Multi-spectral Quadtree based Image Segmentation

Ben G.H. Gorte

International Institute for Aerospace Survey and Earth Sciences (ITC),
Enschede, the Netherlands.
e-mail: ben@itc.nl

KEYWORDS: Remote sensing, algorithms, multi-spectral, image segmentation, quadtrees

ABSTRACT

The paper gives a description of an image segmentation method, which is based on multi-spectral image data. It is embedded in a quadtree based GIS and Image Processing system. Generally, the system gives the possibility to integrate remotely sensed data, map data and attribute data. It offers raster processing capabilities, combined with high resolutions, with modest storage and processing time requirements. By subdividing an image into segments, assuming that these correspond to objects in the terrain, the integration of R.S. and GIS can be strengthened.

1 Introduction

For the purpose of widening for post-graduate and M.Sc. students in our institute the opportunities to study and investigate spatial data structures, a modest quadtree software system was gradually developed during the last few years. In the course of this activity, a stage was reached were quadtree based image segmentation could be implemented without too much additional effort. Although this subject has received attention in literature since the seventies [3], it did not become widely accepted in the field of analysis of remotely sensed imagery. For example, in very respected textbooks in this field, such as [4] and [7], image segmentation is not mentioned. Also the major commercial digital image processing software packages do not include image segmentation modules. Nevertheless, image segmentation is intuitively appealing. Human image vision generally tends to divide the image into homogeneous areas first, and characterize those areas more carefully later. Applying this approach to digital image analysis software leads to a segmentation step, which divides the image into segments that correspond — in the ideal case — to meaningful objects in the terrain, followed by a supervised classification step, in which each segment is compared with class characteristics that are derived from training data. In contrast to usual classification methods, the comparison does not have to be limited spectral properties, but can also take spatial characteristics of segments (size, shape and adjacency to other segments) into account.

The remainder of this paper focuses on quadtree based segmentation. The success of segmentation depends on the availability of:

- High resolution imagery, such that the relevant objects are represented by a significant number of pixels; otherwise there is no point in segmentation.
- Powerful hardware: fast and with a lot of memory
- An efficient implementation, regarding the sizes of remote sensing images.

A special case, which is typical for earth observation applications, is multi-band imagery. Grey-scale segmentations ([5], [1]) of the individual bands give different sets of segments. In this paper a method is presented that segments a multi-spectral image into one unique set of objects.

2 Quadtrees

Quadtrees serve as a spatial data model, in the sense that they allow the storage of data about various types of spatial objects and phenomena, as well as the operations on such data. In this study, area based quadtrees are used [6], which are conceptually equivalent to raster maps. Therefore, raster based GIS analysis operations are also defined in the quadtree domain and a large number of them can be implemented efficiently [2]. The advantage is that the specifications and implementations of many GIS operations, such as mapcalculations and other kinds of overlay, are straightforward in the raster (and quadtree) domain.

On the other hand, the raster data structure tends to lead to large data volumes, which need a lot of space and processing time. In case of (however advanced) "ordinary" compression techniques, the space requirements are relieved, but the processing times increase, because the actual processing will still take place pixel by pixel, and expansion / compression steps must be added.

The quadtree data structure and software help to decrease the storage and processing time requirements at the same time, especially at high resolutions. Roughly, storage requirements increase linearly with resolution when using quadtrees, and quadratically using rasters. The challenge of quadtrees is to create algorithms that work in the quadtree domain, which means that they do not expand

the data to raster format at any stage. In that case, processing times depend on the quadtree data set sizes, which leads to a significant gain at high resolutions.

Another advantage of the raster data structure is the ease of integration of map and image data. This advantage is equally valid for the area based quadtree approach. Unfortunately, not much is gained in terms of space and time, when processing images as quadtrees. However, quadtrees allow to combine data layers with different resolutions without having to re-sample one to the other.

2.1 Data Structure

Linear, sequential quadtrees (no indexing)

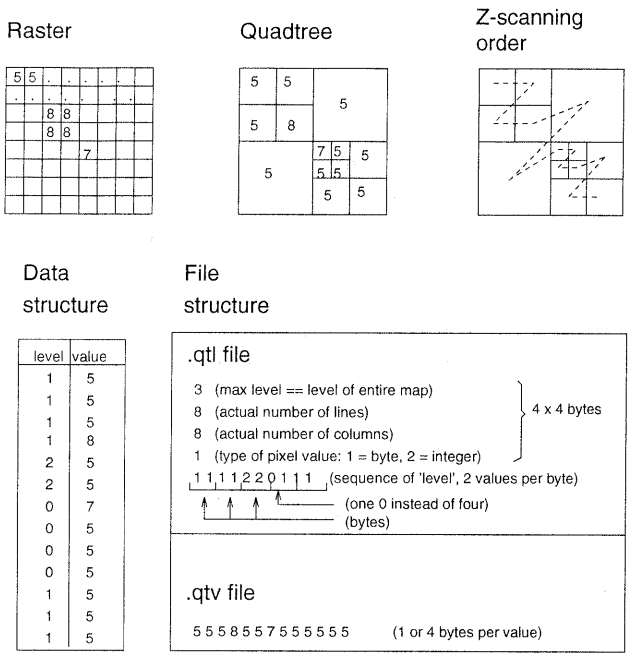

Figure 1: Quadtree data structure

2.2 Operations

The package in its current state is useful to demonstrate quadtrees in education or to explore them during research in the field of spatial data structures and operations. The programs have in common that they read input and write output sequentially and simultaneously, without excessive buffering in internal memory. Therefore, there are no (practical) limitations to the sizes (resolutions) of the data sets to be processed. The entire data set does not have to reside in internal memory at any time.

The following modules are present:

General: raster to quadtree and quadtree to raster conversions, image calculations, statistical analysis (histograms, multi-band statistics), simple map and image generalization, which allows to create levels of detail (LOD) at different representation scales.

Image Analysis: Training data analysis and maximum likelihood classification, principal component transformation, RGB to IHS transforms.

GIS: Map overlay and map calculation, aggregation functions, determination of topology (region adjacency), connected component labeling.

The segmentation algorithm is based on the one for connected component labeling — in fact, the latter will appear to be a special case of the former. Also, the map calculation module will be involved in the segmentation process, as well as the connected component labeling. Therefore, we describe these three modules with somewhat more detail.

Connected component labeling: a program that assigns to each homogeneous region a unique value. The output quadtree values have the type "integer", which allows over 2,000,000,000 regions - more than there will ever be pixels in the input. It's interesting to notice that the structure of the quadtree does not change with this operation.

The program assumes 4-adjacency: a pixel has only four neighbors (above, below, left and right), which are taken into account when connectivity is established, instead of 8 neighbors (the diagonal ones don't count) In the "very high resolution quadtree filosophy", region pairs that are 8-adjacent without being also 4-adjacent are very unlikely to occur,

Image and map calculations are carried out by a program which allows overlaying data layers by performing arithmetical, mathematical, logical and relational operations on corresponding pixels in different layers.

This program also provides the link between spatial and attribute data. If pixel values have the meaning of *object number*, attribute values can be found at any pixel by indexing the attribute table with the pixel value. See the result of segmentation in Figure 2.

Region Adjacency software can be used to establish adjacency between pixel values in a quadtree. The result is a relational table with two columns; if somewhere in the quadtree a pixel with value p is neighboring a pixel with value q, then (p, q) will be a record in the table. The table is sorted in ascending order of (primarily) the first column and (secondarily) the second column. The value in the second columns is always larger than the one in the first; if p is less then q, you will find a record (q, p) in the table. Therefore, every combination is listed only once.

The operation makes most sense if the quadtree is filled with regions that have unique numbers, such as the result of an image segmentation process. In that case it generates region adjacency information, which can be incorporated in subsequent classification.

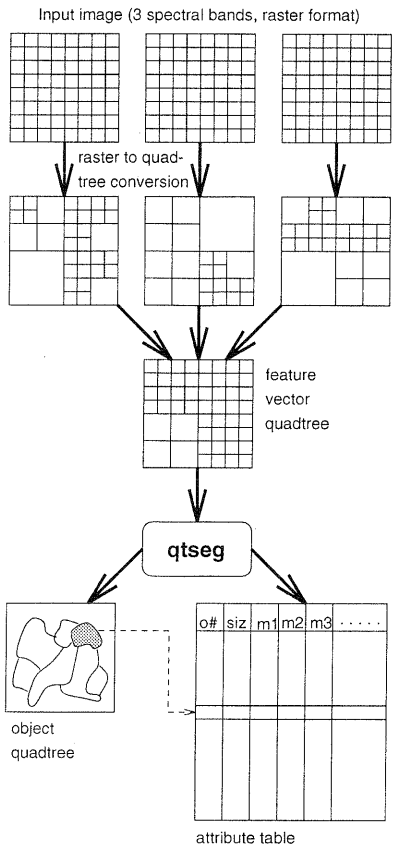

Figure 2: Data structures for quadtree segmentation of multi-spectral images

3 Image Segmentation

The implementation presented here uses a multi-band image as input and gives one segmentation as output: One set of objects, where each object has multi-spectral properties (mean vector and variance-covariance matrix). Moreover, topological (object adjacency) information can be retrieved as well as, of course, object locations, sizes and perimeters.

The presented method performs segmentation by recursively combining (merging) pairs of pixels, leafs and regions. It uses data from multiple (currently two or three) input bands, that are combined into a single *feature vector quadtree* first. Currently, the criterion for merging is very simple: With a user-selected threshold T, the Euclidean distance between the feature vectors of two candidates may be not larger than $2 \times T$ and none of the variances and covariances after merging may exceed T^2. Note that the above mentioned connected component labelling is a special case: only one band and $T = 0$.

Like in the other programs in the package, the quadtree is scanned sequentially, which implies a single traversal through the image in Z-scan order (Figure 1). Therefore, the process is recursive and works bottom-up. It starts trying to combine individual pixels (within quadrants) first, and looks at possibilities to combine objects in larger quadrants later.

The program relies on a highly dynamic data structure consisting of an **index table** and an **object table**. The object table has one record for each (intermediate) object, in which the object size and spectral attributes are stored. In case of three spectral bands, these attributes are: the sums of the pixel values in band 1, 2 and 3 over the entire object (S_1, S_2, S_3), the sum-of-squares (S_{11}, S_{22}, S_{33}) and the sums of the cross-products (S_{12}, S_{13}, S_{23}). These are used in the calculations of the mean values and the covariance matrix for the object.

An object is entered in the table when a "new" leaf from the input is read. A new entry in the index table points to the object. When processing a quadrant, the values to either side of the boundaries between the sub-quadrants are taken from a **stack**. Via the index table, the spectral data are retrieved from the object table and used in the merge criterion.

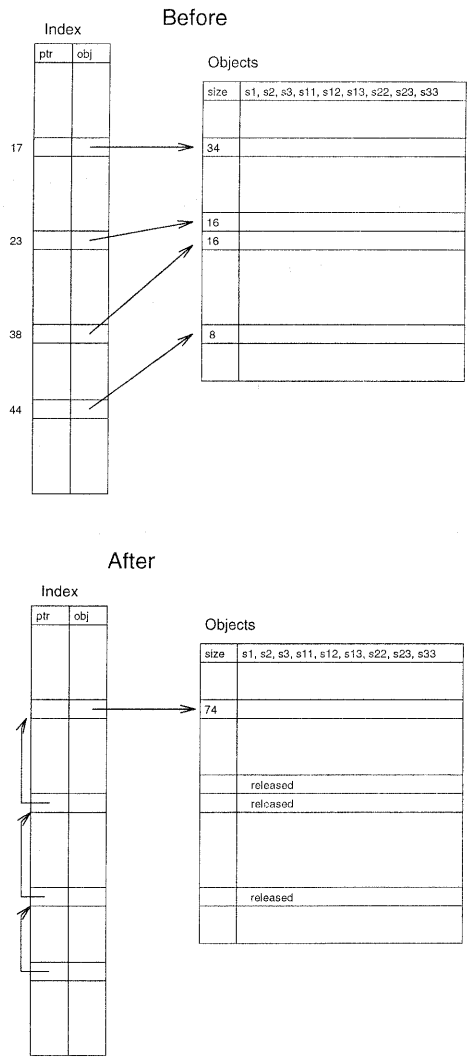

Figure 3: Index and Object table before and after processing fig. 4

If two objects can be merged, their respective attribute values (sizes and sums) are added and stored in the table entry of the object with the lowest object number. The other object is removed from the table. Also the index table is updated: the higher entry will point to the lower one. Figure 3 shows the states of the index and object table before and after processing the quadrant in Figure 4.

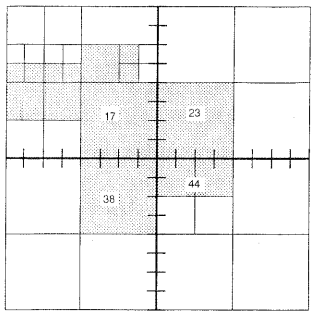

Figure 4: Segmentation at intermediate level

After the quadrant is finished, also the (new) values at the outer boundaries are known. They are stored at the next higher level of the stack, from where they will be retrieved when the next larger quadrant (containing this one) gets processed.

When the entire quadtree is processed in this way, which is when the program reaches the highest level, the index table is updated: All entries that have an object number associated with them are moved to the beginning of the table; the pointers of all other entries are updated so that they will point to the end of the chains. Then the input quadtree is read again and the output (segmented) quadtree is produced. Finally, an attribute table is created from the object table, by transforming sums and sizes into means and covariances. The attribute table is stored on disk and can be used in subsequent analysis.

3.1 Iteration

Due to the recursive z-scan order, the process has a slight tendency to create segments of regular shapes, according to the quadrants. This effect could be completely removed by making the process perform a few iterations, with increasing threshold values. With one threshold value, the process only merges, and because it works quadrant by quadrant, it first attempts to merge within quadrants. When starting with a lower threshold value than the final one, the risk of inadvertedly merging sub-quadrants reduces. Irregular shapes will already be formed, however, and will be the basis for further merging later, when higher threshold values come into effect.

3.2 Small objects

The application of the image segmentation process to satellite images causes a large amount of small segments (say, less that five pixels in size) to be created. One reason may be, of course, that due to the limited resolution of satellite imagery, there are many of such small objects in the terrain.

More important, however, is the effect of *mixed pixels*, especially at the boundaries of objects with quite different spectral signatures. In the feature space, those mixed pixels are too far away from both objects, and therefore they cannot be merged with one of them. The question is what to do with them. From a segmentation point of view, we would like them to be incorporated into larger (neighboring) segments. To achieve this, we can relax the merging criterion, by increasing the threshold value especially for small segments. However, the spectral values of the boundary pixels will "contaminate" those of the entire segment (unless we don't update the values of the larger segment when merger is due to criterion relaxation — this was not investigated, however) and influence a later classification. Another possibility is to leave the small segments (mixed pixels) out of the classification procedure and classify only the large ones. The above-described map calculation program can be used to make the selection of large segments, based on the sizes in the attribute table. Under the assumption that objects are relatively large, compared to the pixel size, there is a slight preference for the second option.

4 Experiment

The segmentation process was applied to a Landsat TM image of the Flevopolder in the Netherlands. The "advantage" of this area is that there are large fields, so segmentation really makes sense. usually, Landsat TM does not satisfy the previously stated condition that objects should consist of a significant number of pixels. The method will be more useful when higher resolution imagery becomes available.

The results are shown in Figures 5 and 7. Using map calculation, combining the segment quadtree with the attribute table, only large segments were selected and a random grey value was assigned to them. Small segments were removed.

The image consists of 1000 × 1000 pixels. With a final threshold value of 6, 180811 segments were created. Despite the large objects in the terrain, many segments are very small: 136870 single pixels and respectively 20053, 5252, 4009 and 2539 segments of two, three, four and five pixels. Figure 5 shows small segments in black and reveals that they are mostly boundary (mixed) pixels.

On the other side of the scale, there are four segments with more than ten thousand pixels. They are water bodies (IJsselmeer and Randmeren), with 11149, 33317, 44069 and 111375 pixels, respectively. The distribution of the sizes of the more moderate objects is shown in Figure 6

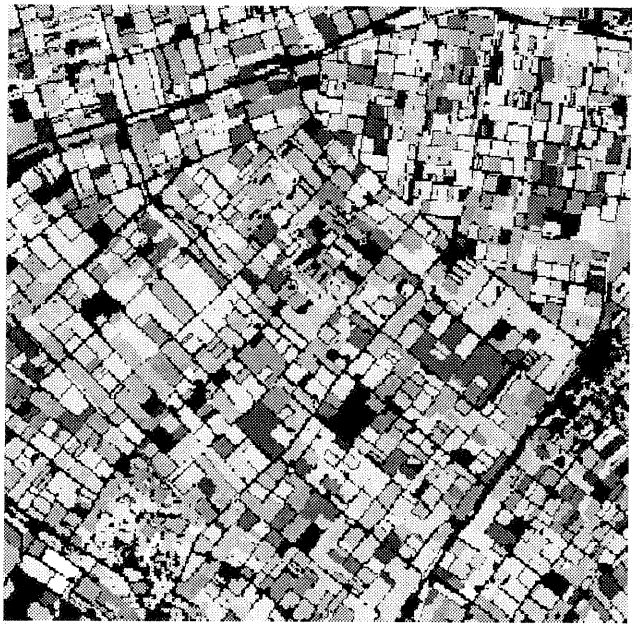

Figure 5: Detail of segmented image. Objects are displayed with random grey values, those that are smaller than five pixels are black

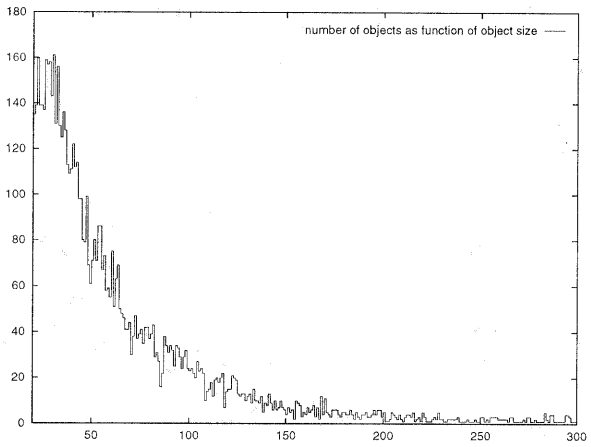

Figure 6: Distribution of object sizes

5 Conclusions

The paper gives a description of an image segmentation method, which is embedded in a quadtree based GIS and Image Processing system. Generally, the system gives the possibility to integrate remote sensing data, map data and attribute data. It offers raster processing capabilities, combined with high resolutions of maps and images, without excessive storage and processing requirements. By subdividing an image into segments, assuming that these correspond to objects in the terrain, the integration of R.S. and GIS can be strengthened. However, the correspondence must probably be further established using classification procedures.

A new aspect in the segmentation process is that it is based on multi-spectral image data. The difference between the proposed method and the existing grey-scale segmentation methods is that in the second case different results are obtained from the individual bands, which must be later combined by overlaying. This comparable to the difference between parallelepiped (box) classification and minimum (Euclidean or Mahalanobis) distance classifier – the latter are usually superior.

For the time being, the merging criterion is a simple one, based on mean spectral values and covariance matrices. In the introduction of this paper, we said that the human vision system tends to segment the image first and to classify the segments later. Probably, for trained image interpreters, it is more realistic to assume that they do both at the same time. In the near future, we will therefore incorporate training statistics in the criterion: subsegments will be merged if the resulting spectral signature still fits to one of the sample distributions.

References

[1] Y.-L. Chang and X. Li, "Fast image region growing", *Image and Vision Computing*, Vol. 13, pp.559-571, 1995.

[2] B.G.H. Gorte, *Experimental quadtree software*, Technical report, ITC, Enschede, 1995.

[3] S.L. Horovwitz and T. Pavlidis "Picture segmentation by a tree traversal algorithm", *J.ACM*, Vol.23, pp.368-388, 1976.

[4] P.M.Mather, *Computer processing of remotely-sensed images, an introduction*, Wiley, 1987.

[5] O.J. Morris, M. deJ. Lee, A.G. Constantinides, "Graph theory for image analysis: an approach based on the shortest spanning tree", *IEE Proc.-Part F.* Vol. 133, pp.146-152, 1986.

[6] H. Samet, *The design and analysis of spatial data structures*, Addison-Wesley, 1990.

[7] J.A. Richards, *Remote Sensing Digital Image Processing, an introduction*, Springer-Verlag, 1993.

Figure 7: Segmented image (1000 * 1000 pixels)