Multiresolution Segmentation of 3D Images by the Hyperstack


Computer Vision Research Group, University of Utrecht, The Netherlands

ABSTRACT

This paper describes recent advancements in the design of the hyperstack, a three-dimensional image segmentation tool. It is based on a multiresolution approach which is mathematically supported by the diffusion equation. The blurring strategy, used to build the scale space, is outlined, including some difficulties that occur in view of the transition from 2D to 3D. We extend the existing prototype, a hyperstack grounded on iso-intensity following, with two new ideas: weighted linking and stand-alone parents. The result of a segmented 3D SPECT image is shown. Theoretical considerations concerning the addition of feature information to guide the segmentation process are briefly mentioned. Finally, we outline a flexible way to obtain several output images from one single hyperstack, and investigate the reduction of the sampling rate by means of interpolation, which will decrease the total amount of processing time.

1. INTRODUCTION

In the past, various approaches to segment digital images have been extensively studied: thresholding, region growing and edge detection. Besides these conventional segmentation algorithms, more recent methods based on the multiresolution concept have proven to be very effective. The latter class comprises the pyramid [1], [2], [3], the stack [4], [5], [6] and the wavelet representation [7]. Multiresolution segmentation techniques have our special interest for two reasons:

1. Biological analog. The biological system, which is as yet unparalleled in distinguishing global objects and detail in a continuous scene, is a multiscale image processor [4].

2. Global information. The conventional segmentation methods fail to take global information into account [8]. Consequently, this makes them particularly unsuitable for processing images with a low signal to noise ratio, or images with a broad range of object sizes.

Furthermore, the multiresolution approach offers us the possibility to include a priori information about the image in the algorithm, which can be used to guide the segmentation process.

For the two-dimensional case a successful multiresolution method for image segmentation is available: the stack [4], [5]. Unfortunately, the transition from 2D to 3D is far from trivial [9]. In particular, two important topics concerning the extension to three dimensions deserve our attention. Firstly, it can be shown that the image boundaries have a strongly increasing influence on the blurring and the segmentation of the image when the number of dimensions is increased [6]. Secondly, the following of extrema – which is extensively studied by Lifshitz [5] – is thwarted by the existence of two types of saddles into which the extrema can annihilate. These so-called hypersaddles are found by examining the invariants of the Hessian matrix in 3D, resulting in four classes of Morse critical points [9].

In this paper we describe an improvement of the hyperstack as a segmentation tool. We extend the simple iso-intensity algorithm outlined in the prototype [9] with two new ideas: weighted linking and stand-alone parents. Both are described below, including some results and theoretical considerations concerning the addition of extra information by feature extraction. We also briefly discuss the
actual segmentation step and the reduction of the sampling rate with respect to computation time.

2. THE DATA STRUCTURE

Our first prototype of the three-dimensional stack was a straightforward extension of the 2D stack, based on Gaussian blurring and iso-intensity following. This approach differed – up to then – from the 3D pyramid in that the scale space was built by images of a constant size in each dimension direction, whereas the pyramid decreases these sizes with a factor of two at each successive level of lower resolution. Another major difference, owing to the lower rate at which Gaussian blurring decreases the resolution of an image, is the absence of the need to apply an iterative algorithm. Consequently, the linking process and the actual segmentation can be separated in the hyperstack.

![Fig. 1. Linkage structure of the hyperstack. The numbers represent voxel intensities.](image)

For the storage of the four-dimensional data structure we use a database of records, in which each record represents a voxel in scale space and its linkages to other voxels (see Fig. 1). One voxel is always related to another via one of the link types parent or child. For reasons of implementation – with an eye towards computation time – we link every child that is not the first one of a parent, by a so-called sibling link to the last linked child of that parent. Thus, the parent-, child- and sibling link types correspond to a voxel being linked to one in a lower, higher or equal level of resolution, respectively.

It would be useful to have an estimate of the total amount of voxels contained in a hyperstack, for example when calculating the amount of memory needed to store a hyperstack. If we denote this number by \( N \), then in a hyperstack consisting of \( l \geq 1 \) levels – \( N \) obeys the following inequality:

\[
  n_0 + (l-1) n_{l-1} \leq N \leq (l-1) n_0 + n_{l-1}, \tag{1}
\]

where \( n_i \) denotes the number of parents at resolution level \( i \) (all voxels at the ground level are included in the hyperstack, and non-parent voxels at higher levels are disregarded). This absolute range of the total amount of voxels needed, however, is far too rough when the aim is to deduce some practical value for \( N \).

In the case of a relatively large number of levels \( l \) (which is a reasonable assumption), \( N \) can be estimated much more closely by assuming that

\[
  n_{i+1} \approx c \cdot n_i, \tag{2}
\]

with \( c \) a constant between zero and one. Eq. (2) has been found to be a satisfactory approximation for all values of \( i \), except for \( i = 0 \), owing to the presence of unpredictable high frequency components in the original image. For all images studied (CT, SPECT), \( c \) ranged between 0.72 and 0.76. Hence, the estimate for \( N \) is rather insensitive to the actual value of \( c \), and can be approximated well by

\[
  N \approx n_0 + 4 \cdot n_1, \tag{3}
\]

which yields an accuracy of more than 95% in all cases considered. Note that this approximation depends on \( n_0 \) and \( n_1 \) only. Within some limits, we are also able to estimate the total number of levels \( l \) in which the hyperstack will be converged to one single top parent \((l \approx 1 - \log n_1, \text{ with } n_{l-1} = 1)\).

From Fig. 1 it follows directly that the total number of parent linkages (arrows up) must
equal the sum of the number of child- and sibling linkages:

\[ N_p = N_c + N_s \]  

(4)

Furthermore, \( n_{l-1} \) will in general be very small with respect to \( N \). Thus, the correlation between \( N \) and \( N_p \) is given by:

\[ N_p = N - n_{l-1} \approx N. \]  

(5)

This result, together with eq. (3) and (4), provides us with a practical estimate for the amount of memory needed.

The selected type of data structure has two important characteristics:

1. **Fast hyperstack scanning.** Ascending the four-dimensional tree structure is equally fast as descending it. Furthermore, the different voxels at one level of resolution belonging to one and the same (part of a) segment can easily be identified. Thus, it should be possible to develop an efficient tree scanning algorithm, which is indeed the case [9]. By scanning the hyperstack we mean descending the entire tree from the top-most level, while using a priority decision rule to determine which type of link to follow next. Similar to the idea of backtracking – when starting from the top level – the order in which the references are followed is child, sibling and parent linkages.

2. **Including feature information.** Feature following through scale space can at low cost be included in the data structure. This follows from the fact that we use sibling linkages rather than connecting each child to its parent by a separate duplex link. All the sibling linkages that can be reached from the one (and only!) child reference of a parent, connect those voxels that logically belong together. That is, they form one part of a segment at that level of resolution. All these grouped voxels obviously have a strong connection with the feature information extracted at the same level in scale space. On the other hand, the segments present in one level below are to be joined in some way or another at the current level. As will be explained later in this paper, we are able to assign a mathematical value to each segment, which indicates the probability that it will merge with another segment. Problems may arise at segments with a low probability value. These segments are indicative of the locations where additional feature information might be helpful.

### 3. Blurring

Based on the idea that the biological visual system deals with imagery data in a multiresolution-like manner, it has been shown that – at least in the continuous case – a Gaussian point spread function should be used to build the scale space [4]. Hence, we use a normalized discretized Gaussian kernel in order to make iso-intensity following possible.

We denote the transitional standard deviation of level \( i \) to level \( i+1 \) by \( \sigma_{i,i+1} \) and the effective standard deviation of a Gaussian – that is the \( \sigma \) that belongs to a one-step convolution of the original image to create level \( i \) – by \( \sigma_i \). The relation between the transitional and the effective standard deviation is:

\[ \sigma_i^2 = \sum_{k=0}^{i-1} \sigma_{k,k+1}^2, \quad 1 \leq i \leq l-1. \]  

(6)

(Notice that \( \sigma_0 = \sigma_{0,1} \)) The value of \( \sigma_i \) plays an important part in determining the size of the blurring kernel. More precisely, we take \( r_i = k \cdot \sigma_i \) as the radius of the kernel, with \( k \) a measure for the accuracy of the convolution.

**Table 1.** Accuracy values for several kernel radii.

<table>
<thead>
<tr>
<th>( k )</th>
<th>( 1D )</th>
<th>( 2D )</th>
<th>( 3D )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>38.3</td>
<td>11.8</td>
<td>3.09</td>
</tr>
<tr>
<td>1.0</td>
<td>68.3</td>
<td>39.4</td>
<td>19.9</td>
</tr>
<tr>
<td>1.5</td>
<td>86.6</td>
<td>67.5</td>
<td>47.8</td>
</tr>
<tr>
<td>2.0</td>
<td>95.5</td>
<td>86.5</td>
<td>73.9</td>
</tr>
<tr>
<td>2.5</td>
<td>98.8</td>
<td>95.6</td>
<td>90.0</td>
</tr>
<tr>
<td>3.0</td>
<td>99.7</td>
<td>98.9</td>
<td>97.1</td>
</tr>
</tbody>
</table>

Since computation time grows exponentially
for increasing \( r \), we pose a limit on \( k \). Nevertheless, the higher the number of dimensions, the lower the accuracy of the convolution will be for any specific value of \( k \). This follows directly from Table 1, containing rounded percentages of the definite integral of a Gaussian point spread function. For the three-dimensional case, using spherical coordinates, this yields a volume \( V_k \):

\[
V_k = \frac{1}{(\sigma \sqrt{2\pi})^3} \int_0^{2\pi} \int_0^\pi \int_0^1 r^2 \sin \theta \, dr \, d\theta \, d\phi \, e^{-\frac{r^2}{2\sigma^2}},
\]

which equals

\[
\text{erf} \left( \frac{k}{\sqrt{2}} \right) = \sqrt{\frac{2}{\pi}} \cdot k \cdot e^{-\frac{k^2}{2}}.
\]

Alternatively, we can blur in the Fourier or Hartley domain. Then, however, we can not handle the boundaries of the image as flexibly as in the spatial domain. This is undesirable for three-dimensional images, because the influence of the boundaries is quite significant, especially for relatively small images [6], [9].

4. THE LINKAGE SCHEME

In order to improve iso-intensity following, two mechanisms are added. Firstly, the strength of linkages between children and parents is quantified. The introduction of this quantification makes the algorithm more general and eases the addition of criteria other than intensity proximity. Secondly, if certain criteria are not met, we do not link a child to its best parent, but make it a stand-alone parent. These so-called roots are isolated parts of the hyperstack tree that define one single segment each. We use a part of the linkage strength component in the criteria for root formation.

4.1 Linkage strength

The linkage strength between a child and its parent is expressed by a number between zero and one. In the present design, two linkage strength criteria are used: (1) the proximity in intensity of the child and the parent, and (2) the size of the segment the parent will represent. Note that the latter forces us to use a small iterative process before the final linkages can be established. This follows from the fact that the segment size of a parent is defined as the total number of voxels in the ground level linked to that parent, after the linkages for the current level have been made. The linkage strength (denoted by \( F_l \)) is defined by:

\[
F_l^2 = \frac{w_{\text{int}} \cdot F_{\text{int}}^2 + w_{\text{seg}} \cdot F_{\text{seg}}^2}{w_{\text{int}} + w_{\text{seg}}},
\]

with

\[
F_{\text{int}} = 1 - \frac{|I_p - I_c|}{I_{\text{max}}}
\]

and

\[
F_{\text{seg}} = \frac{S_p}{S_{\text{max}}},
\]

Obviously, the influence of the factors \( F_{\text{int}} \) and \( F_{\text{seg}} \) depends on the ratio of their weight factors \( w_{\text{int}} \) and \( w_{\text{seg}} \). The \( I_p \) and \( I_c \) refer to the intensity of the parent and the child, respectively, with \( I_{\text{max}} \) as maximum difference between these two. The segment size of the parent is denoted by \( S_p \), with \( S_{\text{max}} \) as the maximal segment size during the last iteration. In practice, about three iterations produce a stable situation with respect to the linkages.

The intensity strength component corresponds in a linear manner with all possible values between the minimum and maximum intensity difference. The segment size component causes a preference for large segments over small ones. This is desirable for two reasons. The first is that it guarantees convergence for regions with a uniform or almost uniform intensity. The second reason is rather pragmatic: preference for large segments speeds up the convergence of the hyperstack tree, because voxels at the edge of an object will sooner make the decision to append to one of the large segments at either side of the edge. (In fact, voxels near edges are seduced by the
4.2 Root creation

The necessity of the creation of roots can be demonstrated by the following. Since repeated blurring of the original image causes small objects to be wiped out, it follows that small objects should be isolated before annihilation occurs. For this purpose, criteria are needed to decide whether or not a root should be created (see Fig. 2).

![Linkage structure containing roots](image)

Up to now, two criteria have been found useful with respect to root creation:

1. Small segment growth. If the segment size of a child would — after linking with its most suitable parent — grow less than a certain percentage, the creation of a root is considered.

2. Large intensity difference. If criterion 1 is met and the intensity difference between the child and its most suitable parent exceeds a certain upper bound, a root is created. For the intensity difference we use \( F_{mi} \) as defined by eq. (9b).

Other criteria, such as the segment size factor \( F_{seg} \) or the spatial distance between a child and its parent, have also been studied, but were found less useful than criteria 1 and 2. (Experiments have shown that 0.97 is a reasonable limit for the normalized intensity proximity.)

For reasons of consistency and implementation, all remaining parents in the top-most level are transformed into roots and appended to the existing list of roots.

5. RESULTS

The hyperstack as described above — based on iso-intensity following, weighted linking and root creation — has been applied to segment a three-dimensional SPECT image (see Fig. 3). This image represents a liver and an enlarged spleen of size \( 64 \times 64 \times 20 \) voxels, of which only the slices 11 through 18 are shown.

Note the distinction between the small dark spots in the liver. Also, the almost unrecognizable vertebral column in the original appears much sharper in the segmented image.

6. FUTURE WORK

In the future we plan to investigate the addition of geometrical information so as to guide the linking- and segmentation process. We are also interested in adapting the hyperstack in order to let one single hyperstack produce more than just one (segmented) image. Finally, we discuss the consequences of applying an interpolation technique to reduce the sampling rate.

6.1 Additional feature information

We divide the feature extraction into two areas of interest: extrema following and edge information. Other (higher order) features may also be worth investigating.

The use of extremal regions as an image description method is based on the stack approach of Koenderink [4]. In short, extremal points at all different levels of resolution are followed until they annihilate into a saddle. In this way, extrema following provides us with a hierarchical description of an image, specifying containment relations between extremal regions. (An extremal
Fig. 3. SPECT image of liver (small object) and spleen (large object) of a child suffering from severe liver parenchymal disease. (a) Original 3D image. (b) Segmented image.
region is an iso-intensity contour in the original image at the same intensity as the extremum it encloses.)

With respect to the 2D stack, Lifshitz showed that – at least theoretically – a nonextremum can escape from the extremal region it originates in [5]. Consequently, it is possible for an extremum to miss the annihilation into the correct saddle. It is to be expected that the existence of two different types of saddles – besides the other two non-degenerate critical points, i.e. maxima and minima – into which extrema should annihilate, will only worsen the problem of escaping nonextremum paths.

Besides extrema following we are currently investigating the addition of edge information. From our point of view, we only use edges as extra information, not as a basis for our segmentation. Therefore, the edge extraction technique should obey the following conditions: (1) the accuracy is adjustable, (2) only real edges are to be found, and (3) it is applicable at any level of resolution.

In conformity with these three requirements, an orientation dependent template matching method, capable of finding edges, has our special interest.

6.2 The segmentation step

The result of a segmentation is an image divided into several separated segments, each with one monotonic intensity. Whenever an application, such as visualization by volume rendering, strongly depends on the quality of the segmentation, then the entire process of building a hyperstack should be traversed each time the results should be adapted. This would result in an unacceptable response time, at least with the current prospective view on hardware. We propose two improvements on this:

1. Multiple trees. By using more than one tree, instead of a single hyperstack, one avoids that all feature information should be combined during the building of the hyperstack. Different segmentation results can be derived by only adapting the algorithm that combines the iso-intensity tree and other feature trees, without the necessity to build a hyperstack completely from the ground level.

2. Fuzzy segmentation. Rather than connecting each voxel in the segmented image uniquely to a segment, we may – according to some probability distribution – allocate it to several segments. Fuzzy segmentation is directly applicable to visualization techniques. Once the desired number of segments has been defined, the appropriate threshold value can be calculated from the probabilities that correspond with the voxels. In an optimum situation this can be done interactively, again without building a new hyperstack for each resulting segmentation.

6.3 Sampling rate reduction

Theoretically, the need to keep the size of the image constant in the resolution dimension is absent. In essence, the inter-pixel distance at each level should be proportional to the standard deviation of the total amount of (effective) blurring [8]. Although the algorithms will become more complex when the sampling rate is reduced after every blurring-and-linking phase, we expect the total computation time to decrease significantly.

Note that the contribution of every interpolating step (say, \(\sigma_p\)) to the total amount of blurring is determined by the deviation of the Fourier transform of the interpolating kernel from the inside of an ideal low-pass filter (which would result in no blurring effect at all) [10]. With this knowledge, the amount of stack blurring becomes \(\sigma_{i,j+1} - \sigma_{i,j}\). We hereby assume that blurring precedes interpolating, which seems – at least theoretically – the only correct order.

Thus, three possibilities deserve further study:

a. Spatial blurring, followed by (spatial) interpolation. The computation time of this sequence depends mainly on the kernel size used in the spatial blurring.

b. Spatial blurring & interpolation in one step. The major disadvantage of combining both in one algorithm is the necessity to recompute the weight coefficients of the kernel for each voxel: there is no guaranty anymore that the
sample points of the kernel coincide with those of the image.

c. Blurring in the frequency domain, followed by spatial interpolation. There is a crossover point where it becomes cheaper to perform the blurring in the frequency domain, but the disadvantage concerning the handling of the boundaries (as stated before) remains valid.

REFERENCES


