Characterization of Curve-Like Structures in 3D Medical Images

Gisela Klette and Mian Pan

CITR, The University of Auckland, Auckland, New Zealand
g.klette@auckland.ac.nz

Abstract

This paper proposes algorithms for the analysis of sets of confocal microscope images of human brain tissue which constitute a 3D volume. The identification of suitable features for curve-like structures in these volumes is required for subsequent classification. The final goal is the distinction between brain tissues of patients with different degrees of neurological deceases. The given volumes show varying distributions, shapes and numbers of astrocytes (i.e., brain cells whose shape resembles that of of a star). The hypothesis is that the "distribution" of astrocytes in brain tissue is related to the number and distribution of branch nodes (clustered into *junctions*) in 3D skeletons of these cells. Segmentation is followed by an application of a modified 3D thinning algorithm. Further analysis is based on our definition of 'junctions' and new methods for locating such junctions. We characterize their distribution based on subdividing the volume into subcubes, also using measures of complexity of junctions and distances between junctions (based on different metrics).

Keywords: skeletons, topological thinning, branch nodes, medical image processing

1 Introduction

Curve-like structures appear in different types of 3D biomedical image analysis (e.g., analysis of blood vessels). Our studies are based on sets of confocal microscope images of human brain tissue which constitute a volume. Figure 1 shows a 3D view of such a volume. This example illustrates the topological complexity of "bright curve-like structures" defined by blood vessels and astrocytes. Blood vessels are basically only apparent because astrocytes are attached to them. The shown example contains a "Y-shaped" blood vessel, reaching from lower left to upper right. Our studies are thus not about 3D structures of blood vessels; however, the given distribution of astrocytes is influenced by them.

The state of neurological deceases can be characterized by medical experts just by looking at these images. These evaluations are based on experience, and they are subjective. Given reasoning points out that the "3D structure" of astrocytes, characterized by "density" (numbers of cells per volume) and "complexity" (numbers of curve-like patterns per volume and their topological features), is essential for this characterization. In our project we identified features which allow objective measurements. In a next step we have to verify these quantitative features with medical experts. Therefor, this report is basically about an image analysis task: given are these volume data. How to analyze the curve-like structures given in these volume data?

The general layout of processing steps is kind of straightforward: (1) segmentation and noise reduction, (2) 3D curve thinning, (3) *branch node*

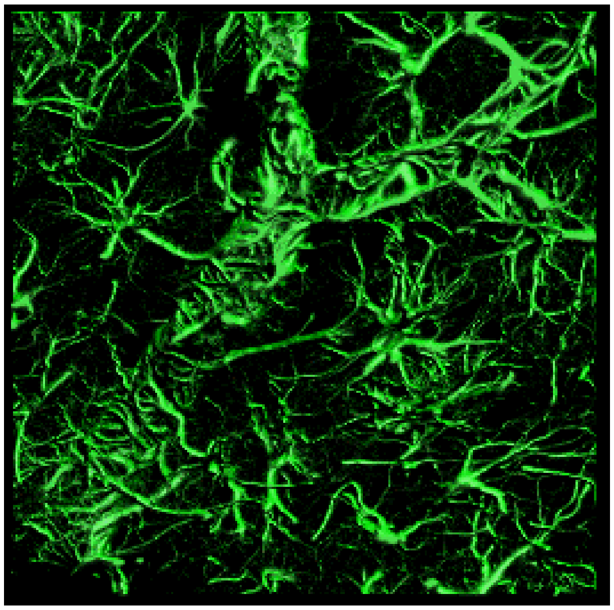

Figure 1: Example of an input data set of 42 slices of 256×256 gray-level images.

identification and arc labeling, and (4) analysis of *junctions*. (The definition of branch nodes follows the classical curve theory of Urysohn and Menger, and junctions will be defined as special clusters of branch nodes.) The paper is structured following these processing steps.

2 Segmentation and Noise Reduction

Global thresholding allowed acceptable results. A threshold was calculated based on assuming bimodal gray level histograms. A result using threshold 55 is shown in Figure 2 (for the data set shown in Figure 1).

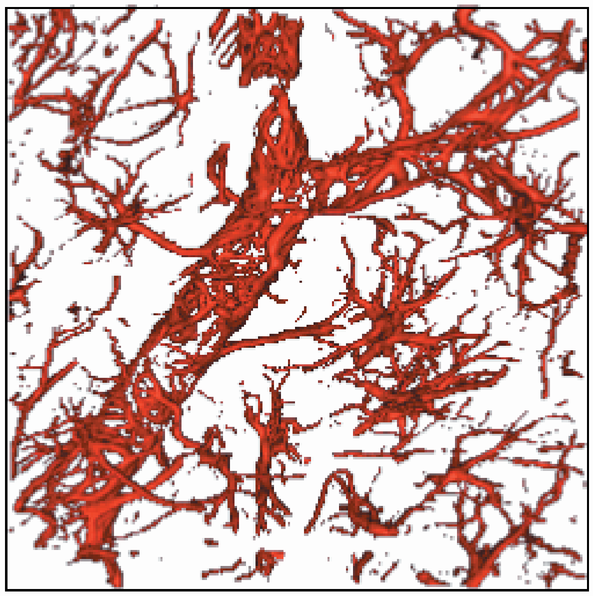

Figure 2: 3D image showing a set of 42 binarized 2D images.

Noise reduction was achieved by applying a sequence (optimized based on experiments) of morphological operations. We decided for 18-closing followed by 6-opening.

3 3D Thinning

3D thinning is a process of repeated removals of simple voxels. In [2, 7] we highlighted the benefit of describing non-simple voxel for achieving a faster version of the thinning algorithm published in [4, 5, 6]. This algorithm applies the concept of deleting simple voxels in 3D sequential 6-subiterations. Original and modified algorithm produce 3D skeletons, where results are basically identical besides length reductions at terminal arcs (having only one endpoint in the 3D skeleton) for the modified algorithm.

In this paper we focus on defining junctions in these skeletons, and we determine the complexity,

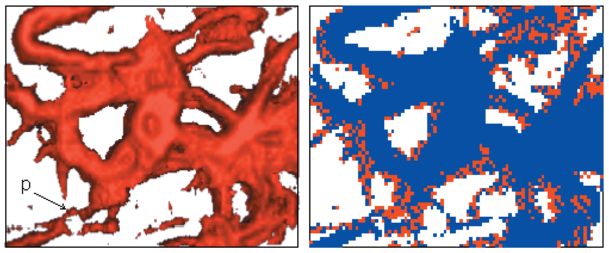

Figure 3: Subcube visualized by VTK (left) and (in form of cubes) by OpenGL (right).

relative location and distribution of junctions to describe astrocytes (and, to some extent, blood vessels). Basically, a junction is defined by arcs of the skeleton (details follow further below).

We subdivide the volume into subcubes (i.e., equally sized small cubes) and we analyze the number of junctions and (different types of) distances between them. Figure 3 shows a subcube of data of the binarized volume shown in Figure 2. As an example of a local configuration, at the voxel labeled by p we have that two non-connected parts "visually overlap" in 3D with respect to the chosen viewing direction.

The results of thinning for the subcube in Figure 3 are shown in Figure 4. In the left image (result of the original algorithm) we used labels to indicate the different parts in the resulting skeletons at some places. The modified algorithm reduces the length of arcs with endpoints $p1$,$p2$,...,$p7$. This is not a disadvantage for the following processes because we are interested to identify junctions, and in measurements between them. Both algorithms deliver topologically equivalent skeletons. We decided for use of the modified algorithm as described in [2].

Our thinning algorithm delivers a skeleton, and (due to the topological invariance of our thinning algorithm) there exists a bijective mapping between the components of the skeleton and the components in the binarized volume.

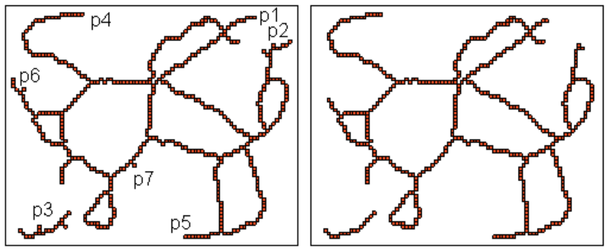

Figure 4: 3D skeletons of original (left) and modified (right) thinning algorithm.

4 Branch Nodes and Junctions

Each component of a skeleton consists of *digital arcs*, defined by two endvoxels, and having no intersection with any other arc of the skeleton. (See [3] for used approaches and basic definitions in digital geometry.) We may use different metrics to measure the length of these arcs (measured between both endvoxels).

We distinguish between three types of voxels in a 3D skeleton: branch voxels, regular voxels, and endvoxels. An arc starts with either a branch voxel or an endvoxel of the skeleton, ends with another branch voxel or another endvoxel of the skeleton, and it consists of a finite number of connected regular voxels. Several branch voxels may cluster together to form a *junction*. Figure 5 displays a junction consisting of three branch voxels, where each corresponds to an endvoxel of one arc. For each of these branch voxels, we need to identify the "opposite" endvoxel of "its" arc.

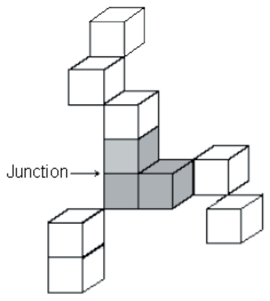

Figure 5: Example of a junction.

4.1 Definitions

We use common adjacency definitions: 4-, 8- (2D), 6-, 18-, 26- (3D) for the grid point model, and 0-, 1- (2D), 0-, 1-, 2- (3D) for the grid cell model. Any of these adjacency relations A_α, $\alpha \in \{0, 1, 2, 4, 6, 8, 18, 26\}$, are irreflexive and symmetric. The *α-neighborhood* $N_\alpha(p)$ of a pixel (voxel) p includes p and its α-adjacent pixels (voxels).

Concepts for describing curve points in a continuous space are applied to the 3D digital space (see [3]). P. Urysohn in 1923 and K. Menger in 1932 proposed equivalent definitions for simple curves (arcs) based on the notion of the branching index of a point in a curve (arc). The *branching index* of a point in a curve was defined as follows:

Let p be a point, ε be a positive real, and $U_\varepsilon(p)$ be the ε-neighborhood of p. A curve γ has branching index $m \geq 0$ at $p \in \gamma$ iff, for any $r > 0$, there is a $\varepsilon < r$ such that the cardinality of the γ-frontier of $U_\varepsilon(p) \cap \gamma$ is at most m.

A *simple curve* is a curve in which every point p has branching index 2. A *simple arc* is either a curve in which every point p has branching index 2 except for two endpoints, which have branching index 1, or a simple curve with one of its points labeled as an endpoint.

Above we listed the three different types of voxels of a 3D curve skeleton. Formally we define these as follows; a voxel p of a 3D curve skeleton S is

- a regular voxel if p has branching index 2;
- a branch voxel if p has branching index of at least 3;
- an endvoxel of S if p has branching index 1;
- a singular voxel if p is either a branch voxel or an endvoxelof S.

In a 3D curve skeleton S, we may have 0-adjacent branch voxels (see Figure 5), and a 0-region of branch voxels of S is called a *junction.* Note that a junction is a non empty 0-connected set of branch voxels. A single branch voxel also represents a junction (with cardinality one).

The *branching index of a junction* J of a skeleton S is the number of regular voxels in S, which are 0-adjacent to one of the branch voxels in J.

It follows that a junction has a branching index greater than 2. For example, the branching index of the junction shown in Figure 5 is 3. The complexity of a junction is measured by its branching index.

A *node* of a skeleton S is either a junction or an endvoxel of S.

The following definition is useful for determining the geometric position of a junction of an arc. Let J be a junction, n be the number of branch voxels p_i constituting J, with $p_i = (x_i, y_i)$, $1 \leq i \leq n$. The *centroid* $c(J)$ of J is a 3D point with coordinates:

$$x = \frac{\sum_{i=1}^{n} x_{p_i}}{n}, y = \frac{\sum_{i=1}^{n} y_{p_i}}{n}, z = \frac{\sum_{i=1}^{n} z_{p_i}}{n} \quad (1)$$

We identify the geometric position of a junction with that of its centroid.

4.2 Algorithms

We uniquely assign labels to all nodes of a skeleton. (All branch voxels of one junction obtain the same label.) This results into labeled nodes of an undirected graph, where edges of the graph correspond to digital arcs between nodes of the skeleton. See Figure 6 for an example.

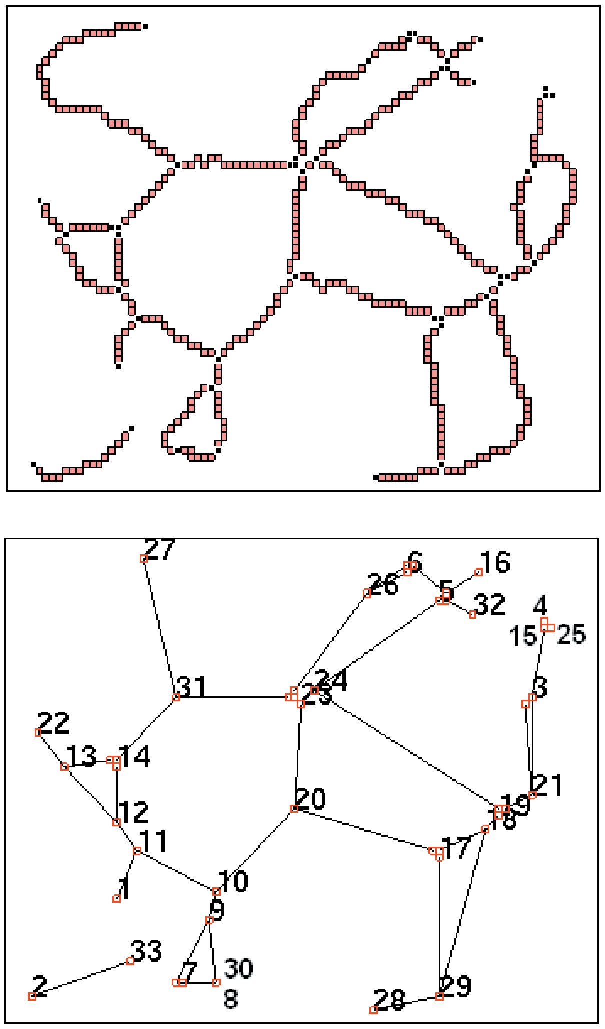

Figure 6: Above: a skeleton. Below: undirected graph after labeling

Due to using only one label for all branch voxels of one junction, different arcs may start from different branch voxels which all have the same label. For example, B_{14} in Figure 6 consists of three branch voxels. Each of them is an endvoxel of a digital arc. We use the centroid of B_{14} as *geometric endpoint* for all these three arcs for the calculation of the Euclidean distance to other nodes.

Skeletons of brain tissue contain a large number of singular voxels and connecting arcs. For accurate length measurements we apply a second labeling process where all arcs are uniquely labeled. All voxels on one arc get the same label; each branch voxel is mapped (say, randomly) to exactly one arc. After these assignments, we apply a (global) DSS-based length measurement (see [3]), where we decided for algorithm **DR1995** as published in [1].

For testing alternatives, we also performed distance measurements simply by using the Euclidean distance between geometric positions of junctions (ignoring the curvedness of arcs), and the number of voxels between junctions (as a simple local length measure, known to be not multigrid convergent to the correct length). The Euclidean distance is indeed only a rough estimation of the length of the digital arc because it only returns the length of the straight line between junctions. The DSS algorithm cuts an arc into a set of digital straight segments and the total length is the sum of the lengths of these segments. We have chosen this algorithm because, besides it general theoretical benefit of being multigrid convergent to the true length, it also proved to be more relevant for characterizing distributions of astrocytes.

The distances between pairs of singular voxels are now the weights in our undirected graph. Based on the cost matrix for this weighted graph together with the coordinates of all singular voxels we can apply traditional algorithms from graph theory (such as algorithms for calculating the minimum path between any two nodes, algorithms for determining the total weight of the minimum spanning tree, or algorithms for finding the diameter of the graph and so on) for further analysis.

We determine the *uniformity* of junctions as follows. The volume data are divided into a set of subcubes (small cubes of identical size). For a fixed branching index j, we count the number of junctions in each cube having branching index j. If the number of junctions with branching index j is equal in every subcube then we say that junctions

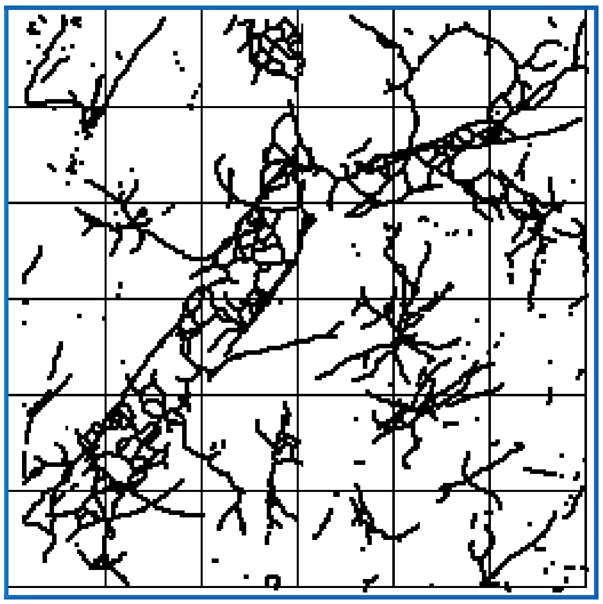

Figure 7: 3D skeleton of the binarized volume shown in Figure 2.

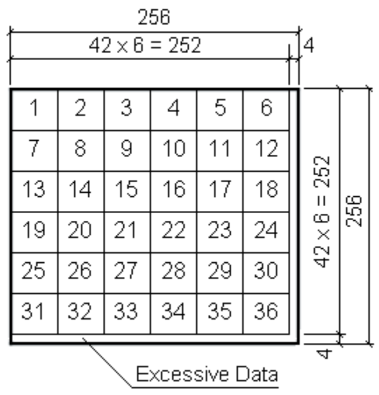

Figure 8: Example of numbered subcubes.

of branching index j are *uniformly distributed* in the whole volume. The deviation from this ideal case characterizes non-uniformity.

For the description of *density* of junctions we calculate the shortest path between (unordered) pairs of distinct junctions with the same branching index j. The shorter the path, the more *dense* are the junctions positioned in 3D space. The total number of junctions in a subcube is a (simple) expression of density of junctions in this subcube.

4.3 Experiments

This short paper only allows to report about a selection of experimental results. For experiments using the number of junctions we subdivided the volume into uniformly sized, disjoint subcubes. For the data set shown in Figure 1, we used 36 subcubes, all of size 42^3. This also generates some excessive data, see Figure 7. (Sliding subcubes experiments are not reported in this paper.) We report about results for this data set.

We analyze the total number of junctions with a branching index between three and seven. In this data set, about 70% of all junctions have branching

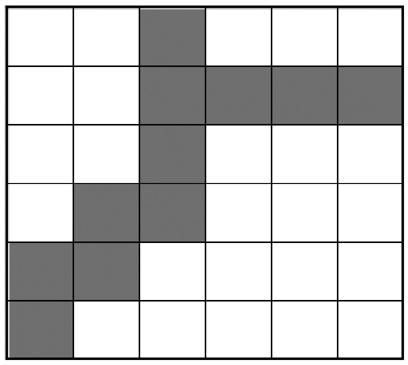

Figure 9: Location of main blood vessel shown as gray cubes

Branching Index j	Number of Bj in Colored Cubes (A1)	Number of Bj in all cubes (A2)	A1 / A2
j = 3	150	276	54.3%
j = 4	53	85	62.4%
j = 5	16	21	76.2%
j = 6	5	7	71.4%
j = 7	2	2	100%
$3 \leqslant j \leqslant 7$	226	391	57.8%

Table 1: Number of junctions per branching index in A1 and A2.

index three, and only two junctions have index seven. The shaded cubes in Figure 9 are identified by numbers of junctions; they coincide with the location of the main blood vessel. These cubes are characterized by a large number of junctions: more than 50% of all junctions (for each branching index between three and seven) are located in these cubes.

Table 1 presents the total number of junctions per branching index for the gray cubes in total (A1), and for the whole volume (A2).

We counted the number of junctions of equal types per cube to find out how they are distributed in the volume. Obviously, they are not (ideally) uniformly distributed (see Table 2) in the whole volume. Most of them are located close to the blood vessel. Subcubes 14, 23 and 29 are not incident with the blood vessel, and they also have a large number of junctions as shown in the diagram.

Experiments based on length measurements showed in general similar results for identifying subcubes with "higher structural complexity". The total length of all arcs increases in subcubes closer to blood vessels. We illustrate again for our example data set. Each subcube shown in light or

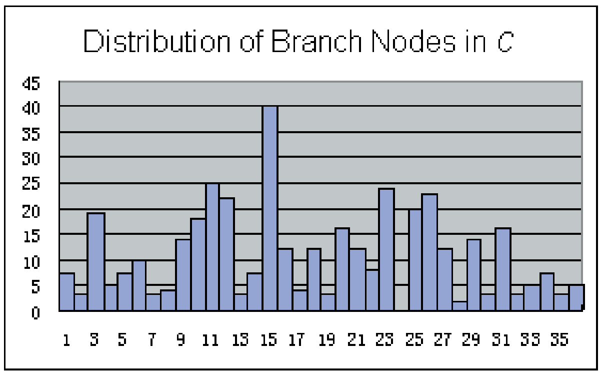

Table 2: The x-axis represents the numbers of subcubes, and the y-axis represents the number of junctions.

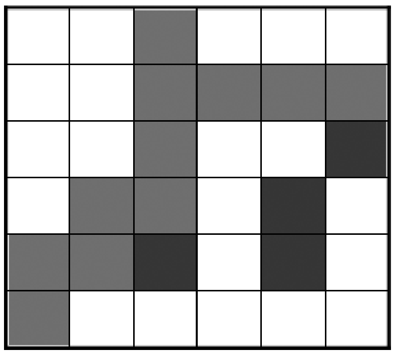

Figure 10: The total length in grey subcubes is greater than 195.

dark gray in Figure 10 has a total length of above 195.

Cubes 18, 23, 27 and 29 are dark grey but not incident with the blood vessel.

In the illustrated experiment we divided the whole volume into pairwise disjoint cubes. This subdivision cuts digital arcs of the skeleton and it produces artificial endvoxels. It can also cut junctions. These type of endvoxels are not problematic for length measurements because they are endvoxels of arcs in a subcube. Cutting junctions results in a different complexity and a different number of junctions. This problem can be solved by considering a sliding cube of fixed size that moves through the volume: after each move of fixed step size we count the number of junctions on the length of arcs. These two features label the center of the subcube at the given position. Allowing different sizes for sliding subcubes is a further way for more detailed studies. Approaches and algorithms are defined and implemented; in a next step quantitative classifications need to be evaluated by medical experts.

5 Conclusions

We propose a sequence of processing steps to identify and quantify junctions in 3D images. Distribution of junctions in subcubes is characterized by numbers (for separate branching indices) and total length of arcs between nodes. Distances between junctions were measured based on different metrics. The result is (in each case) a weighted undirected graph (for each subcube) which can be used to apply graph algorithms. Experimental results support the hypotheses that there is a close relationship between distributions of junctions and distribution of astrocytes.

Acknowledgment

Input data and medical advice was provided by the Biomedical Image Research Unit (BIRU) of the School of Medicine of The University of Auckland; our sincerely thanks to Colin Green and his colleagues. The authors also acknowledge the discussions with Reinhard Klette which proved to be very helpful.

References

[1] I. Debled-Rennesson and J.-P. Reveilles. A linear algorithm for segmentation of digital curves. *Int. J. Pattern Recognition Artificial Intelligence*, **9**:635–662, 1995.

[2] G. Klette and M. Pan. 3D topological thinning by identifying non-simple voxels, In Proc. *10th IWCIA*, pages 164–175, LNCS 3322, Springer, Berlin, 2004.

[3] R. Klette and A. Rosenfeld. *Digital Geometry – Geometric Methods for Digital Picture Analysis.* Morgan Kaufmann, San Francisco, 2004

[4] K. Palagyi and A. Kuba. A 3D 6-subiteration thinning algorithm for extracting medial lines. *Pattern Recognition Letters*, **19**:613–627, 1998.

[5] K. Palagyi, E. Sorantin, E. Balogh, A. Kuba, C. Halmai, B. Erdohelyi, and K. Hausegger. A sequential 3D thinning algorithm and its medical applications. In Proc. *Information Processing Medical Imaging*, page 409–415, LNCS 2082, Springer, Berlin, 2001.

[6] K. Palagyi, J. Tschirren, E. A. Hoffman, and M. Sonka. Assessment of intrathoracic airway trees: methods and in vivo validation. In Proc. *CVAMIA-MMBIA '04*, pages 314-352, LNCS 3117, Springer, Berlin, 2004.

[7] M. Pan and G. Klette. A revision of a 3D skeletonization algorithm. Technical Report CITR-TR-143, Computer Science Department, The University of Auckland, Auckland, New Zealand.