Connected Components in 3D Neighbourhoods

Gunilla Borgefors
Centre for Image Analysis, Swedish University of Agricultural Sciences
Lägerhyddvägen 17, SE-752 37 Uppsala, Sweden
gunilla@cb.uu.se

Ingela Nyström
Centre for Image Analysis, Uppsala University
Lägerhyddvägen 17, SE-752 37 Uppsala, Sweden
ingela@cb.uu.se

Gabriella Sanniti di Baja
Istituto di Cibernetica, CNR
Via Toiano 6, IT-80072 Arco Felice (Napoli), Italy
gsdb@imagm.na.cnr.it

Abstract

We have developed a 3D connected component counting and labelling algorithm suitable for small neighbourhoods. Such an algorithm is useful in many voxel classification tasks. Its main advantage becomes apparent for large size images, when counting connected components in the local neighbourhood of every voxel is necessary. We have used the algorithm in a skeletonization algorithm for volume objects, that repeatedly counts connected components in the local neighbourhood of each skeletal voxel. Our algorithm is simple and fast; it identifies one component at a time in a recursive manner. We give solutions to all three connectivity cases in the $3 \times 3 \times 3$ neighbourhood. Extended connectivities and neighbourhoods are considered both in 2D and 3D.

1 Introduction

Connected component labelling groups the object elements of a digital image into regions by assigning the same label to adjacent elements. Connected component labelling facilitates further image analysis tasks: the regions can be counted, they can be individually manipulated and their properties can be measured. General 2D connected component labelling has been thoroughly investigated in the past, see for example [7, 8].

Labelling object components is useful not only when working with large size images to single out and count the individual regions, but also when counting the number of object components in small neighbourhoods. That is important, e. g., when designing topology preserving removal operations for thinning transformations. In 2D, the 4- and 8-connected components in the special case of a 3×3 neighbourhood can be counted using the crossing number [9], or the connectivity number [12], respectively. There are no simple generalisations of these approaches to higher dimensions or to neighbourhoods larger than 3×3.

General 3D connected component labelling algorithms, see for example [10, 11], could be used to count components in small neighbourhoods. However, they are too tedious, especially when counting connected components in the local neighbourhood of every voxel in a large image. We have used this approach in a skeletonization algorithm for volume objects, that counts connected components in the local neighborhood of each skeletal voxel in some of the steps. The time spent on the connected component counting step was a significant part of the total time of the skeletonization algorithm.

One approach to count connected components in local 3D neighbourhoods, can be found in [1], where graph theory is used. The algorithm is differentiated into a number of special cases. Another approach, that is efficient, can be found in a recent paper, see [6], where the $3 \times 3 \times 3$ neighbourhood is divided into octants. It can, however, only handle one of the three connectivity cases possible in the $3 \times 3 \times 3$ neighbourhood.

We propose a new 3D connected component counting and labelling algorithm suitable for small neighbourhoods. Such an algorithm is useful in many voxel classification tasks. It is specialized for local neighbourhoods, as information about the configuration of the voxels in the neighbourhood is used. In this paper we give solutions in the $3 \times 3 \times 3$ neighbourhood (all three connectivity cases). The same algorithm can be used for other local neighbourhoods, as when calculating the *topological numbers* defined in [1], once the appropriate configuration tables are built. The algorithm does not only count connected components, but actually performs a labelling of the neighbourhood. It is straightforward to understand and implement; it identifies one component at a time in a recursive manner. Hence, it is simple and fast.

2 Definitions

The central voxel in a $3 \times 3 \times 3$ neighbourhood has three types of neighbours:

- 6 voxels are sharing a *face* with the central voxel
- 12 voxels are sharing an *edge* with the central voxel
- 8 voxels are sharing a *point* with the central voxel

Face-neighbours define 6-connectivity. Face- and edge-neighbours define 18-connectivity. Face-, edge-, and point-neighbours define 26-connectivity. The 6-connectivity corresponds to 4-connectivity in 2D, and the 26-connectivity corresponds to 8-connectivity in 2D. If 26-connectedness is selected for the object, 6-connectedness must be used for the background, and *vice versa*.

Consider a $3 \times 3 \times 3$ neighbourhood as a small image. Our algorithm performs a connected component counting and labelling of the objects in this binary $3 \times 3 \times 3$ image. The components can be n-connected, where $n \in \{6, 18, 26\}$. If the aim is to count connected components in the neighbourhood of a voxel, (which is different from counting connected components in an image), the central voxel should be set to 0, and reset correctly afterwards. If the connected components of the background is to be counted, the input should be the inverse of the image.

3 Counting and Labelling 3D Connected Components

Input:

A (small) binary image.
L, a list of n-neighbours for every voxel.
V, a one-dimensional array with one position per voxel. Initialize V to $\overline{0}$. V will be used to recursively identify the voxels belonging to the same n-connected component.

Main loop:

Loop through all voxels. If the current voxel is an object voxel and not yet labelled, it is given a new label g. The corresponding position in V is set to 1. Then the recursive function is called. (If the current voxel already is labelled, nothing has to be done.)

Recursion:

Recursive inspection of V until component g is complete:
Loop through V. Once a 1 is found, the neighbours of the corresponding voxel (found in L) are checked. If a neighbour is an object voxel and not yet labelled, it is given the current label g and the corresponding position in V is set to 1 (as its neighbours have to be checked). Then the current position in V is marked as "done", e. g. set to 2. The recursive inspection of V continues until no 1 is found in V.

Output:

An image with its connected components labelled consecutively. The highest label indicates the number of n-connected components in the image.

4 Lists of n-Neighbours

The recursive 3D connected component counting and labelling algorithm described above is general in the sense that it does not change with image size or connectivity (or, indeed, with dimension). It is, however, only reasonable to use it for small images (local neighbourhoods). The only difference from case to case are the lists of neighbours, L. We give the lists for $3 \times 3 \times 3$ images.

The voxels in a $3 \times 3 \times 3$ image have a number of n-neighbours, where $n \in \{6, 18, 26\}$. Each type of connectivity needs its own look-up table, where a voxel and its n-neighbours are listed in each row. The two look-up tables for 6- and 18-connectivity are subsets of the look-up table for the 26-connectivity, see Table 1 and Figure 1. Face-, edge-, and point-neighbours are written in **bold**, normal, and *italic*, respectively. The look-up table for the 18-connectivity case includes only face- and edge-neighbours. The look-up table for the 6-connectivity case includes only face-neighbours. The sizes of the tables are 6×27, 18×27, and 26×27, respectively, (maximum number of neighbours $\times$ number of voxels in image). All three $3 \times 3 \times 3$ connectivities are listed, even though the 18-connectivity is seldom used.

Corresponding lists could be prepared for larger images.

5 Computation Complexity

The recursion (checking the neighbours of neighbours of neighbours...) is seldom deep as the components in tiny neighbourhoods like these, can neither be very large nor very complex. Computation time is saved in the recursive inspection of V, as voxels already completed in the main loop do not have to be considered; the loop through V is thus getting shorter and shorter. Another shortening of loops follows from the configuration of the list of neighbours, L, which is implemented as a two-dimensional array. The number of neighbours of a certain voxel may not fill a row in the array, see Table 1. The other entries are filled with the value -1 (or some other value which does not correspond to a neighbour), and the loop through the row is interrupted as soon as that value is found.

6 Implementation

Our implementation is written in C and running on a DEC Alpha (a standard UNIX workstation). When the skeletonization algorithm in [4] is using this algorithm in some of the steps instead of a general 3D connected component labelling algorithm, see [11], the computation time is reduced from hours to a few minutes for objects in $128 \times 128 \times 128$ (2 Mbyte) images.

In Figure 2 an example of 3D connected component labelling of a $3 \times 3 \times 3$ 26-neighbourhood is shown.

7 Conclusions

We have implemented a recursive 3D connected component counting and labelling algorithm that can be a helpful tool and give major speed-up, when used instead of more general algorithms. It is designed to work in the local neighbourhood of a voxel. Its main advantage is shown when counting connected components repeatedly (for example as many times as there are object voxels) in large size images, but only in the local neighbourhood of each voxel.

Our algorithm is suitable for various voxel classification tasks in local neighbourhoods. We have found it useful when reducing volume objects to skeletal representations, see [3, 4]. On a 3D surface, the voxels can be classified as border voxels or internal voxels by checking how the surface crosses a $3 \times 3 \times 3$ neighbourhood of the voxels. This may be done by counting the 6-connected components of background voxels and the 26-connected components of skeletal voxels in the neighbourhood, which was a bottleneck when using a general 3D connected component labelling algorithm. We have limited the implementation to the $3 \times 3 \times 3$ neighbourhood, as for our purposes this was the desired neighbourhood. However, our recursive connected component counting and labelling algorithm is easily extended to other dimensions and other neighbourhoods, once the appropriate lists of neighbours are built. It can be applied in the 2D case to count 4- or 8-connected components in the 3×3 neighbourhood, even if it is easier using methods based on the crossing or connectivity numbers. Unlike these methods, our algorithm can be extended to count "knight-connected" components in the 5×5 neighbourhood, where connection is allowed through "the knight's move" in chess, see [5]. The extended 5×5 neighbourhooud has been used as a basis for a distance transform and a medial axis concept, see [2].

Extended 3D neighbourhood configurations can be interesting when investigating 3D digital topology in general. For example, our algorithm is well-suited for the calculation of the topological numbers defined in [1]. Tables for these neighbourhoods describing the correct connectivities, similar to Table 1, could easily be built.

Acknowledgments

We thank Prof. Ewert Bengtsson and Dr. Bo Nordin at the Centre for Image Analysis for provided support.

References

[1] G. Bertrand. Simple points, topological numbers and geodesic neighbourhoods in cubic grids. *Pattern Recognition Letters*, 15:1003–1011, Oct. 1994.

[2] G. Borgefors. Centres of maximal discs in the 5-7-11 distance transform. In *Proceedings of 8th Scandinavian Conference on Image Analysis, Tromsø, Norway*, pages 105–111, 1993.

[3] G. Borgefors, I. Nyström, and G. Sanniti di Baja. Surface skeletonization of volume objects. In P. Perner, P. Wang, and A. Rosenfeld, editors, *Proceedings of SSPR'96 - Leipzig: Advances in Structural and Syntactical Pattern Recognition*,

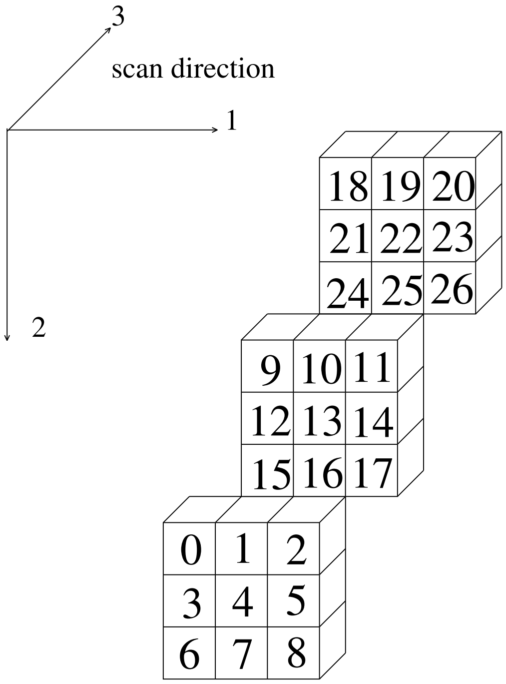

Figure 1: Numbering of voxels in a $3 \times 3 \times 3$ image slice by slice.

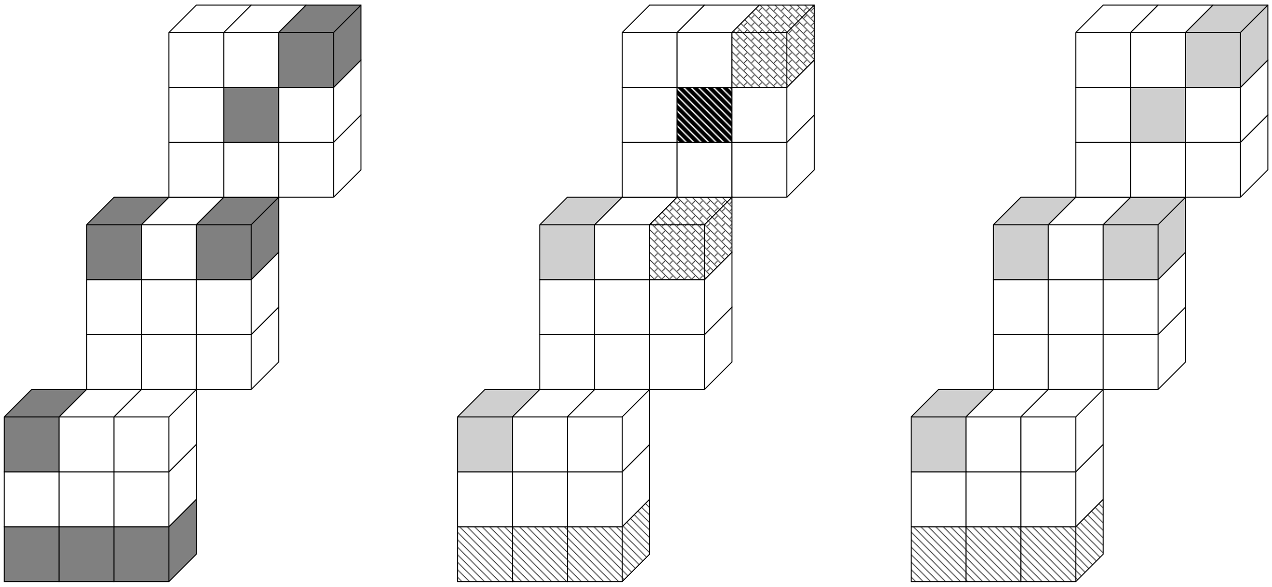

Figure 2: An example of 3D connected component labelling of a $3 \times 3 \times 3$ 26-neighbourhood shown slice by slice. Object voxels in grey, background voxels in white, and labelled voxels in different grey-rasters. Left: binary input. Middle: four 6-connected components. Right: two 26-connected components.

Table 1: List of 26-neighbours for the 27 voxels in a $3 \times 3 \times 3$ image. See Figure 1 for the numbering. Face-, edge-, and point-neighbours are written in **bold**, normal, and *italic*, respectively. The look-up table for the 18-connectivity case includes only face- and edge-neighbours. The look-up table for the 6-connectivity case includes only face-neighbours.

Voxel	**Neighbours**												
0	**1**	**3**	4	**9**	10	12	*13*						
1	**0**	**2**	3	**4**	5	9	**10**	11	*12*	13	*14*		
2	**1**	4	**5**	10	**11**	*13*	14						
3	**0**	1	**4**	**6**	7	9	*10*	**12**	13	15	*16*		
4	0	**1**	2	**3**	**5**	6	**7**	8	*9*	10	*11*	12	**13**
	14	*15*	16	*17*									
5	1	**2**	**4**	7	**8**	*10*	11	13	**14**	*16*	17		
6	**3**	4	**7**	12	*13*	**15**	16						
7	3	**4**	5	**6**	**8**	*12*	13	*14*	15	**16**	17		
8	4	**5**	**7**	*13*	14	16	**17**						
9	**0**	1	3	*4*	**10**	**12**	13	**18**	19	21	*22*		
10	0	**1**	2	*3*	4	*5*	**9**	**11**	12	**13**	14	18	**19**
	20	*21*	22	*23*									
11	1	**2**	*4*	5	**10**	13	**14**	19	**20**	*22*	23		
12	0	*1*	**3**	4	6	*7*	**9**	10	**13**	**15**	16	18	*19*
	21	22	24	*25*									
13	*0*	1	*2*	3	**4**	5	*6*	7	*8*	9	**10**	11	**12**
	14	15	**16**	17	*18*	19	*20*	21	**22**	23	*24*	25	*26*
14	*1*	2	4	**5**	*7*	8	10	**11**	**13**	16	**17**	*19*	20
	22	**23**	*25*	26									
15	3	*4*	**6**	7	**12**	13	**16**	21	*22*	**24**	25		
16	*3*	4	*5*	6	**7**	8	12	**13**	14	**15**	**17**	*21*	22
	23	24	**25**	26									
17	*4*	5	7	**8**	13	**14**	**16**	*22*	23	25	**26**		
18	**9**	10	12	*13*	**19**	**21**	22						
19	9	**10**	11	*12*	13	*14*	**18**	**20**	21	**22**	23		
20	10	**11**	*13*	14	**19**	22	**23**						
21	9	*10*	**12**	13	15	*16*	**18**	19	**22**	**24**	25		
22	*9*	10	*11*	12	**13**	14	*15*	16	*17*	18	**19**	20	**21**
	23	24	**25**	26									
23	*10*	11	13	**14**	*16*	17	19	**20**	**22**	25	**26**		
24	12	*13*	**15**	16	**21**	22	**25**						
25	*12*	13	*14*	15	**16**	17	21	**22**	23	**24**	**26**		
26	*13*	14	16	**17**	22	**23**	**25**						

pages 251–259. Springer-Verlag, Berlin Heidelberg, 1996.

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

[5] P. P. Das and B. N. Chatterji. Knight's distance in digital geometry. *Pattern Recognition Letters*, 7(4):215–226, Apr. 1988.

[6] Y. Ge, D. R. Stelts, and D. J. Vining. 3D skeleton for virtual colonoscopy. In K. H. Höhne and R. Kikinis, editors, *Proceedings of* 4^{th} *VBC'96: Visualization in Biomedical Computing*, pages 449–454. Springer-Verlag, Berlin Heidelberg, 1996.

[7] B. Kruse. A fast algorithm for segmentation of connected components in binary images. In *Proceedings of* 1^{st} *Scandinavian Conference on Image Analysis, Linköping, Sweden*, pages 57–63. Studentlitteratur, 1980.

[8] A. Rosenfeld and A. C. Kak. *Digital Picture Processing*, volume 2, chapter 11. Academic Press, Inc., New York, 2 edition, 1982.

[9] D. Rutovitz. Pattern recognition. *Journal of the Royal Statistical Society*, 129:504–530, 1966.

[10] L. G. Shapiro. Connected component labeling and adjacency graph construction. In T. Y. Kong and A. Rosenfeld, editors, *Topological Algorithms for Digital Image Processing*, pages 1–30. Elsevier Science, North-Holland, 1996.

[11] L. Thurfjell, E. Bengtsson, and B. Nordin. A new three-dimensional connected components labeling algorithm with simultaneous object feature extraction capability. *CVGIP: Graphical Models and Image Processing*, 54(4):357–364, July 1992.

[12] S. Yokoi, J.-I. Toriwaki, and T. Fukumura. An analysis of topological properties of digitized binary pictures using local features. *Computer Graphics and Image Processing*, 4(1):63–73, 1975.

This is a copy (references updated) of

G. Borgefors, I. Nyström, and G. Sanniti di Baja:
"Connected components in 3D neighbourhoods"
In *Proceedings of* 10^{th} *Scandinavian Conference on Image Analysis, Lappeenranta, Finland*,
pages 567–572, June 1997.