IADIS International Journal on Computer Science and Information Systems
Vol. 2, No. 1, pp. 96-110
ISSN: 1646-3692

A PLATFORM FOR REGION SPACE ANALYSIS IN BINARY PARTITION TREES

Huihai Lu, John C. Woods, and Mohammed Ghanbari *Department of Electronic Systems Engineering, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, UK*

ABSTRACT

In this paper, we present a novel region-based image analysis platform. With a human in the loop, it provides an efficient method for object extraction, image segmentation, and filtering. The region space is organised into a Binary Partition Tree representation where small homogenous regions are successively merged. The hindsight of the merging process is therefore stored within the tree structure and the evolutions from the seed to root nodes are studied. By analysing these evolutions, discontinuities caused by the merging of heterogeneous regions are identified and the corresponding tree nodes short listed. These nodes indicate salient details within the image and from them tree simplification and region based filtering becomes possible. Given a simpler version of the original tree, manually assisted object extraction and image segmentation can be easily achieved and the experimental results show that our method facilitates the extraction of semantic content from the image.

KEYWORDS

Object extraction, image segmentation, image filtering, binary partition tree

1. INTRODUCTION

Given the explosion of multimedia applications, the ability to segment objects from digital images has a huge potential market impact. Many applications, such as medical image analysis, non-destructive defect inspection, food sorting, visual effects, and content-free image indexing and retrieval, are built on the assumption that these decompositions can be achieved and manipulation can take place at the object level. However the ability to extract semantic objects has confounded the research community for decades and is a fundamental barrier in image cognition.

Segmentation of an unknown image into a collection of semantic objects is, with exception of synthetic images, an ill posed problem. One of the main obstacles is to bridge the gap

between image regions segmented based on low level image properties and semantically meaningful objects. In this paper, we present an efficient platform in which hierarchical image regions are correlated with physical world objects with limited user intervention.

Region description is well provided under MPEG 7 (Avaro, O., 2001, Sikora, T., 2001), and recognises the importance of a region based approach. In literature (Gonzalez, R., 2002, Tsaig, Y., 2002), the procedure to generate a region based description of images is to take a set of over-segmented initial small regions (Vincent, L., 1991, Collins, R., 2003) and merge them progressively until a stopping criterion is met. This process is conveniently represented using a Binary Partition Tree (BPT) (Salembier, P., 2000).

The BPT represents an image in a compact and multi-scale representation. If it is browsed, one or more branches can be found whose combination, when reconstructed in the pixel domain, represent the object of interest. Therefore, object segmentation in the BPT domain becomes a problem of locating the prevalent branches. Normally a tree consists of many thousands of nodes, making manual retrieval from it labour intensive. Furthermore, the large number of nodes and limited display space make the visualisation of a tree cluttered, with the large proportion occluding one another causing many branches to be inaccessible. A tree analysis framework is proposed which provides a means to simplify the BPTs by pruning redundant tree branches for ease of human-computer interaction and also results in an image filtering tool which has applications for image archiving, compression and browsing. It is developed based on the findings of a set of subjective tests (Lu, H., et al, 2006) where 15 average users were asked to locate pre-defined semantic objects from trees. The test results showed that discontinuities in the evolutions of region statistics from the leaf nodes to the root highlight salient image details. The proposed framework explicitly models region evolutions based colour information and uses a modified second derivative function to detect discontinuities and measure their strengths.

In (Salembier, P., 2000), a pruning strategy followed by a marker propagation technique is proposed. This was provided as an image filtering tool and also can be viewed as a tree simplification technique since the resultant tree contains fewer nodes than the original. Our work deviates from this in the following fundamental ways. First, in (Salembier, P., 2000), tree branches are pruned according to a threshold criterion such as size, one node at a time. As an alternative what we do is to filter the tree by analysing the bottom-up evolution of region statistics from the leaf nodes to the root. No threshold is directly imposed on the tree nodes themselves which gives the user greater flexibility to control the pruning process. Second, a tree simplified by (Salembier, P., 2000) gives rise to many trunk branches which have had their sub-branches and leaf nodes removed. Therefore, the resultant trees may no longer be binary trees since some parents will not have two child nodes. They do not therefore lend themselves to further binary tree processing techniques. Our approach simplifies the tree whilst still complying with the binary tree definition.

The organisation of this paper is as follows. Section 2 describes the processing pipeline used to generate BPTs. In Section 3, the framework for analysing BPTs is presented. The use of the framework for tree simplification and image filtering is shown in Section 4 and experimental results are given in Section 5. The human-computer interaction platform is introduced in Section 6. Finally, conclusions are drawn in Section 7.

2. BPT CREATION

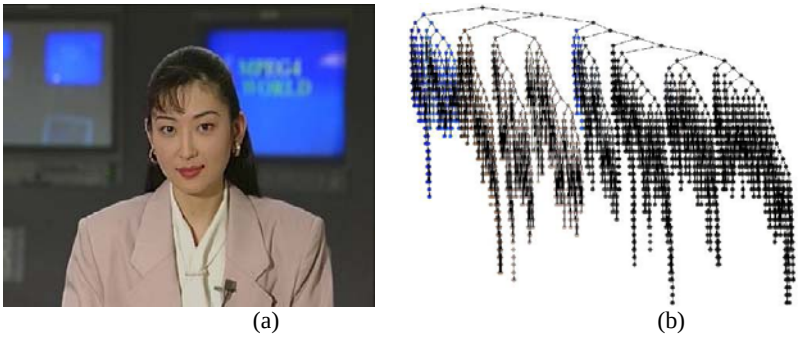
BPTs are generated based on a region merging process which is uniquely specified by a region model and merging order (Salembier, P., 2000). Let R_i denote the *i*-th region in the tree. The region model, $M(R_i)$ is defined as the average colour of the region in *CIE L*a*b** colour space. The merging order for a pair of adjacent regions R_i and R_j is defined as

$$O(R_i, R_j) = N(R_i)\left|M(R_i) - M(R_i \cup R_j)\right|_2 + N(R_j)\left|M(R_j) - M(R_i \cup R_j)\right|_2 \quad (1)$$

where $N(R_i)$ is the number of pixels in R_i.

The BPT generation pipeline is generalised as follows (Lu, H. et al, 2006). A colour image is first processed using the Watershed transform (Vincent, L., 1991). The result is an over-segmented graph containing many small and often semantically meaningless regions. These regions are then uniquely labelled. The pre-segmentation is followed by the computation of a Region Adjacency Graph (RAG). The initial RAG is subjected to a merging process which computes similarities between all possible pairs of neighbouring regions according to (1) and merges the most similar one into a single region. The RAG map is updated accordingly. The merging process continues until a single region covering the entire image is obtained. The merging history is logged using a BPT. If visualisation is required, the tree is parsed and drawn by a dynamic tree drawing algorithm (Moen, S., 1990). The overall process is threshold-free and does not require user intervention.

An example of a BPT generated from the Akiyo sequence is shown in Figure 1. Akiyo is regarded by the research community as an easy test sequence for foreground/background separation and object-based coding like MPEG4 (Battista, S. et al, 2000). However the tree in Figure 1(b) shows the density of regions generated by the immersion process and how complex the browsing of the tree can be. In general real-world images result in a large number of initial regions but the boundaries and semantic details are preserved.



(a) (b)

Figure 1. An example BPT. (a) Akiyo image. (b) Its corresponding BPT.

In (Bennstorm, C. and Casas, J., 2005), Bennstorm proposed a BPT creation technique which used a combination of three different homogeneity criterions including colour, size, and boundary condition to determine the merging order. Motion can also be used when generating

a tree. For example, in (Dorea, C. et al, 2005), a region merging strategy based on motion similarities across multiple frames was proposed. It allows the system to make merging decisions at each time instant based on global characteristics of the entire sequence.

In (Salembier, P., 2000), the BPT has been used for a number of different image processing applications such as visual browsing, object segmentation, and image filtering. For fast visual browsing, the authors employed a rate-distortion optimisation technique to find a partition of the image contained in the tree which minimises the distortion whilst the rate is under a given budget. For image segmentation, two methods were proposed. The first one is called the direct approach which progressively deactivates the arcs between parents and child nodes following the merging sequence until the stop criterion is met (the required number of regions, for example). The second method is a marker propagation approach. Markers are first manually placed in the tree and then propagated upward until there is a conflict between sibling nodes such as they are assigned to different markers. For the image filtering, each tree node is examined individually and those who do not satisfy a pre-defined criterion are removed from the tree. In (Salerno, O. et al, 2004), Salerno proposed an object recognition scheme based on shape matching. The BPT was used as a reduced problem space in which a set of tree nodes are found whose combination matches a given reference contour.

In general, there are four tree-based techniques used in the literature: 1) the direct use of merging order; 2) examining each tree node individually; 3) the sole use of tree structure; and 4) examining the local relations between sibling nodes. The full hindsight of the merging process stored in the tree has yet to be explored. In Section 3, we propose a generic analysis framework which combines both local image properties and the overall tree structure.

2.1 Pseudo tree levels

Binary trees are a standardised structure used in database applications. However its use is limited in the image processing literature and the terminology is lacking. We introduce the term *pseudo tree level* which is a collection of tree nodes L that must satisfy the following two conditions.

1) Any node in L is not an offspring or parent of another node in L.

2) The summed size of all the regions in L is equal to the size of the image.

The pseudo tree level provides a flexible way to define levels in hierarchical image representations. Each pseudo level is itself a segmentation of the image. By selecting different pseudo levels it is possible to declare a range of segmentations at varying levels of detail. Furthermore, within an individual pseudo level it is also possible to have considerably different region sizes and levels of detail.

Figure 2 shows two example pseudo tree levels. The tree and original image are shown in Figure 2(a). Two pseudo levels are manually selected and indicated by a dotted and a dash dotted line in the tree. Their pixel domain equivalents are shown in Figures 2(b) and (c), respectively. Given a BPT, it is clear that enumeration of all possible pseudo tree levels is a NP-hard problem.

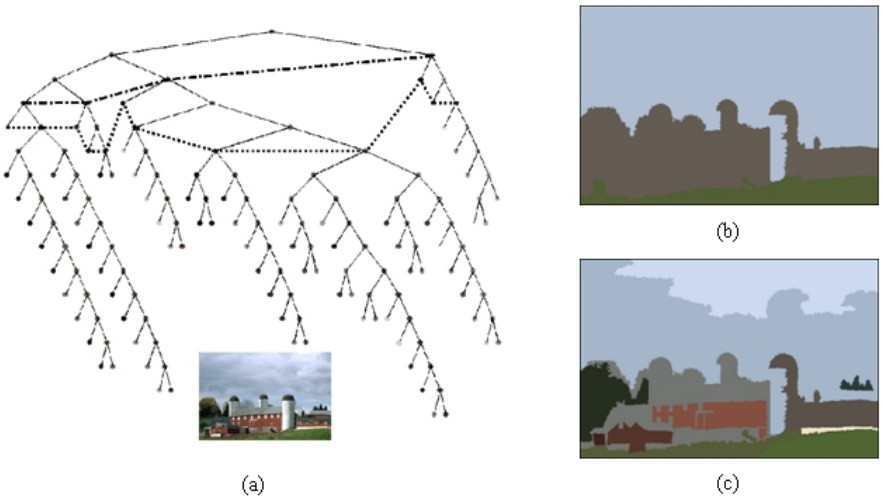


Figure 2. Two example pseudo tree levels. (b) and (c) Segmentations of the image corresponding to the pseudo tree levels indicated by the dash dotted and dotted lines respectively.

3. REGION EVOLUTIONS IN BPTS

From the subjective tests (Lu, H. et al, 2006), it is found that the discontinuities in region evolutions play an important role when browsing tree branches; they correspond to salient image content.

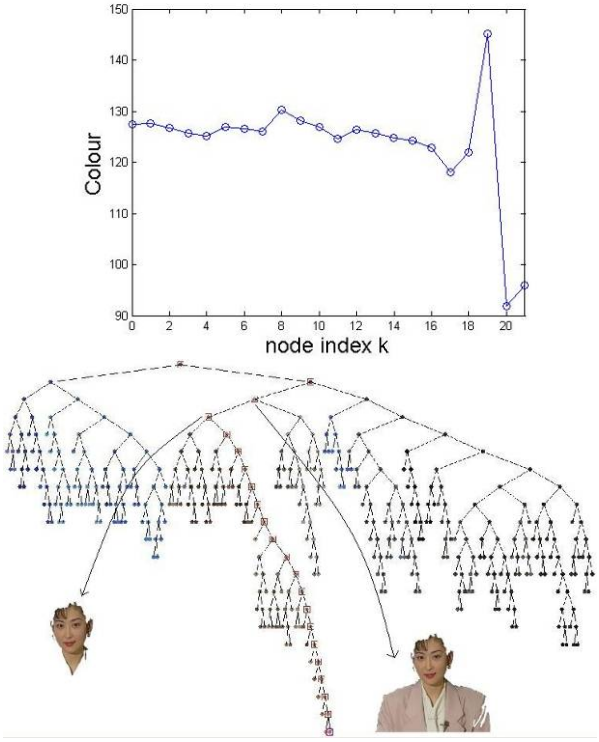
In this section, we formulate the region evolution as *evolvements* and provide a novel framework to detect the discontinuities. The tree analysis technique proposed here is completely separate from the BPT generation process, i.e. the analysis framework is independent of the tree creation.

Let $B = \{B_s, s = 1, \ldots, S\}$ denote the set of leaf nodes in the tree, where S is the total number of leaf nodes. From each leaf B_s to the root node, a unique upward path P_s is defined. Let L_s denote the number of nodes along the path P_s; k the node index ranging from 0 (the leaf node) to L_s (the root). An *evolvement* is defined as the changing history of a chosen region feature along a path P_s. It is a function of node index k and denoted as $e_{P_s}(k)$. In the following we drop the path condition from the evolvement functions for easy notation.

A number of different region features can be extracted from a tree node according to its underlying image properties. This makes the definition of an evolvement function quite flexible, with each of its possible variants having different characteristics. In this work, we propose the use of mean *CIE L*a*b** colour. For a path P_s, the mean colour evolvement function $e(k)$ is defined as

$$e(k) = \bar{c}(R_k), \quad k = 0, \cdots, L_s - 1. \tag{2}$$

Continuing to use Akiyo as an example, Figure 3(a) shows a typical mean colour evolvement curve. The corresponding BPT is shown in Figure 3(b) where the nodes associated with the path are enclosed by the square boxes. The tree has been simplified from Figure 1(b) using the technique described in Section 4 for visualisation. It is clear from the figure that there are two discontinuities in the curve at nodes 18 and 19 which correspond to Akiyo's face and a combination of her face and suit as shown in Figure 3(b).



(a) (b)

Figure 3. An example of mean colour evolvement. (a) The evolvement curve. (b) Its corresponding BPT.

When examining an evolvement, it is plotted against the node index k for ease of visualisation which makes *continuative* merges apparent. A continuative merge is the continuation of similar behaviour and is found on a linear segment in the plot. Examples include the growing of a homogeneous region or a series of equally difficult merges with heterogeneous constituents. In either case, the evolution of the region feature maintains a given gradient therefore resulting in a linear segment. A flat segment indicates a largely homogeneous region, whereas step or ramp behaviour is indicative of the merging of heterogeneous regions. When targeting salient content, the discontinuity from one type of segment to another is of particular interest. This exhibits as either concave or convex hulls in the plot. The onset of an evolvement step or ramp shows where a homogenous region is followed by heterogeneous, whereas the end of a step or ramp shows the end of a series of

heterogeneous merges leading to the creation of a larger region or the completion of the image. In this work, the onset and end points are detected by a modified second derivative of the evolvement with respect to the node index k. This is equivalent to the rate of change of the feature.

Given an evolvement $e(k)$, the node index k is discrete and uniformly increases from 0 to $L_s - 1$ with an interval of 1, the second derivative of $e(k)$ can be approximated by the difference

$$e''(k) = e(k-1) + e(k+1) - 2\,e(k), \quad k = 1, \cdots, L_s - 2 .$$

Note that $e''(k)$ is undefined at $k = 0$ and $L_s - 1$, *i.e.* the first and last node in the evolvement. The discontinuities in the evolvement is detected by the function $\mathrm{M}(k)$ which is defined as

$$\mathrm{M}(k) = Sign(k)\, mag(e''(k)) \tag{3}$$

where $mag(e''(k)) = |e''(k)|_2$ is the magnitude of $e''(k)$. $Sign(k)$ determines the sign of $\mathrm{M}(k)$ such that if the magnitude of the first derivative of $e(k)$ is increasing towards $k+1$ it is 1 otherwise -1.

4. TREE SIMPLIFICATION AND IMAGE FILTERING

Consider an evolvement $e(k)$ and its associated path P_s. If a transition from a willing to a reluctant merge occurs at node index k, resulting in a discontinuity in the evolvement, it implies that a growing homogenous region starts to merge with other heterogeneous regions. We can therefore reasonably trim all sub-branches below R_k whilst preserving important image detail between R_k and its heterogeneous sibling. If all such points can be detected and marked in the tree, the simplification can be done by pruning branches.

According to the framework given in the previous section, a transition from willing to reluctant is represented as the onset of a step or ramp in the evolvement and is detected by the modified second derivative function $\mathrm{M}(k)$ (3) w.r.t. the node index k. In the current implementation, the tree node associated with the first salient transition along the path according to a threshold is considered as a candidate for pruning. It is detected at C if C is the smallest value in $[1, L_s - 2]$ for which

$$\mathrm{M}(C) > \alpha$$

where α is a user-supplied threshold which controls the significance of simplification. The higher the value of α, the more simplified the tree becomes.

In general, there may be several noticeable peaks along $\mathrm{M}(k)$ resulting in multiple candidates for pruning. However, experience shows (Salembier, P. et al, 1998) that the first detected peak is robust compared to approaches which employ the median, highest, or last.

Consider a parent node at k and its child at $k-1$ which are both flagged as candidates. The child (the first peak) is selected since choosing the parent causes the salient detail identified by the child to be lost. This approach is equivalent to the Min Decision technique described in (Salembier, P. et al, 1998). This method also permits a hierarchical simplification by applying a series of increasing thresholds to the original tree.

The unified approach for tree simplification and image filtering is given as follows. Let T denote the tree to be processed. Set an appropriate value for the threshold α.

1) Make an empty pool Ω
2) For each upward path from a leaf node to the root
    a. Generate its evolvement according to (2).
    b. Compute the modified second derivative of the evolvement, $\mathrm{M}(k)$ (3).
    c. Find the first peak in $\mathrm{M}(k)$ which is higher than the threshold α.
    d. Save the corresponding tree node to the pool Ω and proceed to the next path.
3) Find the lowest possible pseudo tree level L in Ω by using the blending algorithm described in Section 4.1.

If each region in L is represented by its average colour, the reconstructed image is a filtered image with salient content preserved. In tree simplification, all the sub-branches below the pseudo level L are pruned and the tree is redrawn.

4.1 Blending algorithm

The output of step 2) in the tree simplification algorithm is a big pool Ω which contains all the nodes having the potential to be new leaf nodes. They are, however, chaotic in nature as each upward path is processed independently meaning that some candidate nodes may have either a direct or indirect ancestor-descendant relationship. To have a set of non-overlapping regions covering the entire image, the pool Ω has to be processed. We propose a blending algorithm which preserves the maximum information available in the pool by constructing the lowest possible pseudo tree level L in Ω. The algorithm is given as follows.

1) Make an empty list L.
2) Assign a mask value of 0 to each node in the tree.
3) For each node in Ω, find its direct ancestors and set their masks to 1.
4) For each leaf node in the tree. If its mask is 0, follow its upward path and find the first node with a parent mask of 1. Save this node to L and set the masks of its underlying leaf nodes to 1.

The blending algorithm proposed here is analogous to the Min Decision technique (Salembier, P. et al, 1998). The main difference is that the min rule is applied to each individual path whereas the blending algorithm has taken into account the interactions between different paths across the entire tree.

Instead of constructing the lowest pseudo tree level, the highest level, which preserves the minimum information in the pool, can also be generated. Results, however, are semantically destructive and not suitable for further analysis.

4.2 Setting threshold α

Setting a suitable value for the threshold α is important as it controls the amount of information to be removed from the tree. However, it is a non-intuitive problem and often solved based on a heuristic. Therefore, in addition to setting the threshold α directly, we also develop a method which allows the user to set the final number of leaf nodes in the simplified tree.

The problem is formulated as the optimisation of α under the constraint that the resulting tree has the target number of leaf nodes. A simple iterative bisection procedure is employed. Let α_h and α_l denote the higher and lower bounds for α. The algorithm starts by simplifying the original tree according to $\alpha_m = (\alpha_h + \alpha_l)/2$ and testing to see in which of the subintervals $[\alpha_l, \alpha_m]$ or $[\alpha_m, \alpha_h]$ the resulting number of leaf nodes lies. The algorithm is then repeated with the new interval as often as needed to locate the target number of leaf nodes to a desired accuracy. The initial lower and higher bounds for α are set to the minimum and maximum values of $\mathrm{M}(k)$ in the tree. In practice, the optimal α is normally obtained with only a few iterations and the computational overhead is small (Section 5 analyses in terms of computer CPU running time).

5. EXPERIMENTAL RESULTS

In the following, we demonstrate an example of tree simplification. The original image is obtained from the Berkeley Segmentation Dataset (BSD) (Martin, D. et al, 2001) and is shown in Figure 4(a). Its corresponding BPT is shown in Figure 4(d). The original BPT is simplified by setting a target number of leaf nodes to 300 and the result is shown in Figure 4(e). Figure 4(b) shows the reconstructed image from the leaf nodes by using random colours for better visualisation.

To demonstrate the advantages of our method, we have also implemented a simple *direct approach*. It finds the pruning branches by progressively deactivating the nodes following the merging sequence until the required number of leaf nodes is generated. Its outcome, therefore, solely depends on the merging order used to generate the original tree. This method has been used for the segmentation purposes in (Salembier, P., 2000). The resultant tree and its corresponding reconstructed image are shown in Figures 4(f) and (c), respectively. It is observed from these figures that the direct method favours regions of similar size irrespective of their salient value to the image, for example, the sky. The reason for this behaviour is that the merging order is calculated based on a size weighted colour distance (see equation (1)). If these figures are compared with Figure 4(e) and (b), it is evident that the BPT simplified by our method is more object-oriented since the pruning candidates are detected based on their saliency to the image content. Therefore, the new set of leaf nodes generated by our approach contains regions with different sizes and tends to describe the semantically meaningful objects in the image.

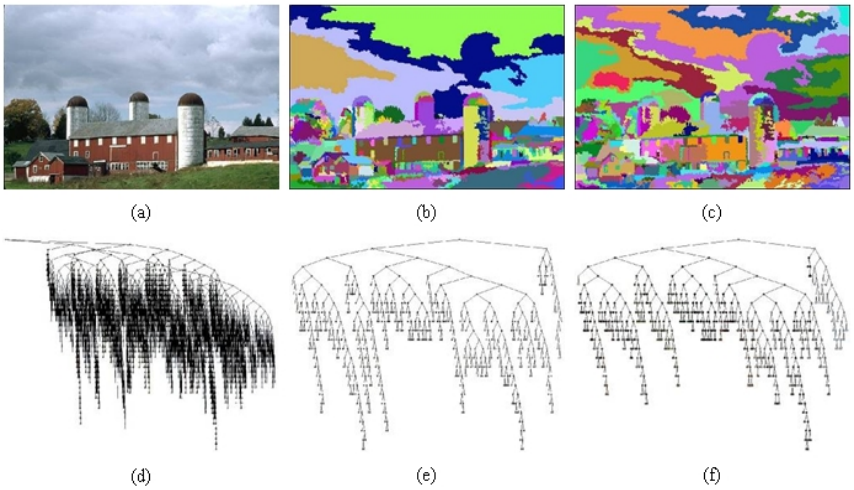


Figure 4. An example of tree simplification. (a) Original image. (b) and (c) Reconstructed images from the BPTs shown in (e) and (f). (d) Original BPT. (e) and (f) BPTs simplified by the proposed and the direct approach, respectively.

More simplification results are shown in Figure 5 where the original images are chosen from BSD image database. The first row shows the original images, and the images reconstructed from the BPTs simplified by our method are shown in the second row. The bottom row shows the images reconstructed from the BPTs simplified by the direct approach. The target number of leaf nodes is set to 300 for all tests.

All the simplification experiments conducted in this section were performed on an Intel Pentium 4 1.7GHz computer with 512 MB memory. Table 1 shows the averaged CPU time required for the different stages of a tree simplification process. From this table, it can be seen that the BPT generation pipeline demands the most computer power whereas the post-processing stages including the threshold α optimisation and final branch pruning are much less expensive to compute.

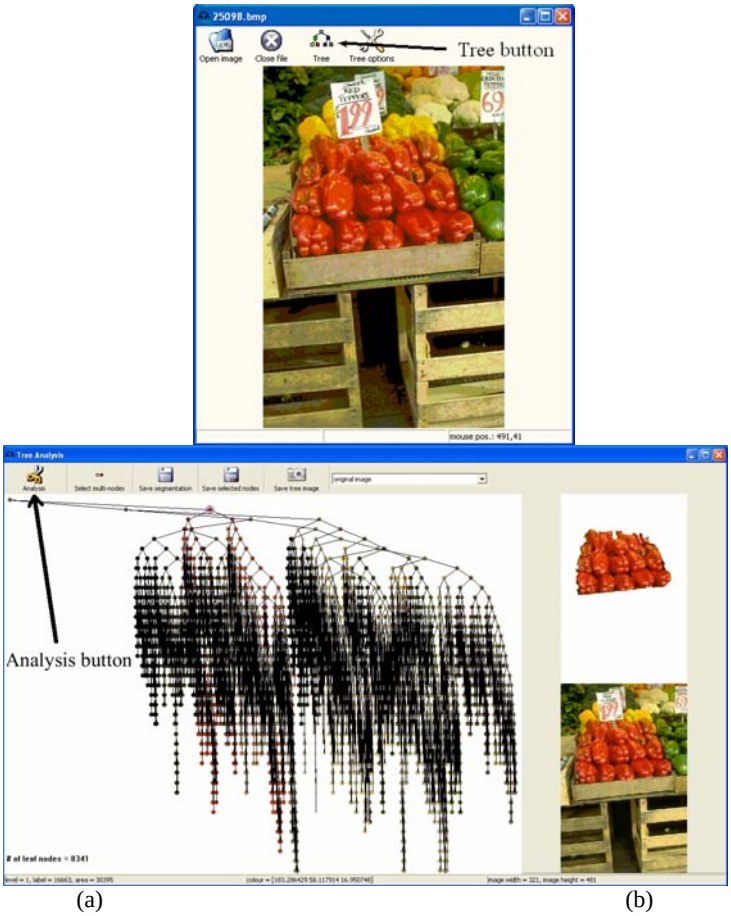
Table 1. Computational complexity of the tree simplification in terms of average CPU running time in seconds

BPT	Evolvement	α optimisation	Pruning	Total
7.81	0.82	0.82	0.82	10.27

Figure 5. Tree simplifications applied to the BSD images 16052, 25098, 172032 and 388016. From top to bottom, the original images, reconstructed images from BPTs simplified by our method, reconstructed images from BPTs simplified by direct approach.

6. HUMAN-COMPUTER INTERACTION

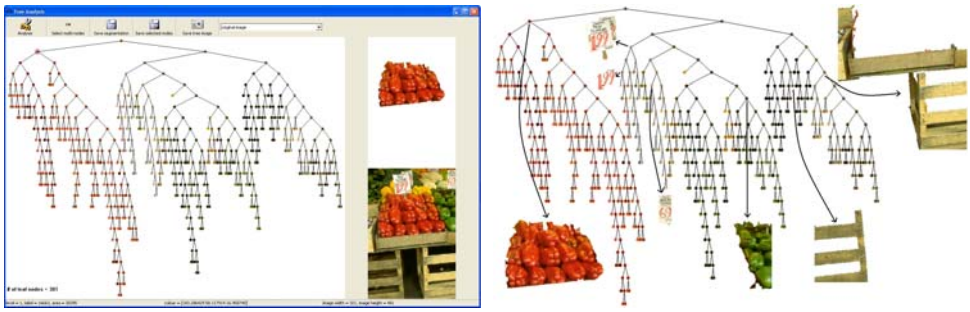
A software platform has been built which enables the manual extraction of semantic objects, image segmentation and filtering in the BPT domain. It is written in C++ and developed under Windows XP SP2. The main user interface is shown in Figure 6(a). The BPT is created by pushing the Tree button on the toolbar. Once the BPT creation process is finished, a new user interface called *tree frame*, visualising the tree structure is created as shown in Figure 6(b). Each tree node is represented by the mean colour of its underlying image area and can be examined by clicking on it. After each attempt the pixel domain equivalent of the selected node is fed back to the user and displayed in the upper right corner of the frame. The image reconstructed from the leaf nodes is displayed in the bottom right of the frame.



(a) (b)

Figure 6. Human-Computer interaction interface. (a) Interface used to load images. (b) Tree frame.

To make the visualisation of a tree simpler and uncluttered for ease of branch browsing, the BPT can be simplified by pushing the Analysis button. A dialog will pop up asking for the target number of leaf nodes. Once the simplification is finished, the tree is redrawn as shown in Figure 7(a). Given this simplified version of the tree, the user can easily navigate between different braches and achieve a fast object extraction as demonstrated in Figure 7(b) where the semantic objects can be easily accessed.



(a) (b)

Figure 7. Tree frame with simplified BPT. (a) Tree frame showing a simplified tree. (b) Semantic object extraction.

The filtered image obtained by reconstructing the leaf nodes can be accessed by selecting the "segmented image (mean colour)" in the drop-down menu of the tree frame as shown in Figure 8(a). The boundaries of the regions are shown in Figure 8(b).

(a) (b)

Figure 8. Filtered image. (a) Filtered image where each region is represented by its mean colour. (b) Boundaries of the regions.

Individual tree branches often relate to semantic objects contained in the image. However this is not guaranteed and an object's constituents may be found across different branches because of the way the merging process has evolved from the seed regions. Prior knowledge of the scene imposed on the merging process can alleviate this problem to some extent. For example, in (Salembier, P. and Gerrido, L., 2000), the mask of a target object is used to impose constraints on merging in such a way that the object of interest is represented with a single node in the tree. In this work, we allow the user to select multiple nodes to represent the object of interest by enabling the "Select multi-nodes" option in the tree frame. Figure 9 shows where two tree nodes are selected to represent the wooden boxes supporting the vegetables.

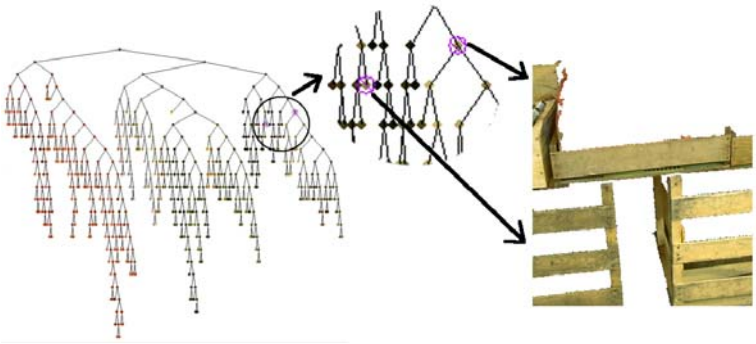


Figure 9. Multiple node selection.

7. CONCLUSION

Images represented by the binary partition tree are suitable for region based image analysis. Each branch in a tree potentially describes a semantic object in the original image. Object extraction in the BPT domain therefore becomes a problem of locating prevalent tree branches. However, the trees generated from real world images are generally large and contain thousands of nodes making manual retrieval labour intensive. The large number of nodes when plotted tend to occlude one another, coupled with limited display space visualisation of the tree is difficult. In this paper, we present a novel BPT analysis framework from which tree simplification and image filtering are derived. Experimental results demonstrate that the framework is able to preserve salient image content whilst the tree density is significantly reduced. Analysis of the computational complexity for the framework also shows that the overhead of the method is low. A complete software package is built which implements the proposed framework and provides an intuitive human-computer interaction interface allowing fast and efficient object extraction, image segmentation, and filtering inviting commercialisation.

REFERENCES

Avaro, O. and Salembier P., 2001. MPEG-7 systems: overview. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, pp 760-764.

Bennstorm, C. and Casas, J., 2005. Object representation using colour, shape and structure criteria in a binary partition tree. *IEEE International Conference on Image Processing*, vol. 3, pp. 1144-1147.

Battista, S. et al, 2000. MPEG-4: a multimedia standard for the third millennium. *IEEE Multimedia*, vol. 7, pp 76-84.

Collins, R., 2003. Mean-shift blob tracking through scale space. *Proceedings of the IEEE Computer Society Conference on Computer Vision and pattern Recognition*, vol. 2, pp. 234-240.

Dorea, C. et al, 2005. A motion based binary partition tree approach to video object segmentation. *IEEE International Conference on Image Processing*, vol. 2, pp. 430-433.

Gonzalez, R. and Woods, R., 2002. *Digital Image Processing*. Prentice Hall.

Lu, H. et al, 2006. Generic object registration using multiple hypotheses testing in partition trees. *IEE Proceedings on Vision, Image, and Signal Processing*, vol. 153, no. 3, pp 323–330.

Martin, D. et al, 2001. A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. *Proc. 8th Int'l Conf. Computer Vision*, pp 416–423.

Moen, S., 1990. Drawing dynamic trees. *IEEE Software*, vol. 7, no. 4, pp. 21–28.

Salembier, P. et al, 1998. Anti-extensive connected operators for image and sequence processing. *IEEE Transactions on Image Processing*, vol. 7, no. 4, pp. 555-570.

Salembier, P. and Gerrido, L., 2000. Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval. *IEEE Transactions on Image Processing*, vol. 9, no. 4, pp 561–576.

Salerno, O. et al, 2004. Object recognition based on binary partition trees. *IEEE International Conference on Image Processing*, vol. 2, pp. 929-932.

Sikora, T., 2001. The MPEG-7 visual standard for content description - an overview. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, pp 696-702.

Tsaig, Y. and Averbuch, A., 2002. Automatic segmentation of moving objects in video sequences: a region labeling approach. *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 7, pp 597-612.

Vincent, L. and Soille, P., 1991. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 6, pp 583–598.