REPRESENTING AND RETRIEVING REGIONS USING BINARY PARTITION TREES

L. Garrido, P. Salembier and J.R. Casas

Signal Theory and Communications Department, Universitat Politècnica de Catalunya
C/ Gran Capità s/n, 08034 Barcelona, Spain, Tel: (+34) 934 011 627, Fax: (+34) 934 016 447
E-Mail: {oster, philippe, josep}@gps.tsc.upc.es, URL: http://gps-tsc.upc.es/imatge

ABSTRACT

This paper discusses the interest of Binary Partition Trees for image and region representation in the context of indexing and similarity based retrieval. Binary Partition Trees concentrate in a compact and structured way the set of regions that compose an image. Since the tree is able to represent images in a multiresolution way, only simple descriptors need to be attached to the nodes. Moreover, this representation is used for similarity based region retrieval.

1. INTRODUCTION

There is currently a strong interest in defining content descriptors for information retrieval. This interest is motivated by the every day increasing number of audio visual documents. Automatic or semiautomatic algorithms are therefore needed to index this huge amount of information. The development of the MPEG7 [1] is an example of such a need.

Most of the research work has been done on the extraction of descriptors from an image [2]. However, in many cases the analysis of the image is done at a certain resolution scale. That is, the image is usually first partitioned, and descriptors are then extracted from this partition. Imposing a certain resolution scale may therefore lead to a loss of some regions that compose the images. A representation of images that allows the analysis at different scales of resolutions is therefore still a topic to be further developed.

In this paper, the Binary Partition Tree is used to represent and retrieve regions included in an image. The Binary Partition Tree is a structured and compact representation of regions that can be extracted from an image. It was originally proposed to process and filter images [3]. However, in this work the multiresolution capabilities of the Binary Partition Tree are discussed. From a MPEG7 point of view, the Binary Partition Tree can be seen as a signal-oriented description scheme.

An example of a Binary Partition Tree is shown in Figure 1. The tree leaves represent the regions of an initial partition. The remaining nodes represent the regions that are obtained by merging the two children regions. In this representation, the root node represents the entire image support. Note that the tree represents a set of regions at different scales of resolution.

The organization of the paper is as follows. In section 2 the creation of the Binary Partition Tree and the attachment of descriptors to its nodes is discussed. In section 3 the Binary Partition Tree

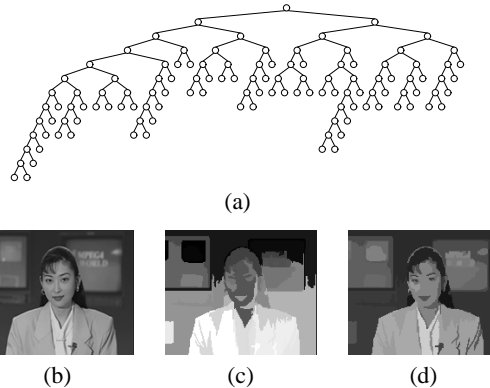---

(a)

(b)   (c)   (d)

Figure 1: Example of Binary Partition Tree. a) Binary partition tree of Akiyo, b) Original image, c) Initial partition with 60 regions, d) Image where each region of the initial partition has been filled with the mean color value of the original image.

is used to retrieve regions based on a query-by-example strategy. Finally, section 4 is devoted to the conclusions.

2. REGION REPRESENTATION

2.1. Creation of the Binary Partition Tree

In Figure 2 an example showing the creation of the Binary Partition Tree is illustrated. Several approaches can be used to create this tree. We have used a segmentation that follows a bottom-up approach. The algorithm first constructs the Region Adjacency Graph of an initial partition (Figure 2b). The initial partition can be the partition of flat zones or any other precomputed partition. Using a region based segmentation such as [4, 5, 6], the Binary Partition Tree is created by keeping track of the regions that are merged at each iteration until one region is obtained. That is, for each pair of neighboring regions a homogeneity measure is assessed. The algorithm then starts merging the pair of neighbors whose distance is lowest. Suppose, for instance, that the first regions to merge are regions 1 and 5 (see Figure 2).This is indicated in the Binary Partition Tree with a node labeled 6 whose children are regions 1 and 5. Regions 1 and 5 then are merged on the RAG, and the distances between nodes are recomputed. Regions 2 and 3 are then merged, resulting in a region labeled 7. This process is iterated until one region is obtained.

The Binary Partition Tree should be created in such a way that the most meaningful regions are represented in its nodes. The ho-
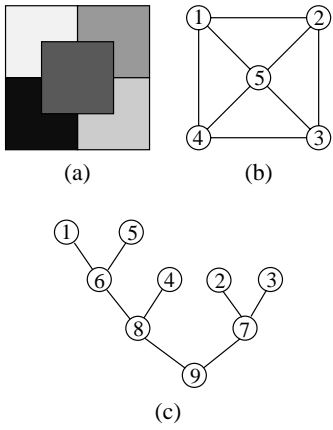
Figure 2: Example of Binary Partition Tree creation. a) Initial Partition, b) Region Adjacency Graph of (a), and c) Resulting Binary Partition Tree.

mogeneity criterion used in the example of Figure 1 is based on color similarity. It should be noticed, however, that the homogeneity criterion is not always restricted to the latter one. For example, if the image for which we create the Binary Partition Tree belongs to a sequence of images, motion information can also be used to generate the tree: in a first stage, regions are merged using a color homogeneity criterion, whereas a motion homogeneity criterion is used to merge regions in the second stage [7, 8]. Furthermore, additional high-level information of previous processing or detection algorithms can also be used to generate the tree in a more robust way. For instance, a mask of an object included in the image can be used to force that the object itself is represented with a node in the tree. Typical examples of such algorithms are face, skin or character recognition.

For similarity based retrieval purposes, region descriptors should be attached to the nodes of the Binary Partition Tree. Since the Binary Partition Tree allows to represent a region in a multiresolution way, only simple descriptors have to be attached to each node since a region can be represented using the descriptors associated to its node, or it can be represented using the descriptors of the descending nodes. Two types of descriptors are attached to the representation: color descriptors, which deal with the inner structure of the region, and geometry descriptors, which deal with parameters such as position, size, rotation and shape of the region.

2.2. Example of region descriptors

The simplest region color descriptor is a constant value, for example the mean YUV value of the associated region in the original image. Although this color descriptor attached to each node may seem insufficient to represent its texture or even its visual appearance in a correct way, it is in fact good enough for our purposes since we have to take into account that the Binary Partition Tree gives us information of how each region is composed. This allows to analyze a region at different scales of resolution. A region can be browsed using only the color descriptor attached to the associated node. This could give us a poor approximation of the region we are dealing with. However, by using the descriptors of the descendant leaf nodes, the region can be reconstructed with much more colors and detail. Note also that there is a correlation between the color model for a node and its children. In most cases it is possible to compute the color model for a node by knowing the

color model associated to its children. For instance, if the model used to represent the color is the mean value of each color component, the model for a certain node can be computed simply by averaging the models of its children. This property can be used for coding the tree in an efficient way.

Geometry descriptors involve position, size, rotation, and shape features. The goal is to obtain a set of shape descriptors that are invariant to the region position, size and rotation. Descriptors of position, size and rotation can be extracted using moment analysis [9]. This technique analyzes the statistical properties of a population of vectors. Consider the population of vectors $\mathbf{x} = [x_1, x_2]^T$, where T stands for vector transposition, and x_1, x_2 are the coordinates of the pixels that belong to a region R. The mean vector is defined as $\mathbf{m}_x = E\{\mathbf{x}\}$, and the covariance matrix of the population of vectors as $\mathbf{C}_x = E\{(\mathbf{x} - \mathbf{m}_x)(\mathbf{x} - \mathbf{m}_x)^T\}$ (whose size is 2×2). Since $\mathbf{C}_x$ is real and symmetric, finding the two orthonormal eigenvalues is always possible. Let $\mathbf{e}_i$ and λ_i, $i = 1, 2$ be the eigenvectors and corresponding eigenvalues of $\mathbf{C}_x$. The next step is to construct the matrix $\mathbf{A}$, whose rows are formed by the eigenvectors of $\mathbf{C}_x$, ordered so that the first row of $\mathbf{A}$ is the eigenvector corresponding to the largest eigenvalue. $\mathbf{A}$ is a transformation matrix that maps $\mathbf{x}$ into vectors denoted by $\mathbf{y}$ by the transformation $\mathbf{y} = \mathbf{A}(\mathbf{x} - \mathbf{m}_x)$. The population of vectors $\mathbf{y}$ corresponds to the rotation and position invariant representation of the region R. Within this framework, $\mathbf{m}_x$ corresponds to the position descriptor, whereas $\mathbf{A}$ is a rotation matrix. It is therefore easy to extract the angle α that generates the matrix. Size invariance is accomplished by normalizing $\mathbf{y} = [y_1, y_2]^T$ by $\sqrt{\lambda_1 + \lambda_2}$, which is equivalent to normalize by the mean squared energy of the population of vectors of $\mathbf{y}$.

Since we are dealing with descriptors extracted from a set of region moments, the set of moments associated to a region obtained by merging two children regions $R = R_1 \cup R_2$ can be obtained by combining appropriately the set of moments of R_1 and R_2. Therefore, we only need to compute the moments for the leaves of the Binary Partition Tree, whereas the moments of the rest of nodes can be computed in a recursive way. Furthermore, using this property of recursivity, efficient compression techniques for image description may be devised.

The external boundary of the region can be coded using techniques like chain code, spline approximations, wavelet [10] or Fourier descriptors [11]. In our work we have used Fourier descriptors to code the contour. The external contour of region R is sampled at equally spaced samples in a clock-wise manner. The position, rotation and size transformation is then applied to the contour. Only some low frequency Fourier descriptors are used to represent the resulting contour. These shape descriptors are invariant to size, rotation and position.

In Figure 3 an example of region representation using the Binary Partition Tree is shown. Note that since lossy Fourier descriptors are used, holes may appear in the reconstructed representation.

3. REGION RETRIEVAL

In this section, our purpose is to show how the Binary Partition Tree representation can be used for region retrieval. That is, we want to find the set of regions of a database that best matches a query in color information and/or shape. The Binary Partition Tree is well suited for this purpose, since it allows to do the query at different levels of resolution.

(a)   (b)

Figure 3: Example of region representation, (a) Original region (b) Reconstruction using the descriptors attached to the leaves of the associated Binary Partition Tree.

3.1. Supported queries

Several types of queries are supported by the Binary Partition Tree. They are listed below:

- Query type I: The simplest way to search for the candidate regions that matches the query is to compare the descriptors of the nodes in the database with those associated to the whole region of support of the query. The query is based only on the descriptors corresponding to the whole region, without taking into account the internal color information and structure of the region. Fast searching can be done if the set of nodes of the Binary Partition Trees that belong to the database are indexed in an efficient way. However, this is out of the scope of this paper. The present technique has the advantage of being fast but its results may not be very reliable since it depends on the discriminating capability of the descriptor of one node.

- Query type II: Another type of query supported by the tree is based on measuring the *visual similarity* between regions. Two regions, candidate and query, have the same visual similarity if their contents in color are the same. Instead of using the descriptors of one node, descriptors attached to a set of descendant nodes of a candidate node are used to perform the matching. Note that the maximum possible resolution is reached when the descriptors attached to the leaves of a subtree are used. This type of search is computationally much more expensive than the first one. Therefore an intensive search over all nodes using this matching is not appropriate.

- Query type III: The third type of query supported by the tree deals with the *structural similarity* between regions. In this case the algorithm should search for regions that match the query having the same sub-components but without taking into account possible differences in color. This search technique is similar to the previous one. However, color and geometry descriptors are used in a different way to perform the matching.

3.2. Query strategy

The user has to provide the search engine with the example of the region (mask and pixels values) to search for. The search through the database is performed by first constructing the Binary Partition Tree of the query. The root node of the Binary Partition Tree of the query represents the region support. The algorithm should then look for the set of regions in the database whose content is similar to that of the query.

If the purpose is to have a rough idea of some possible candidates, the first type of search technique can be used (Query type I). The algorithm only has to look for the nodes in the database whose descriptors are similar to those of the root node of the query region tree.

Reliability is obtained by using a search technique based on internal color regions or structure (Query type II and III, respectively). Since performing this search technique through the whole database is computationally expensive and slow, the search is divided into two stages. In the first stage, a preselection of the nodes is computed using the first search technique (Query type I). This first search selects some candidates in the database that may match the query. In the second stage, each of these candidates is further analyzed and matched with the query with more reliability. In this case, the descriptors attached to the nodes of the subtree of the candidate regions are used. Both visual (Query type II) and structural (Query type III) similarity are measured, and the user is able to select which weight should be given to each of them.

In this second stage, the matching between the query and the candidate is not done, as expected, using the tree structure. Instead, the matching is performed using a pixel-based representation of the region that is extracted from the tree. This choice has been made because the structure of the Binary Partition Tree is not a reliable information. This is due to the fact that all the merging steps necessary to go from an initial partition to a partition with one region are represented within the Binary Partition Tree. Therefore, some merging steps are more reliable than others. For instance, three regions that are very similar in gray level may merge in any order depending on the similarity measure used to merge them and the level of noise included in each region. Different tree structures can therefore be obtained, even if the final resulting region is the same.

The second reason for not using the Binary Partition Tree structure is related to the MPEG7 approach. MPEG7 does not want to standardize the indexing algorithm nor the search engine. During the search process, it is very unlikely to exactly know how the trees stored in the database have been created. In a large number of cases, the Binary Partition Tree of the query and the candidate may be created using different criteria.

3.3. Distances between regions

For Query type I, as stated before, the search is performed directly on the Binary Partition Tree. The algorithm only searches for the set of nodes in the database whose shape descriptors best match the ones of the root node of the Binary Partition Tree associated to the query. Similarity is measured simply by computing, for instance, the mean squared error between them. Other information, such as position, size, orientation and over-whole color can also be used if required.

In order to perform a Query of type II or III, a normalized reconstruction of the candidate and the query over an image must be performed, since we will not use the tree structure for this purpose. That is, the leaves of the subtree associated to the candidate node and the leaves of the query are restored on an image with respect to a predetermined size, rotation and position of the over-whole region. These parameters are fixed a priori so that the reconstructed regions fit well in an image of about 150×150 pixels. For both candidate and query, we obtain the associated partition and YUV color image. The reconstructed query (resp. candidate) color image is denoted with Q (resp. C) and its associated partition with Q_p (resp. C_p).

The corresponding partitions are defined as $Q_p = \cup_i Q_i$ with $i = 1, \ldots N_q$ and $C_p = \cup_j C_j$ with $j = 1, \ldots N_c$. N_q and N_c are, respectively, the number of regions of the partition of the query and the candidate.

The color values YUV of the pixels q and c inside each reconstructed image are defined by $Q(q) = (Y_q, U_q, V_q)$, $q \in Q$ and $C(c) = (Y_c, U_c, V_c)$, $c \in C$.

3.3.1. Visual similarity

An overall measure of "visual similarity" is the root mean squared error (RMSE) computed between the color components of the pixels of the query and the candidate regions over the overlapping area. Different weights on the color components may be selected depending on the search strategy.

$$V_{dist} = RMSE\left(Q(x), C(x)\right), \ x \in C \cap Q \qquad (1)$$

3.3.2. Structural similarity

The measure of structural similarity is somewhat more complex. It should allow to search for regions having similar subcomponents but different colors. For that purpose, two measures are assessed. First, the regions Q_i of the query are filled with a given (constant) color model in order to get the optimal reconstruction of the candidate. The RMSE (E_C) between the candidate and the latter reconstruction is then measured. This gives us an idea of how well the partition of the query "explains" the colors of the candidate. Inversely, the regions C_i are filled with a given (constant) color model in order to get the optimal reconstruction of the query. The RMSE (E_Q) between this reconstruction and the query gives us an idea of how well the partition of the candidate "explains" the colors of the query partition.

The maximum of both errors in taken as the measure of structural distance

$$S_{dist} = max\left(E_C, E_Q\right)$$

The worst case has to be taken into account since, for instance, if the query partition is finer than the candidate partition, the associated error $E_C = 0$, whereas $E_Q \neq 0$.

3.3.3. Match parameter

The validity of V_{dist} and S_{dist} is restricted to the overlapping area of Q and C. This is taken into account by means of a confidence parameter k defined as follows

$$k = \frac{2 \cdot area\left(C \cap Q\right)}{area\left(C\right) + area\left(Q\right)}$$

3.3.4. Matching distance

For visual similarity (resp. structural similarity), a match parameter V_{match} (resp. S_{match}) is defined from V_{dist} (resp. S_{dist}) normalized to fit into the range $[0, 1]$ and then weighted by the confidence measure k defined above.

$$V_{match} = k \cdot \left(1 - \frac{V_{dist}}{V_{norm}}\right) \ \text{and} \ S_{match} = k \cdot \left(1 - \frac{S_{dist}}{S_{norm}}\right)$$

The final value used to rank the candidates for the best math is computed as

$$match = \alpha \cdot S_{match} + (1 - \alpha) \cdot V_{match}$$

where $\alpha \in [0, 1]$ sets the balance between visual similarity ($\alpha \rightarrow 0$) and structural similarity ($\alpha \rightarrow 1$).

3.4. Examples

In order to test the proposed algorithm, a database of Binary Partition Trees has been constructed. Several MPEG4 test images have been processed. Some of them contain human shapes, like the images belonging to the Akiyo, Claire, Weather and News sequences. Randomly selected images are also introduced in order to enlarge the database. In the sequel, the query is a human body (Akiyo in Figure 3b).

The Query type I selects some candidates nodes among all the nodes of the Binary Partition Trees included in the database. The selection is based only on Fourier descriptor similarity. No restriction to position, orientation nor color is done.

The first 40 results of a Query type I are introduced into the second stage. Figure 4 shows us the results for a visual similarity query. The first 6 results correspond to Akiyo, and the next results are regions belonging to the shoulder of Akiyo.

An example of Query type III is shown in Figure 5. As in the previous case, the 40 first candidates of the Query type I are taken in order to be analyzed in the second stage. As can be seen, the proposed similarity criterion orders the regions according to their internal structure, without taking into account their visually different contents. As a result, regions from Claire and News, which are similar to Akiyo in structure, have been retrieved.

4. CONCLUSIONS

In this paper, we have discussed the interest of Binary Partition Trees to represent regions or images at different scales of resolution. It allows to represent in a compact and structured way the different regions an image is made of. Taking into account the multiresolution capabilities of the Binary Partition Tree, only simple descriptors have to be attached to each node. Moreover, recursive computable descriptors allow devising efficient compression techniques. A particular application dealing with region retrieval has also be presented.

5. REFERENCES

[1] MPEG. MPEG-7: Requirements document. Technical Report ISO/IEC JTC1/SC29/WG11/w2461, MPEG, Atlantic City (NJ), USA, October 1998.

[2] Y. Rui, T.S. Huang, and S.-F. Chang. Image retrieval: Past, Present and Future. *Journal of Visual Communication and Image Representation*, 1998.

[3] P. Salembier and L. Garrido. Binary Partition Tree as an efficient representation for filtering, segmentation and information retrieval. In *International Conference on Image Processing (ICIP)*, Chicago (IL), USA, October 1998.

[4] J. Crespo, R.W. Shafer, J. Serra, C. Gratin, and F. Meyer. A flat zone approach: a general low-level region merging segmentation method. *EURASIP Signal Processing*, 62(1):37–60, October 1997.

[5] L. Garrido, P. Salembier, and D. García. Extensive operators in partition lattices for image sequence analysis. *EURASIP Signal Processing*, 66(2):157–180, April 1998.

[6] B. Marcotegui. Segmentation algorithm by multicriteria region merging. In *Mathematical Morphology and Its Applications to Image Processing*, pages 313–320, Atlanta (GA), USA, May 1996. Kluwer Academic Publishers.

Figure 4: Example of Query type II processing. Ordered from left to right and from top to bottom, the first nine results of the query are shown.

[7] P.E. Eren, Y. Altunbasak, and A.M. Tekalp. Region-based affine motion segmentation using color information. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, pages 3005–3008, Munich, Germany, April 1997.

[8] L. Garrido and P. Salembier. Region based analysis of video sequences with a general merging algoritm. In *European Signal Processing Conference (EUSIPCO)*, pages 1693–1696, Rhodes, Greece, September 1998.

[9] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Addison-Wesley, 1992.

[10] G.C.H. Chuang and C.C. Kuo. Wavelet descriptor of planar curves: theory and applications. *IEEE Transactions on Image Processing*, 5(1):56–70, January 1996.

[11] P.J. van Otterloo. *A Contour-Oriented Approach to Shape Analysis*. Prentice-Hall, 1991.

Figure 5: Example of Query type III processing. Ordered from left to right and from top to bottom, the first nine results of the query are shown.