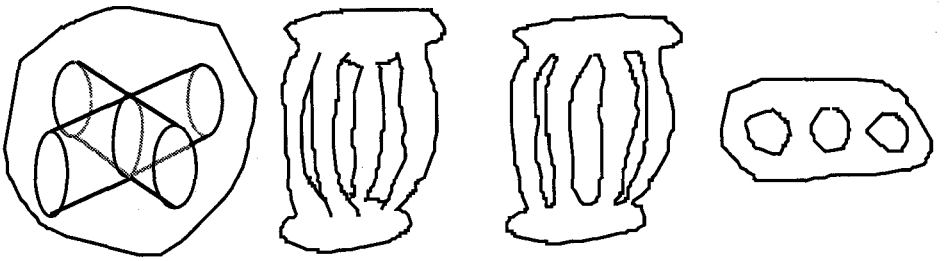


Fig.7. Continuous transformation of objects

The Euler number can be used for a topology preservation check by taking a 3^N neighbourhood, assuming a net surface around the structure found in this neighbourhood and counting the Euler number before and after the removal of the central voxel. If the number changes the central voxel was a 6-connected skeleton point and should remain. Fig. 8a shows an example of a structure in a 3^3 neighbourhood with N = 2. After the removal of the central voxel (fig. 8b), N = 0.

Note that only the edges, nodes and faces directly adjacent to the central voxel need to be inspected for the change detection. This was used by Lobregt [7] in 1981 for his 3D skeleton, where the Euler number was calculated by adding the 8 contributions, looked up in 8 LUTs using the 2 x 2 x 2 neighbourhoods around the central voxel as input.

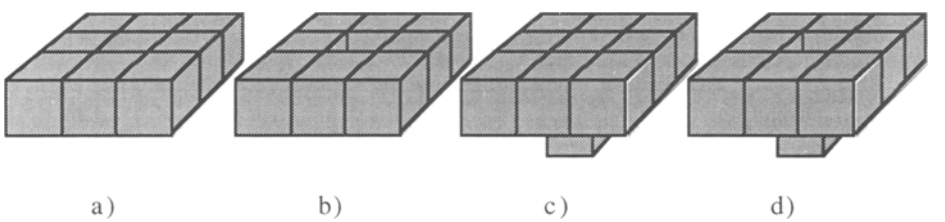


Fig.8. Effect of Euler count in a 3 x 3 x 3 neighbourhood

Figs. 8c and 8d show an object with and without central voxel present for which the Euler count remains N = 2 for both situations. In fact from the viewpoint of the net surface, there are now two touching objects (clusters) instead of one. This was already remarked by Toriwaki in 1982 [9]. He introduced a second rule: Apart from the Euler number, the number of clusters in an 18 connected environment should also not change. Verwer [10] used in 1991 the Lobregt LUTs and a LUT to store all possible clusters in the 18-connected environment for a fast implementation of the 3D skeleton.

Fig. 9 shows the effect of the two rules. In fig. 9a,b a 6 connected foreground surface is broken by a 26 connected background curve when the central voxel changes value. In fig.s 9c,d a 6 connected space curve is split by a 26 connected surface when the central voxel changes value. In fig. 9a and 9b the Euler number does not change, however the number of 6-connected foreground clusters change within the 18-connected environment. Beware that the voxel marked with a * does not belong to the 18-connected environment. Fig. 8c and 8d show that the cluster count rule preserves also 6 connected foreground space curves.

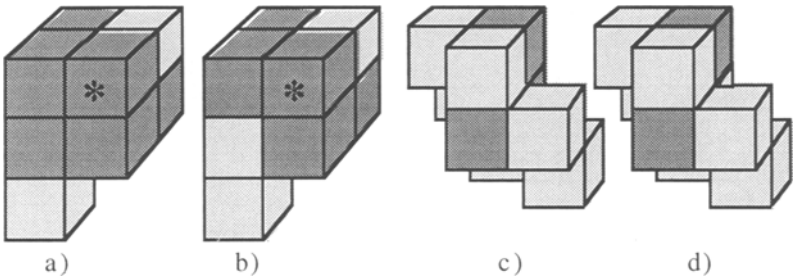


Fig.9. The Lobregt / Toriwaki skeleton breakpoint rules

The algorithm of Malandain [8] published in 1992 for a 26-connected skeleton is entirely based on cluster counting:

A voxel may be removed if:

- The number of 6 connected background clusters connected to the central voxel = 1, before removal of the central voxel.

and:

- The number of 26-connected foreground clusters connected to the central voxel = 1, after the removal of the central voxel, checked in the full 18 connected neighbourhood.

Transforming this rule with logic inversion gives:

A voxel must stay if:

- The number of 6 connected background clusters connected to the central voxel ≠ 1, before removal of the central voxel.

or:

- The number of 26-connected foreground clusters connected to the central voxel $\neq 1$, after the removal of the central voxel, checked in the full 18 connected neighbourhood.

It can be easily seen in fig. 9 that the first rule preserves the 26-connected surfaces and the second rule preserves the 26-connected space curves.

Jonker [4, 5, 6] described thinning as a hit-or-miss operation with a large set of masks. For 3D breakpoint detection this set of mask could be split in a subset that preserves the surfaces and a subset that preserves the space curves. The surface subset being generated by generating and intersecting all possible 26 connected foreground surfaces with all possible 6-connected background space curves. The space curve subset being generated by generating and intersecting all possible 26 connected space curves and intersecting them with all possible 6 connected background surfaces. Fig. 9d can be seen as an example of a surface preserving mask. Fig. 9b can be seen as an example of a space curve preserving mask. By exhaustive comparison it was proven that the three methods described above are equal [5, 6]. Interpreting the rules and figs. above makes clear why this is so.

4 Extending to 4D

The most uncomplicated rules in 3D are the cluster count rules. Are they extendible?

From the 3D procedure we know that skeletonization in 4D is:

Thinning hypervolumes to (curved) volumes, to (curved) surfaces, to space curves, and finally to points. The skeletonization either stops at a closed object structure or due to some endpoint condition.

From the mask sets approach in 3D we learn that we can make 4D sets by:

- Intersecting 32 connected curved volumes with 8 connected space curves.
- Intersecting 64 connected surfaces with 8 connected surfaces.
- Intersecting 80 connected space curves with 8 connected volumes.

It can be envisioned that cluster counting makes sense for the first and second situation. In both cases the central pixel prevents forming the junction of two clusters to one, i.e. perforate the volume with a curve. However, the second situation is not so obvious. Reason to look again at the Euler number for higher dimensions.

The Euler number for simple separating contours (= net surface in 3D), is [1]:

$$N = 1 - (-1)^{dimension} \qquad (2)$$

The Euler number calculation per dimensions is:

1D: N = n(odes)

2D: $N = n - e + f(aces)$

4D: $N = n - e + f - v(olumes)$

5D: $N = n - e + f - v + h(ypervolumes)$ (3)

In the previous section it was shown that tunnels through objects in 3D decrease N by 2. (2 faces less) And each tunnel is assumed not to cross another tunnel. If so, actually two extra tunnels are made instead of one and the contribution of N = -4. In 1D and 2D there are no tunnels, in 3D only one type, in 4D there are two types of tunnels: a tunnel with intrinsic dimension of 1, and a tunnel with an intrinsic dimension of 2. In 5D there are 3 types of tunnels (1D, 2D and 3D). For both type of tunnels, drilling in 4D removes two volumes and two simple net surfaces of the third dimension are merged to one. This makes the contribution N + 2 - 2 = N. So the Euler number of a separating volume in 4D will always be 0. No matter what the object looks like. Drilling a 1D or 3D tunnel in a 5D object, removes 2 hypervolumes and merges two 4D net surfaces, contributing: N - 2 + 0 = N - 2. Drilling a 2D tunnel in a 5D object is done by taking a 5D object with a 1D tunnel (N-2), and stretch it, bend it and then close two of the sides. This destroys the 1D tunnel, contributing to the Euler sum: N - 2 + 4 = N + 2. Concluding, in 5D:

$$N = 2 - 2* t_{1D} + 2 * t_{2D} - 2 * t_{3D} \quad (4)$$

It is possible to convert a 4D object to a 5D space by extending the object over one dimension. E.g. extending from 2D to 3D; a unit square can be extended to a unit cube. Practically this can be done by placing the 3^4 neighbourhood into a 3^5 neighbourhood and filling in the extra fifth dimension with zeros. Just as in the 3D case the Euler number can be calculated by adding the contributions of 16, 2^4 sub neighbourhoods of the 3^4 neighbourhood, by way of a Look-Up-Table (with 65536 entries). The Euler number can change due to three different causes, see (4): 1D tunnels, 2D tunnels or 3D tunnels. However, due to the fact that 4D objects are used in a 5D world 3D tunnels do not exist and (4) degenerates to:

$$N = 2 - 2* t_{1D} + 2 * t_{2D} \quad (5)$$

A 4D skeleton can now be made by obeying the following rules. A point may be deleted if:

- The number of foreground clusters does not change (in the 32 connected environment).

and

- The number of background clusters does not change (in the 80 connected environment).

and

- The Euler number does not change.

The first condition preserves the space curves (intrinsic dimension 1). The second condition preserves the curved volumes (intrinsic dimension of 3). The last

condition preserves the curved surfaces (intrinsic dimension of 2), together with the first two conditions. The proper operation of the 4D skeleton has been made evident by testing it with the 4D equivalent of the 3D noise contaminated test images as introduced in section 2 and shown in fig. 5. The change of the 5D Euler number applied to 4D space is a mixture of three events, 1D tunnels, 2D tunnels or a change of clusters. If the number of background clusters do not change, the number of 2D tunnels does not change. Consequently, if the number of clusters from foreground and background do not change, the Euler number change indicates a change in 1D tunnels within the 4D object. The Euler sum truly applied to 5D objects in a 5D world counts also the possible 3D tunnels. Foreground cluster counting and background cluster counting is not enough now, to detect the cause of change in the Euler sum. Consequently, as foreground cluster counting and background cluster counting is all we can do in any dimension, the Euler/cluster count approach is only applicable in dimension 3 and (with a trick) in dimension 4. The 4^{th} dimension is the limit.

Table 1 gives speed figures of the Euler/cluster count method in 3D and 4D. In C with LUTs on a Sparc 1+ on 32^N noise contaminated testimages for the first iteration only.

Tabel 1. Speed indication in seconds of Euler/cluster count method for 3D and 4D.

	3D	4D
solid sphere	0.128	377.23
hollow sphere	0.148	330.00
space curve	0.020	1.30
path finding	0.137	858.47

5 Conclusions

The aim of this paper was to provide insight in skeletonization of 4 dimensional binary images. We illustrated its need and pointed on the fact that the Hit-or-Miss approach gives a handle to hardware implementation and hence fast skeletons. We treated the approache of three different 3D skeletons that yield identical results, and showed their similarities in the rules for the connectivity check. As proving the correctness of a skeleton is quite cumbersome especially in 4D space, we generated 4D test images based on circular objects of various intrinsic dimension, such as 4D hollow hyper balls, circles, etc. and contaminated them with foreground noise. We extended the 3D Euler count & cluster count method to 4D. As the Euler sum in 4D is always 0, we solved this by embedding 4D objects in a 5D space to calculate the 5D Euler number. The 5D Euler number changes due to 3 events: 1D tunnels, 2D tunnels and 3D tunnels. As the latter does not occur in 4D objects,

together with a foreground and a background cluster count, the number of equations is equal to the number of unknowns, and a breakpoint change can be detected. This indicates, however, that making skeletons in this way is limited to the 4^{th} dimension.

6 References

[1] Hilbert D, Cohn-Vossen S (1932) Anschauliche Geometrie, Springer Verlag, Berlin.

[2] Jonker P.P, Dekker S.T, Verwer B.J.H (1988) A hardware architecture for robot path planning, IAPR Workshop on Computer Vision, Special Hardware and Industrial Applications, Tokyo, Japan, 100-104.

[3] Jonker P.P, Ermes J.P.F.A.M. (1994), Parallel processing in collision free robot path planning, Abstracts 13th Benelux Meeting on Systems and Control, Veldhoven The Netherlands, March 2-4 p. 54.

[4] Jonker P.P., Komen E.R., Kraaijveld M.A. (1995) A scalable, real-time, image processing pipeline, Machine Vision and Applications, vol. 8, no. 2, 1995, 110-121.

[5] Jonker P.P., Vossepoel A.M. (1995) On skeletonization algorithms for 2, 3 .. N dimensional images, in: D. Dori, A. Bruckstein (eds.), Shape, Structure and Pattern Recognition, World Scientific, Singapore, 71-80. ISBN 981-02-2239-4

[6] Jonker P.P. (1995) A Toolbox for Mathematical Morphology in 2D Images..Soft and Hardware Implementation Aspects. Comparing 2D & 3D skeletonization algorithms. Mathematical Morphology for 4D images. Summer School on Morphological Image and Signal Processing. Zakopane, Poland, Sept 27-30. Invited lectures Vol 2. ISBN 83-904743-1-x

[7] Lobregt S., Verbeek P.W, Groen F.C.A, (1980) Three dimensional skeletonization: Principle and algorithm IEEE Trans. Patt. Anal. Machine Intell. vol. 2, pp. 75-77

[8] Malandain C, Bertrand B (1992) Fast Characterixation of 3D Simple Points. Proceedings of the 11th ICPR, Vol III, The Hague

[9] Toriwaki J., Yokoi S. , T. Yonekura T, Fukumura F (1982) Topological properties and topological-preserving transformation of a three dimensional binary picture. in Proc. Int. Conf. Patt. Recogn., Munich 414-419

[10] Verwer B.J.H (1991) Distance Transforms; Metrics, Algorithms and Applications. Ph.D. Thesis Faculty of Applied Physics, TU-Delft, Ch 5, 47-57