

Erosion and Dilation on 2D and 3D Digital Images: A new size-independent approach

J. Rodriguez * and D. Ayala

Universitat Politècnica de Catalunya, Software Department (Departament de Llenguatges i Sistemes Informàtics)
Av. Diagonal 647, 08028 Barcelona, Spain
Email: [jrodri,dolorsa]@lsi.upc.es

Abstract

This paper presents a new approach to achieve elementary neighborhood operations on both 2D and 3D binary images by using the Extreme Vertices Model (EVM), a recent orthogonal polyhedra representation applied to digital images. The operations developed here are erosion and dilation. In contrast with previous techniques, this method do not use a voxel-based approach but deal with the inner sections of the object. It allows to build an image size-independent algorithm. The proposed method also admits the use of structuring elements of arbitrary size and allows to treat 2D and 3D images in identical way using the same algorithm.

1 Introduction

A 3D digital image (or volume dataset) can be represented as a map $f : Z \otimes Z \otimes Z \rightarrow R$, in such a way that every point is assigned a value representing its color. In a binary image the image set is $\{0, 1\}$. In [10] a 3D digital image is defined as an union of voxels i.e. upright unit cubes whose vertices have integer coordinates. Analogously, a 2D digital image is defined as a union of pixels. In order to generalize this term, we will call the elements in a n -dimensional digital image n -voxels.

The most extensive class of binary images processing operations is sometimes collectively described as morphological operations [5]. This includes erosion and dilation which are the base of most morphological operations such as opening, closing or hit-or-miss transform. The operations are used in several tasks such as elimination of small spurious objects, smoothing of object boundaries,

object separation and elimination of small holes in objects. All of the published algorithms for these elementary operations have a voxel-based approach and, therefore, their performance depends on the size of the image.

In this work we propose a new approach to perform erosion and dilation operations using the Extreme Vertices Model (EVM) as a representation scheme of binary data. This representation model allows to describe 2D and 3D binary images in a very concise way and to build erosion and dilation algorithms which are capable to process both 2D and 3D digital images in identical way. Our proposal admits the use of box-shaped structuring elements of arbitrary size. Finally, the algorithms developed have a performance which is independent of the amount of n -voxels of the input set.

The section below surveys previous work to improve erosion and dilation algorithms and introduces coarsely our proposal. Then, in Section 3 we review the EVM, showing those definitions and properties used in this work and discussing the ability and performance of the EVM as a concise description of digital images. The following section describes our proposal for both erosion and dilation operations, using the EVM. Finally, some experimental results, conclusions and future work are presented.

2 Review of Erosion and Dilation

2.1 Erosion and dilation

Let A and B be sets in Z^d , $d > 0$. Let $(B)_x$ denote the *translation* of B by x and let \hat{B} denote the *reflection* of B with respect to its origin.

The erosion of A by B , $A \ominus B$, is defined as [5]:

$$A \ominus B = \{x | (B)_x \subseteq A\} \quad (1)$$

*This author has been supported by a grant from CONICIT (Venezuelan Council of Research in Science and Technology)

which, in words, says that the erosion of A by B is the set of all points x such that B, translated by x, is contained in A.

The *dilation* of A by B, $A \oplus B$, is defined as [5]:

$$A \oplus B = \{x | (\hat{B}_x \cap A \neq \emptyset)\} \quad (2)$$

Thus, the dilation of A by B is the set of all x displacements of the origin of \hat{B} such that \hat{B} and A overlap by at least one nonzero element.

Set B is commonly referred to the *structuring element* in erosion, dilation and in other morphological operations. These operations can also be expressed in terms of the Minkowski operators [11].

From the above definitions, erosion and dilation can be described simply in terms of adding or removing n-voxels from the binary image, according to certain rules which depend on the pattern of the neighboring n-voxels and the size and shape of the structuring element. Therefore, erosion can be explained as a process where the interior of the object is shrunk as much as the size of the structuring element, whereas dilation is a process that elongates the interior of the object.

2.2 Related work

A number of different approaches to improve 2D algorithms have been reported. All of them use a more suitable representation of the image data to improve the performance. Boomgaard and van Balen [14] use a bitmap representation of binary images because the CPU operates on 32 pixels in parallel. Verwer and van Vliet [15] use a queue of the contour pixels to avoid traversing all the image each time. Parker [8] uses maps of distance and Young et al. [16] used a run-length representation of the image.

There are also 3D approaches. Zuiderveld [17] proposes a method for successive dilation operations of 3D images by using a queue of contour voxels, following the idea of Verwer and Vliet [15]. Nevertheless, this proposal demands an overhead of memory. Thurfjell et al. [12] proposes a method for fast erode and dilate using the semiboundary representation [13]. These two proposals only traverse the contour n-voxels in the 3D data set. However, they can not use structuring elements of arbitrary size. Furthermore, the algorithms for 2D and 3D data sets vary radically. In other words, they need different algorithms for each case. Finally, in all of

these methods the cost, both in terms of memory requirements and in terms of algorithmic efficiency, increases when the binary data size increases. In [7] the authors present an approach to improve the efficiency of erosion and dilation algorithms. The cost of these algorithms increases with square (cubic) complexity with respect to the size of the structuring element in the 2D (3D) case. Their approach, then, reduces the complexity by decomposing the structuring element into several smaller ones.

2.3 Our proposal

The conventional techniques and their improvements perform the process using a voxel-based approach. In contrast, our proposal identifies the inner sections of the object directly from the EVM representation, then it operates (shrinks or elongates) these sections and, afterwards, computes the new EVM representation of the transformed object.

In contrast with previous implementations of erosion and dilation, our method treats 2D and 3D images with the same algorithm. The algorithm is a recursive process where the recursion is done over the dimension. It decomposes the 3D image into 2D sections which are themselves considered 2D images and, then, decomposes these 2D images into 1D sections. This is the base case of the recursion and the algorithm actually performs 1-dimensional transformations. Our approach admits the use of box-shaped structuring elements of arbitrary size.

The visualization of the resulting object is as easy as the visualization of the original one, with no overhead of memory. Finally, the cost of these operations is independent of the amount of n-voxels of the input data set. Instead, it is associated with the orthogonal shape of the object allowing to operate large sets of co-planar adjacent n-voxels in a single step. Moreover, our approach is also independent of the size of the structuring element.

3 Review of the EVM

Orthogonal polyhedra (OP) are polyhedra with all their faces oriented in three orthogonal directions and are also named rectangular or isothetic. Orthogonal Pseudo-polyhedra (OPP) will refer to regular and orthogonal polyhedra with a non-manifold boundary. Figure 1 shows an OPP with a non-manifold vertex and three non-manifold edges.

The *Extreme Vertices Model (EVM)* is a very concise representation scheme in which any OP or OPP can be described using only a subset of its vertices. A digital image is geometrically analogue to an orthogonal polyhedron and can therefore be represented by the EVM. The EVM is actually a complete (non-ambiguous) solid model: it is an implicit *boundary representation (B-Rep)* model, i. e., all the geometry and topological relations concerning faces, edges and vertices of the represented OPP can be obtained from the EVM.

In the following we review some definitions and properties that are needed in this paper. All concepts are more extensively explained and proved in a previous paper [2].

3.1 Definitions

Let P be an orthogonal pseudo-polyhedron (OPP). A *brink* is the maximal uninterrupted segment built out of a sequence of collinear and contiguous two-manifold edges of P . The ending vertices of a brink are called *extreme vertices (EV)*. Figure 1 shows an OPP with a brink from vertex A to vertex E (both extreme vertices). In this brink, vertices B, C and D are non-extreme vertices.

The *EVM* is a representation scheme in which any OPP is described by its (an only its) set of EV.

A *plane of vertices (plv)* is the set of vertices lying on a plane perpendicular to a main axis of P . A *slice* is the region between two consecutive planes of vertices. A *section (S)* is the resulting polygon from the intersection between P and an orthogonal plane. Each slice has its representing section. Figure 1 shows an OPP with its planes of vertices and sections perpendicular to the X axis.

All these definitions can be extended to any dimension [4]. In this paper we are concerned with dimension ≤ 3 . Planes of vertices and sections obtained from a 3D object are, then, 2D orthogonal polygons. From them, we can obtain their 1D lines of vertices and their 2D slices with their corresponding 1D sections. Finally, lines of vertices and 1D sections are 1D objects which are composed of one or several brinks.

In the EVM model the set of EV can be ordered in six possible ways depending on the coordinate values: XYZ, XZY, YXZ, YZX, ZXY, and ZYX. In an XYZ ordered EVM, planes of vertices perpendicular to the X axis appear ordered from low x values to high x values and, in each of them, lines

of vertices parallel to the Y axis appear also ordered from low y values to high y values.

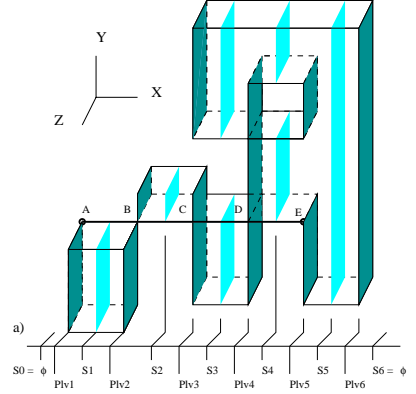


Figure 1: An OPP with a marked brink from vertex A to vertex E. Its planes of vertices and sections perpendicular to the X axis are shown in dark and light grey respectively.

3.2 Properties of the EVM

The first property concerning the EVM is that coordinate values of non-extreme vertices may be obtained from EV coordinates.

The second property allow sections to be computed from planes of vertices and vice-versa:

$$S_0(P) = \overline{S_n(P)} = \emptyset$$

$$\overline{S_i(P)} = \overline{S_{i-1}(P)} \otimes^* \overline{plv_i(P)}, i = 1 \dots n \quad (3)$$

$$\overline{plv_i(P)} = \overline{S_{i-1}(P)} \otimes^* \overline{S_i(P)}, i = 1 \dots n \quad (4)$$

where $\overline{plv_i(P)}$ and $\overline{S_i(P)}$ denote the projections of $plv_i(P)$ and $S_i(P)$ onto a main plane parallel to P , n is the number of planes of vertices and \otimes^* denotes the regularized XOR operation. Note that in order to operate with the projections we need not take into account the coordinate of the extreme vertices that corresponds to the projecting plane.

Applying the definition of the \otimes operation, this last equation can be expressed as:

$$\overline{plv_i(P)} = \overline{S_{i-1}(P)} \otimes^* \overline{S_i(P)}$$

$$= (\overline{S_{i-1}(P)} -^* \overline{S_i(P)}) \cup (\overline{S_i(P)} -^* \overline{S_{i-1}(P)})$$

$$i = 1 \dots n \quad (5)$$

and, thus, we can decompose any plane of vertices into two terms that we will call *forward difference* and *backward difference* (*FD* and *BD* for short).

$$\overline{FD_i(P)} = (\overline{S_{i-1}(P)} -^* \overline{S_i(P)}) \quad (6)$$

$$\overline{BD_i(P)} = (\overline{S_i(P)} -^* \overline{S_{i-1}(P)}) \quad (7)$$

The following property, based on *FD* and *BD*, guarantees that the correct orientation of all faces of *P* can be obtained from its EVM: *FD_i(P)* is the set of faces on *plv_i(P)* whose normal vector points to one side of the main axis perpendicular to *plv_i(P)*, while *BD_i(P)* is the set of faces whose normal vector points to the opposite side. Figure 2 shows an OPP with its sections and with its *FD* and *BD*. This property together with the second one (equations 3 and 4) provide proof that the EVM is a complete B-Rep model.

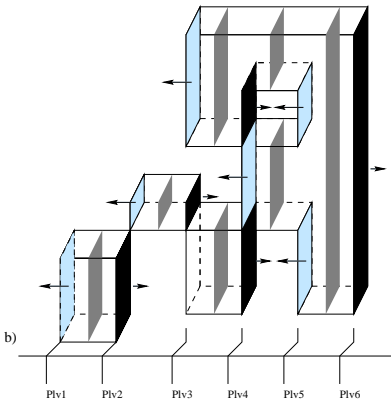


Figure 2: Sections are shown in dark grey, *FD* in black (pointing to the right) and *BD* in light grey (pointing to the left).

The following property concerns the XOR operation: Let *P* and *Q* be two d-D ($d \leq 3$) OPP, having *EVM(P)* and *EVM(Q)* as their respective models, then,

$$EVM(P \otimes^* Q) = EVM(P) \otimes EVM(Q) \quad (8)$$

This property means that the XOR operation works in OD, because it is applied to the EV of the model. Therefore, sections are obtained from planes of vertices and vice-versa by applying the XOR operation to the extreme vertices.

General Boolean operations between OPP can be carried out by applying recursively the same

Boolean operation over the corresponding OPP sections. This algorithm is presented in [1] and consists in a geometric merge between the EVM of both operands which involve as a basic operation the XOR between EV.

3.3 Conversion algorithms

In a previous work [9] a conversion algorithm from a 3D digital image to the EVM is presented. This algorithm consists in a traversal of the set of voxels in the image detecting in a very simple way all the EV, which are stored in the EVM.

The EVM is a B-Rep model which contains the information of all the boundary in an implicit way. Therefore, we can compute the boundary of an EVM representad object from the EVM. In [3] a conversion algorithm from the EVM to a hierarchical B-Rep is presented. A hierarchical B-Rep contains the geometric information of all vertices (coordinates) and faces (oriented normal vectors) and the topological relations *face* : {*edges*} and *edge* : {*vertices*}. The obtained B-Rep model is composed by a relatively few large orthogonal faces instead of a large number of little triangular faces, as in marching cubes like approaches [6], or a large number of little quadrangular faces, as in other block-form approaches as the semiboundary representation (see Figures 3, 4 and 5).

3.4 Performance of the EVM

EVM is a very concise model but, for storage and transmission purposes, it can be further compressed. In [9] several compression techniques applied to EVM have been presented and evaluated. We compared the EVM with the semiboundary representation (SB) [13] because SB was the block-form representation that best performed as we concluded from the consulted literature. Now we present two tables that sum up the comparison between EVM and SB showing that EVM is more concise than SB. The models referred in these tables are shown in Figures 3, 4 and 5.

Table 1 compares the SB and EVM models in terms of memory (bytes).

Table 2 compares the number of faces obtained when the EVM and the SB models are converted to a B-Rep model. In this table also is shown the number of triangular faces for the marching cubes representation (MC).

Memory (bytes)	SB	EVM
boxes(16x16x10)	3438	720
skull(96x96x69)	105979	60924
ctHead(256x256x94)	453016	291136

Table 1: Memory comparisons

BRep(faces)	MC-BRep	SB-BRep	EVM-BRep
boxes(16x16x10)	1349	1236	36
skull(96x96x69)	97467	49030	16407
ctHead(256x256x94)	336283	180288	54570

Table 2: Number of faces

4 Erosion and Dilation using the EVM

Let $EVM(P)$ be the EVM representation of an OPP, P . Let plv_i and S_i be the i th plane of vertices and section of P respectively, $i = 0, 1, \dots, n$.

The erosion (dilation) of P using the EVM is a sequential process that, for each coordinate axis, traverses all the planes of vertices of $EVM(P)$ and computes all its sections of $EVM(P)$; then, it *shrinks* (or *elongates*) these sections and recomputes the new planes of vertices obtaining the EVM of the eroded (dilated) P .

The *shrink* and *elongate* operations are recursive processes which transform the sections only when they become one-dimensional, i.e., a list of collinear brinks. The shrink operation, $Shrink(P)$, is defined by:

$$\begin{cases} Shrink(S_{i-1}) \otimes Shrink(S_i), i = 1 \dots n & dim > 1 \\ Shrink-1D(P) & dim = 1 \end{cases}$$

where dim is the dimension of P , S_i is the i th section of P and $Shrink-1D(P)$ consists in shrinking each brink of the one-dimensional P . The shrink-

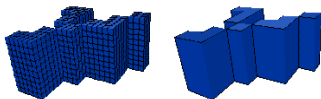


Figure 3: SB-BRep and EVM-BRep of boxes

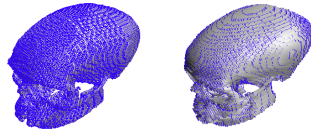


Figure 4: SB-BRep and EVM-BRep of skull

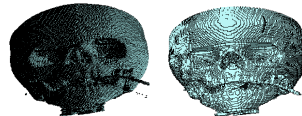


Figure 5: SB-BRep and EVM-BRep of ctHead

ing operation depends on the size of structuring element. An identical recursive process can be defined for $Elongate(P)$.

Applying the properties described by equations 3 to 5 (see 3.2), it is clear that $Shrink(P)$ recomputes all the planes of vertices of P in terms of its transformed sections. All of these recomputed planes of vertices together define the eroded P . Figure 6 illustrates the recursion for the erosion of a single polyhedron in one of the three needed directions.

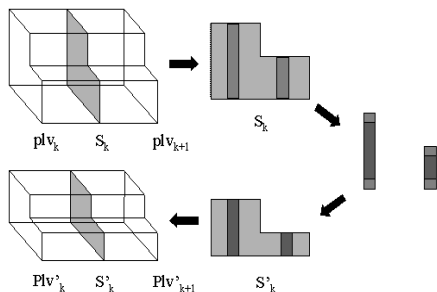


Figure 6: Single-direction erosion of a polyhedron

In this process the structuring element is simply the measure to shorten (or elongate) the brinks at the lowest dimension. The structuring element is not a mask as in the voxel-based approaches. Instead, it is one measure for each dimension of the object. Thus, we can define structuring elements of arbitrary size and also, with different ratios for each di-

rection. Nevertheless, the structuring elements are box-shaped.

The following algorithm shows this recursive process for the erosion operation along one of the main axes (the C axis). Dilation only differs in the implementation of the 1D-transformation, changing the call to *Shrink-1D* for *Elongate-1D*.

```

PROCEDURE C-shrink (INPUT p: EVM, dim: int, ratio: int
OUTPUT q: EVM)
VAR
  plv, S0, S1, ShrunkS0, ShrunkS1, New-plv: EVM; nc: int
ENDVAR
IF (dim = 1) THEN
  Shrink-1D (p, ratio, ShrunkS1);
ELSE
  dim:= dim - 1;
  S0:=  $\emptyset$ ;
  ShrunkS0:= S0;
  DO
    plv:= ReadPlv (p, dim);
    nc:= GetCoord (plv, dim);
    S1:= S0  $\otimes$  plv;
    C-shrink (S1, dim, ratio, ShrunkS1);
    New-plv:= ShrunkS0  $\otimes$  ShrunkS1;
    SetCoord (New-plv, nc);
    PutPlv (New-plv, q, dim);
    S0:= S1;
    ShrunkS0:= ShrunkS1;
  WHILE(NOT(EndEVM(p)))
ENDIF
ENDPROC

```

This algorithm uses the basic operations of the EVM: *ReadPlv* (p, dim) returns the current plane of vertices of p . *PutPlv* (p, q, dim) adds the plane or line of vertices p to the EVM q . *EndEVM* (\cdot) detects the end of the model. The \otimes operation between two sections produces a plane or line of vertices and between a section and a plane or line of vertices produces the next section (see 3.2). Finally, *GetCoord* (p, dim) gets the constant coordinate of $EVM(p)$, which is a plane or a line of vertices, according to its dimension, dim , and *SetCoord* (plv, nc) sets the corresponding coordinate of plv as nc .

It is important to note that the recursive algorithm above is called once for each ABC ordering (ABX, ABY and ABZ) of the EVM. Thus, the transformation of the 1D sections done by the procedure *Shrink-1D* (or *Elongate-1D*), shrinks (or elongates) only the corresponding C-coordinate each time. Procedure *Shrink-1D* shrinks a 1D section which consists of one or more brinks. All brinks of this line are shortened in both extreme vertices by the specified ratio and some of them may become null. Alternatively, in the C-elongate procedure, *Elongate-1D*, elongates all brinks and some

of them can merge (see Figure 7).

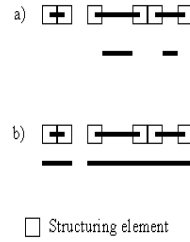


Figure 7: Procedures (a) Shrink-1D and (b) Elongate-1D applied to a 1D section composed by three brinks. In (a) the first brink becomes null and in (b) the two last brinks are merged

5 Experimental Results

Now we present some results obtained with our EVM oriented approach. Figure 8 shows an original image with its corresponding eroded image and Figure 9 shows a progressive erosion of a picture (*magallanes*). Figure 10 shows another example of erosion and dilation using a square structuring element (*test2D*) in which the erosion operation has eliminated a thin part. Figure 11 presents a non-uniform dilation of a CT slice called *testCT*. The dilation is applied only in the X direction and, therefore, the object appears more and more broader in each dilation step but its height remains unchanged. Finally, Figure 12 shows dilation of the frontal part of a skull obtained from CT data (*skull*).



Figure 8: Original image (above) with the corresponding eroded one (below). In the middle, gray pixels would be the eliminated pixels in a voxel-oriented approach



Figure 9: Progressive erosion (*Magallanes*)

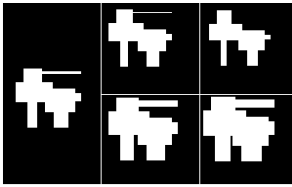


Figure 10: Erosion and dilation of *test2D*

Table 3 presents, for each data set, its resolution, the number of boundary n-voxels of the image, the number of brinks within its EVM representation and the processing time for both erosion and dilation operations. We can note that the processing time increases only when the number of brinks grows.

Table 4 shows a double-size copy of the former data sets. For each one, we have obtained the EVM representation again. We can note that the number of boundary elements has increased a lot. However the processing time remains almost the same because the number of brinks is almost the same too. It proves that our algorithm's performance is image



Figure 11: Non-uniform dilation *testCT*

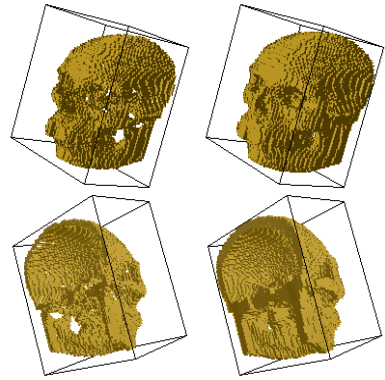


Figure 12: Two views of the dilation of a skull extracted from a CT data set (*skull*)

Data set	#bound	#brinks	Erosion	Dilation
Magallanes 400x159	9692	1173	0.19	0.19
test2D 100x100	2516	14	0.01	0.01
testCT 96x96	2158	224	0.04	0.02
skull 96x41x69	36184	6839	12.39	13.42

Table 3: Processing time (in seconds)

size-independent.

6 Conclusions and future work

Previous authors have presented important improvements of the morphological operations erosion and dilation by using suitable representation of the data, but they must to use different schemes to represent 2D and 3D images. Therefore, they had to build different algorithms for each case. We have developed an algorithm which deals with both cases in identical way because of its recursive behavior. Our proposal, also admits the using of arbitrary size structuring elements and works directly on the EVM representation without using any auxiliary data structure. Furthermore, these operations become image-size independent in our approach. The EVM representation, in our experience, has become a useful representation scheme for digital

Doubles	#bound	#brinks	Eros.	Dilat.
Magallanes (767x318)	37124	1173	0.22	0.21
test2D (200x200)	10064	14	0.01	0.01
testCT (192x192)	8632	224	0.04	0.03
skull (192x82x138)	187592	6859	13.9	14.78

Table 4: Processing time for double-sized images

2D and 3D images and the developed morphological operations increase the potential of this model. We hope, in the future, to complete this work by studying operations as connected component labeling, thinning, detection of non-manifold zones and holes and computing the genus of 2D and 3D digital images, using the EVM model.

Acknowledgements

This work has been partially supported by a CICYT grant TIC99-1230-C02-02.

References

[1] A. Aguilera and D. Ayala. Orthogonal Polyhedra as Geometric Bounds in Constructive Solid Geometry. In C. Hoffmann and W. Bronsvort, editors, *ACM SM'97. Atlanta (USA)*, pages 56 – 67, 1997.

[2] A. Aguilera and D. Ayala. Domain extension for the extreme vertices model (EVM) and set-membership classification. In *CSG'98. Ammerdown (UK)*, pages 33 – 47. Information Geometers Ltd., 1998.

[3] A. Aguilera and D. Ayala. Converting Orthogonal Polyhedra from Extreme Vertices Model to B-Rep and to Alternative Sum of Volumes. *Computing Suppl. Springer-Verlag*, (14):1 – 28, 2001.

[4] O. Bournez, O. Maler, and A. Pnueli. Orthogonal Polyhedra: Representation and Computation. In *Hybrid Systems: Computation and Control*, LNCS 1569, pages 46 – 60. Springer, 1999.

[5] R. González and R. Woods. *Digital Image Processing*. A. Wesley, 1993.

[6] W. Lorensen and H. Cline. Marching cubes: A high resolution 3D surfaces construction algorithm. *Computer Graphics*, 21(4):163 – 169, 1987.

[7] C. Lürig and T. Ertl. Hierarchical volume analysis and visualization based on morphological operators. In *IEEE Visualization'98*, pages 335 – 341, 1998.

[8] J. Parker. A system for fast erosion and dilation of bi-level images. *Science computer*, 5(3):187 – 198, 1990.

[9] J. Rodríguez, D. Ayala, and A. Aguilera. Representation and boundary extraction of a 3d digital image using the EVM model. Technical Report LSI-00-67-R, UPC-LSI, 2000.

[10] A. Rosenfeld, T. Kong, and A. Wu. Digital surfaces. *CVGIP: Graphical Models and Image Processing*, 53(4):305 – 312, 1991.

[11] S. Sethia and S. Manohar. Minkowski operators for voxel based sculpting. *Computers and Graphics*, 22(5):593 – 600, 1998.

[12] L. Thurfjell, E. Bengtsson, and B. Nordin. A boundary approach to fast neighborhood operations on three-dimensional binary data. *CVGIP: Graphical Models and Image Processing*, 57(1):13 – 19, 1995.

[13] J. Udupa and O. Odhner. Fast visualization, manipulation and analysis of binary volumetric objects. *IEEE Computer Graphics and Applications*, 4(1):53 – 62, 1991.

[14] R. van Boomgaard and R. van Balen. Methods for fast morphological image transforms using bitmapped binary images. *CVGIP: Graphical Models and Image Processing*, 54(3):252 – 258, 1992.

[15] L. van Vliet and B. Verwer. A contour processing method for fast binary neighborhood operations. *Pattern Rec. Letters*, 1(1):27 – 36, 1988.

[16] I. Young, R. Peverini, R. Verbeek, and P. van Otterloo. A new implementation for the minkowski operators. *Computer Graphics and Image Processing*, 17:189 – 210, 1981.

[17] K. J. Zuidervelt. *Visualization of multimodality medical volume data using object oriented methods*. PhD thesis, U. Utrecht. Deutch, 1995.