Discrete Shapes and Planes

Multiresolution Representation of Shapes in Binary Images II: Volume Images

Gunilla Borgefors[1], Gabriella Sanniti di Baja[2], and Stina Svensson[1]

[1] Centre for Image Analysis, Swedish University of Agricultural Sciences, Lägerhyddvägen 17, SE-752 37 Uppsala, SWEDEN
email:gunilla,stina@cb.uu.se
[2] Istituto di Cibernetica, Italian National Research Concil, Via Toiano 6, IT-80072 Arco Felice (Naples), ITALY
email:gsdb@imagm.na.cnr.it

Abstract. Multiresolution representations of discrete patterns are of great interest, specially when working with volume images. The huge amount of data that volume images contain at high resolution can be considerably compressed at lower resolution, and while the obtained representation still can be suited for simple shape analysis tasks, provided that shape is adequately preserved when resolution decreases. In this paper, we present new methods for building shape preserving binary resolution pyramids in three dimensions. The performance of the methods is quantitatively evaluated.

1 Introduction

In [1], methods for building shape preserving binary resolution pyramids in two dimensions are described. In this paper we present the corresponding methods in three dimensions.

Multiresolution representations of discrete patterns are of great interest for several reasons. Whatever image application, the best resolution can be used at each step, and the results can be propagated through the pyramid. Being able to use low resolution, i.e. small images, in parts of the computations becomes especially important for volume images, as they contain huge amounts of data.

A typical example where resolution pyramids are useful is the matching phase in object recognition. Typically, the discrete pattern (as well as any available prototype) is represented by a graph, whose number of nodes depends on resolution. Graph comparison can first be performed using all prototypes in the library, but only at low resolution levels, where a small number of nodes are involved. This will sort out the most promising matchings. Detailed comparison can then be performed with a reduced number of prototypes at higher resolution levels, where a larger number of nodes have to be taken into account.

In two dimensions the most straightforward example of a multiresolution shape representation is the 2×2 binary pyramid. The corresponding multiresolution image in three dimensions is the $2 \times 2 \times 2$ binary pyramid, where voxels are either black (pattern) or white (background). Each level of the pyramid is

a three dimensional array, where the size of the array is 1/8 of the size of the array at the immediately previous higher resolution level. A decimation process is used to build the successive resolution levels starting from the highest, original, resolution level. The next, lower, resolution level is built by partitioning the array in $2 \times 2 \times 2$ blocks of voxels, *children*, and associating a single voxel, *parent*, to each block. The new voxel is set to black or white depending on the colour of its children (and, in some cases, of their neighbours), according to some fixed rule. The process is repeated for the previously computed lower resolution representation, and then further iterated to build all possible resolution levels, ending in a single voxel in the lowest resolution level. If the original image is not $2^n \times 2^n \times 2^n$, planes are added as needed.

As it was the case in two dimensions, rules more sophisticated than the logical OR and AND operations should be used to build a shape preserving binary pyramid. In fact, all the shortcomings of these occurring in two dimensions are equally present in three dimensions, where shape modifications occur even more rapidly. If OR-pyramids are built, the set of black voxels soon becomes a large amorphous blob, due to the large number of $2 \times 2 \times 2$ configurations that contain at least one black voxel and thus get a black parent. Pattern regions sufficiently close to each other merge, filling in tunnels and cavities initially present in the pattern, or causing creation of new tunnels and cavities. If AND-pyramids are built, the pattern is soon dramatically shrunk. Narrow regions of the initial pattern either completely vanish or become disconnected as the resolution decreases. This is because only the $2 \times 2 \times 2$ configuration consisting of eight black voxels gets a black parent. In both cases, the shape of the pattern is not adequately preserved. Something in between the AND and OR pyramids is needed to produce better results as concerns shape preservation.

The position of the grid used to partition the array into $2 \times 2 \times 2$ blocks also affects the shape of the resulting pattern. The grid can be shifted in eight different positions over the image, originating eight differently shaped patterns at the next resolution level. A combination of the eight possible resulting images is expected to produce more stable representations. The general problem of placement of a lower resolution grid over a higher resolution becomes worse the higher the dimension of the image is, but is seldom addressed in the literature.

It should be remarked that changes of the digital topology of an object are unavoidable when image resolution is decreased. It is impossible to represent the connectivity, tunnels, and cavities of a three dimensional object unless there is a sufficient number of voxels. For this reason, we do *not* claim to preserve topology using the methods in this paper, but concentrate on perceived shape. In an attempt to quantify the goodness of our methods, we also introduce a voxel-counting measure. However, we do take topology into account when possible and it *is* fairly well preserved, especially as concerns maintenance of pattern connectedness.

2 Some Definitions

Let I_1 be the $2^n \times 2^n \times 2^n$ original black and white image, stored in the bottom level (the first level) of the pyramid. Each of the successive n levels of the pyramid (I_2,I_3,...,I_{n+1}) is built from its preceding level. Every voxel of level I_k, $2 \leq k \leq n-2$, is either black or white, depending on the colours of the voxels in I_{k-1}. Since the last three levels consist of only 1, 8, and 64 voxels, respectively, they are not meaningful for shape representation. We define, in the following, our resulting resolution pyramid as a structure consisting of $n-2$ levels, where the $8 \times 8 \times 8$ level is the last one.

All our methods will be illustrated on the same small ($2^6 \times 2^6 \times 2^6$) test image shown in Fig. 1. The image consists of a cylinder which ends in a cone and on the top of that a sphere with a tunnel, all in the digital Euclidean metric. This $2^6 \times 2^6 \times 2^6$ image is the first level of the pyramid, I_1. As this level is always the same, whatever scheme used to build the pyramid, we will in the following show only the three lower levels, consisting of $2^5 \times 2^5 \times 2^5$, $2^4 \times 2^4 \times 2^4$, and $2^3 \times 2^3 \times 2^3$ voxels, respectively. Using a larger image as an illustration is not necessary, as shape and topology preservation problems are more prominent at low resolutions. We have, however, tested the methods on many different images. A few examples will be discussed in Section 6.

Fig. 1. First level, I_1, of a binary pyramid, where a $2^6 \times 2^6 \times 2^6$ test image is stored.

3 Combining Eight Binary Images

The first approach to construct a shape preserving 3D pyramid is based on the combination of the eight binary images, which are obtained by shifting the partition grid to all possible positions. For a given position of the partition grid, an image of the same size as the successive level of the pyramid is built from its preceding level, using a simple rule. We might resort to the commonly used OR and AND operations to decide about voxel colour, or use mathematical operations counting the number of black and white children. The latter possibility is more flexible than simply using OR and AND operations and will be preferred in this paper. The eight resulting images are then combined, again using some arithmetic rule, to form the successive level of the pyramid. This new level is then partitioned in the eight possible ways, and the next level is built, as before. The process continues until level $8 \times 8 \times 8$.

The building rules we adopt compute the sum of the values of the eight children, $\sum v$. These rules also allow us to obtain the OR- and the AND-pyramid. In the OR-pyramid a parent becomes black if $\sum v \geq 1$ and in the AND-pyramid a parent becomes black $\sum v = 8$. Other arithmetic rules could be used, based on $\sum v$, i.e. $\sum v \geq n$, where $n = 2, ..., 7$. Although the shapes of the resulting patterns are better than those provided by the OR- and the AND-pyramid, shape and topology are still barely preserved if we use only a single grid position.

Formally, the pyramid is built in the following way. Let $A_{i+1,1}$, $A_{i+1,2}$, $A_{i+1,3}$, $A_{i+1,4}$, $A_{i+1,5}$, $A_{i+1,6}$, $A_{i+1,7}$ and $A_{i+1,8}$ be the eight different images, obtained by shifting the grid used to partition I_i into $2 \times 2 \times 2$ blocks. The rule $\sum v \geq n$, where $n = 1, ..., 8$, is used to build $A_{i+1,j}$, $j = 1, ..., 8$. When combining these eight images, the sum of the eight voxels values in the same position, $\sum g$, is computed for all voxels. The combined level I_{i+1} is obtained setting to black all voxels with $\sum g \geq m$, where $m = 1, ..., 8$.

Pyramid rule and image combinations where $\sum v \geq n$ and $\sum g \geq 9 - n$, $n = 1, ..., 8$, give satisfying results, see Section 6. In Fig. 2 we show the result when $n = 1$ is used, which is equivalent to first "OR-ing" and then "AND-ing."

Note that although these "combined" pyramids are much less sensitive to the placement of the $2 \times 2 \times 2$ grid, the results are still not translation independent. When the grid is shifted a new plane of white voxels must be added. Depending on in what direction the grid is shifted, the plane can be placed at the top or at the bottom, in front or behind, or, to the left or to the right of the image, which again gives eight possibilities.

4 Using Intermediate Grey-Level Images

Handling eight volume images simultaneously is per definition time and memory consuming. Therefore, we have devised a method where a combination of the images that would be obtained in correspondence with the different grid positions can be achieved without actually building the eight images. The combination is coded as an intermediate grey-level image, which contains the same information

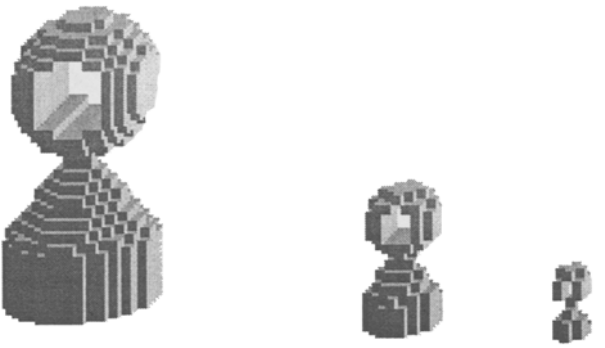

Fig. 2. Pyramid computed by AND-ing eight OR images

as the eight shifted binary images. By binarizing this grey-level image, according to some rule, the next binary pyramid level can be computed.

Suppose that v is a voxel in level i, having even coordinates, i.e. v belongs to row $2j$, column $2k$ and plane $2l$, for some j, k and l. Depending on the position of the grid on the i level image, the voxel v belongs to one of the eight $2 \times 2 \times 2$ blocks shown in Fig. 3. Each of these blocks would generate one voxel (either black or white) on each of the eight next level binary images. The grey-level image at resolution level $i + 1$, G_{i+1}, of the pyramid is built by taking into account the value that v would have in all eight images, by using the $3 \times 3 \times 3$ linear filter shown in Fig. 4. The weights in the filter correspond to the number of times a neighbouring voxel to v is present in the blocks in Fig. 3. The voxel itself is present in all, whereas the point-neighbours are present only in a single block. In G_{i+1} the possible values of a voxel v ranges from 0 (when v and all its neighbours are white) to 64 (when v and all its neighbours are black).

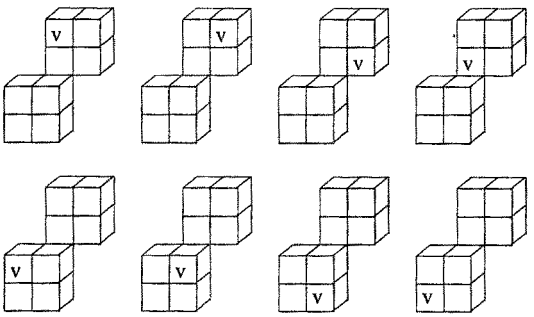

Fig. 3. Possible positions of the grid around the voxel v.

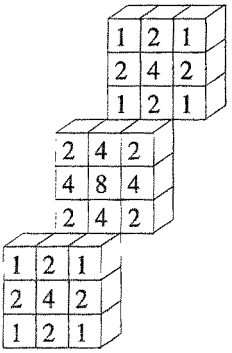

Fig. 4. Mask of weights for computing the intermediate grey-valued image.

The intermediate image G_{i+1} can be binarized by using different criteria. The simplest would be to threshold the image at a suitable level, but we achieved better results by more sophisticated rules. Note that, simple thresholding by setting all voxels with grey-level $f > 0$ ($f = 64$) to black does *not* produce the OR (AND) image.Generally, voxels with small values (close to 0) should become white, as they have few black neighbours and would probably be white in the next level of the binary pyramid in any grid position. Analogously, voxels with large values (close to 64) should be set to black, as they have mostly black neighbours and would probably be black in almost all eight images obtained by shifting the grid. For the voxels with intermediate values, the final colour is decided not only by the value of the voxel itself, but also by the values of its neighbours. This means that the values of a parent is based not only on the values of its children, but on the values in a $7 \times 7 \times 7$ neighbourhood.

The two best criteria for shape preservation that we have found are the following. Let a voxel v have value f.

- **Criterion 1:**
  If $0 \leq f \leq 20$, then set v to white.
  If $21 \leq f \leq 43$, then set v to black if it is locally maximal along either the horizontal, the vertical or the depth direction, otherwise set it to white.
  If $44 \leq f \leq 64$, then set v to black.

The rule for $21 \leq f \leq 43$ is intended to preserve pattern connectedness while avoiding unnecessary thickening. The resulting pyramid for the test image by applying *Criterion 1* to the grey-valued intermediate images is shown in Fig. 5.

- **Criterion 2:**
  If $0 \leq f \leq 20$, then set v to white.
  If $21 \leq f \leq 43$, then set v to black if f is larger than the average value of a $3 \times 3 \times 3$ neighbourhood centred on v, otherwise set it to white.
  If $44 \leq f \leq 64$, then set v to black.

The rule for $21 \leq f \leq 43$ is intended to preserve the most "significant" out of several neighbouring voxels. The resulting pyramid for the test image by applying *Criterion 2* to the grey-valued intermediate images is for our simple example *exactly* the same as for *Criterion 1*, so see Fig. 5 again, but this is not usually the case.

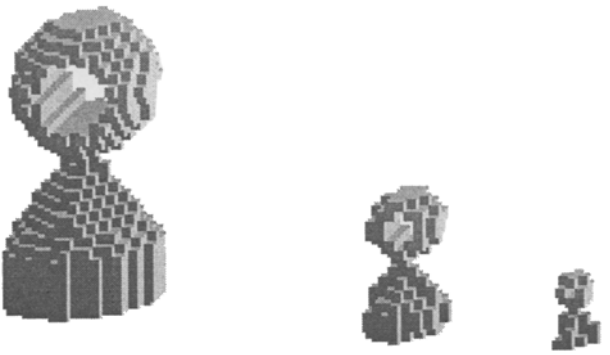

Fig. 5. Pyramid made by using intermediate grey-valued image levels using *Criterion 1*. The tunnel in the sphere is preserved in all three resolution levels.

5 Implementations

The implementations of the described algorithms are written in C and integrated in IMP (IMage Processing), a general image analysis software developed at *Centre for Image Analysis*. The software is using the X-Windows standard and Motif for the user interface. For the visualization of the images we also use IMP.

Even though we are working with 3D images, when run on a DEC Alpha (a standard UNIX workstation) it takes about one second to compute the binary pyramid for a $128 \times 128 \times 128$ image when using the method with intermediate grey-level images. The pyramid that combine eight binary images at each level is slower to build, but the times are still reasonable. It takes about 5 seconds for a $64 \times 64 \times 64$ image and about 40 seconds for a $128 \times 128 \times 128$ image.

6 Results

The results for our methods and the motivations for the thresholds at *Criterion 1* and *2* are presented in this section, using three different test images. The first one is the same as earlier, see Fig. 1. To build the second image we have

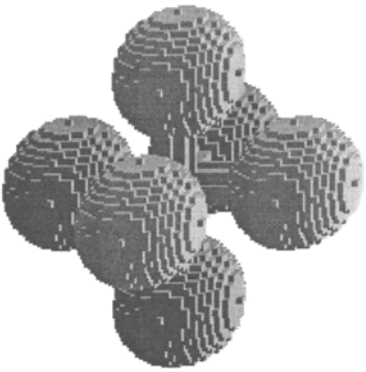

Fig. 6. Test image consisting of six hollow spheres connected by three cylinders

Fig. 7. Test image consisting of an HIV particle

connected six hollow spheres by three cylinders, see Fig. 6. The third is a real image consisting of an HIV particle, see Fig. 7.

To get one objective, quantitative measure of the performance of our methods, we have counted object voxels in the different pyramid levels for the different methods. Let $V(I_i)$ be equal to the number of object voxels in level I_i. We define

the measure $M(I_i)$ as

$$M(I_i) = \frac{8^{i-1} \times V(I_i)}{V(I_1)},$$

i.e. when the $M(I_i)$ is close to 1.00 the object in level I_i contains the "correct" number of object voxels. If it is > 1 there are "too many" and if < 1 "too few". This gives some sort of measure of how good the methods are even if it does not take shape into account. Since *Criterion 1* and *Criterion 2* give very similar results, we only present both methods for the first image. We have also looked at the topology of the results for a large number of test images. The conclusions support those for the presented examples.

- **Image one, the tower in Fig. 1** The original image contains 20035 voxels. In Table 1 the result is shown for the method of combining eight binary images, and in Table 2 the result when using the intermediate grey-level image. When using the method of combining eight binary images at each level according to the arithmetic rules $\sum v \geq n$ and $\sum g \geq 9 - n$, the connectivity is preserved for $n = 1, ..., 6$. The grey-level method works well for all the thresholds in the image, from the connectivity point of view.

- **Image two, the spheres in Fig. 6** The original image contains 23172 voxels. In Table 3 the result is given when using the intermediate grey-level image. A wide range of thresholds give similar results. In the $8 \times 8 \times 8$ image the cavities in the spheres are not preserved and when the lower threshold is low the six spheres turns into a blob-looking thing.

- **Image three, the HIV particle in Fig. 7** The original image contains 13396 voxels. See Table 4 for the result when using the intermediate grey-level image. The shape of the particle is well preserved in level I_2 and I_3. Because of the complex structure the object is hard to represent in an $8 \times 8 \times 8$ image.

Table 1. The result for image one when building the pyramid by using a combination of binary images.

$\sum v$	$\sum g$	$M(I_2)$	$M(I_3)$	$M(I_4)$
1	8	1.01	0.98	0.95
2	7	1.01	0.97	0.74
3	6	1.00	0.98	0.82
4	5	1.00	0.97	0.61
5	4	0.99	0.96	0.69
6	3	1.00	0.95	0.61
7	2	0.99	0.97	0.69
8	1	1.00	0.97	0.69

Table 2. The result for image one when building the pyramid by using intermediate grey-level images. The first three columns describe the result when using *Criterion 1* and the last three when using *Criterion 2*.

lower threshold	higher threshold	$M(I_2)$	$M(I_3)$	$M(I_4)$	$M(I_2)$	$M(I_3)$	$M(I_4)$
16	44	1.03	1.07	1.23	1.03	1.07	1.23
17	44	1.03	1.06	1.20	1.03	1.06	1.20
18	44	1.01	1.03	1.12	1.01	1.03	1.12
19	43	1.01	1.03	1.15	1.01	1.03	1.15
19	44	1.01	1.02	1.12	1.01	1.02	1.12
19	45	1.01	1.02	1.12	1.01	1.02	1.12
20	43	1.01	1.01	1.10	1.01	1.01	1.10
20	44	1.01	1.00	1.07	1.01	1.00	1.07
20	45	1.01	1.00	1.07	1.01	1.00	1.07
21	43	1.01	1.00	1.07	1.01	1.00	1.07
21	44	1.00	1.00	1.07	1.00	1.00	1.07
21	45	1.00	1.00	1.07	1.00	1.00	1.07
22	44	0.97	0.97	0.89	0.97	0.97	0.89

Table 3. The result for image two when building the pyramid by using intermediate grey-level images and binarize the images with *Criterion 1*.

lower threshold	higher threshold	$M(I_2)$	$M(I_3)$	$M(I_4)$
16	44	0.98	1.01	1.81
17	44	0.98	1.00	1.75
18	44	0.98	0.99	1.59
19	43	0.96	0.98	1.37
19	44	0.96	0.97	1.35
19	45	0.95	0.96	1.28
20	43	0.96	0.95	1.22
20	44	0.96	0.94	1.17
20	45	0.95	0.93	1.10
21	43	0.94	0.93	1.08
21	44	0.94	0.93	1.06
21	45	0.93	0.92	0.99
22	44	0.94	0.89	0.93

From this, we can conclude that values of the lower threshold 20 and the higher threshold 44 in the method that uses intermediate grey-level images are not very critical. A small change of threshold does not change the results significantly. Connectivity, tunnels, and cavities are preserved in all images except for some cases of the $8 \times 8 \times 8$ image.

When we use the method that combine eight binary images at each level, $\sum v \geq n$ and $\sum g \geq 9 - n$, where $n = 1, ..., 8$, the results are quite similar for all n, and all are significantly better than using just the OR or the AND operations. It should, however, be pointed out that the best method to preserve

Table 4. The result for image three when building the pyramid by using intermediate grey-level images and binarize the images with *Criterion 1.*

lower threshold	higher threshold	$M(I_2)$	$M(I_3)$	$M(I_4)$
16	44	1.00	1.34	2.18
17	44	1.00	1.31	1.87
18	44	0.99	1.26	1.57
19	43	0.99	1.19	1.41
19	44	0.98	1.19	1.41
19	45	0.97	1.19	1.41
20	43	0.96	1.11	1.07
20	44	0.95	1.11	1.07
20	45	0.94	1.11	1.07
21	43	0.95	1.07	1.07
21	44	0.94	1.07	1.07
21	45	0.93	1.07	1.07
22	44	0.92	1.01	0.96

thin links is to AND eight OR-images. This result differs from the 2D case, where intermediate values of v and g were found to be preferable.

7 Conclusions

Several new methods have been presented that improve shape preservation when representing volumetric patterns in binary resolution pyramids. Our first approach uses only binary images. A binary pyramid is built by using suitable combinations of the eight images obtained by shifting the $2 \times 2 \times 2$ partition grid. The rules are simple: first count the number of black children and build the eight possible lower resolution images, then count the number of times a voxel is black in any of the eight images. We have found that AND-ing eight OR-images generally gave the best results for the many patterns we have tested.

Our second approach replaces the eight binary images with one grey-level image, computed by a $3 \times 3 \times 3$ linear filter, which take into account the eight $2 \times 2 \times 2$ configurations that can be placed within the $3 \times 3 \times 3$ neighbourhood of a voxel. This grey-valued image is binarized by double thresholding and any of two proposed intermediate rules. These rules do not only consider the grey-level value of the voxel itself, but also those of its neighbours, thus using the context of a voxel to determine its significance in the pattern. This method is much faster than the previous one.

To determine the necessary thresholds in the grey-level methods, we have introduced a simple quantitative measure of the quality of the results. It is shown that a small change in threshold values does not change the resulting pyramid very much.

Our methods are easy to implement and produce much better results than the ones obtained by the "standard" OR/AND pyramids. They also preserve topol-

ogy reasonably well, but topology can not be accurately preserved in decreasing resolution no matter what you do. This is especially true in three dimensions.

Acknowledgements

The authors wish to thank Giuliana Ramella for the implementation in the 2D case. Also, thanks to Ingela Nyström, for great support with the implementation in the 3D case.

References

1. G. Borgefors, G. Ramella, and G. Sanniti di Baja. Multiresolution representation of shape in binary images. In S. Miguet, A. Montanvert, and S. Ubéda, editors, *Discrete Geometry for Computer Imagery (DGCI'96)*, pages 51–58. Springer Verlag, Berlin 1996. Lecture Notes in Computer Science 1176.