Image Segmentation by Construction of Shortest Spanning Tree Using Prim's Algorithm

S Sibi, Swarna Priya R.M., Pranov Kumar, Raghavendra Singh, Anitha Baskar

Abstract— The paper aims at segmenting an image. The goal of image segmentation is to simplify and/or change the representation of an image into something that is more meaningful and simpler to analyze. In this paper we consider segmenting the image using the shortest spanning tree. To find the shortest spanning tree the image is first mapped into a graph. There have already been research conducted in this region of image processing where the segmentation of an image is performed using the shortest spanning tree. In the previous research papers the shortest spanning tree is found using the Kruskal's algorithm. In this paper we will find the shortest spanning tree using the Prims algorithm and compare the changes in the end results of the two researches conducted .Within the scope of this paper we will be proposing the use of prims algorithm for solving image segmentation.

Index Terms — image segmentation, Kruskal's algorithm, mapping, Prims algorithm, representation of image, shortest spanning tree

1 INTRODUCTION

Image segmentation has been a hot topic of research since a long time. In Image segmentation we divide the image regions into a number of regions based upon certain characteristics like intensity values. Etc.
Image segmentation is to classify or cluster an image into several parts (regions) according to the feature of image, for example, the pixel value or the frequency response. Up to now, lots of image segmentation algorithms exist and be extensively applied in science and daily but in this paper we focus upon how the concepts of graph theory can be used to segment the image A problem with any method of segmentation is to define the level of detail which is of interest. Sometimes fine detail is important, while in other cases larger-scale features are more significant. This variation in the 'scale of interest' means that most methods developed and currently used to date that are suitable for one situation may not be so for another. Then in order for segmentation techniques be effective in all cases, they must be able to describe regions in a hierarchical order of importance.

- *S.Sibi is currently pursuing bachelors' degree program in Information Technology engineering in Vellore institute of Technology, India, PH-09789732932. E-mail: sibisenth@gmail.com.*
- *Swarna Priya R.M. is currently Teaching for bachelors degree program in Information Technology engineering in Vellore institute of Technology, India.*
- *Pranov Kumar is currently pursuing bachelors' degree program in computer science engineering in Vellore Institute of Technology, India.*
- *Raghavendra Singh is currently pursuing bachelors' degree program in computer science engineering in Vellore Institute of Technology, India.*
- *Anita Baskar is currently pursuing bachelors' degree program in computer science engineering in Vellore Institute of Technology, India.*

For example, if two regions are required then the image must be partitioned in such a manner that the two most significant regions are found. If a third region is desired, a further partition must be made so that the next most significant region is obtained. Consequently, if only a few regions are generated, global features of the image are represented and, as more regions are generated, more detail in the image signal is uncovered. One must be wondering why especially graph is used. Other algorithms also are present for the same purpose but the graphs are preferred because of its many advantages. One of the advantages of using graph is that the connectivity information of graphs allows independent work to be performed on different parts of the original image, and allows one to reconnect them to yield an accurate result as if processing had occurred globally.
A number of other research work has been conducted on this topic in past. In 2004, Felzenszwalb introduced a segmentation method [1] based on Kruskal's Minimum Spanning Tree algorithm. The edges are considered on the basis of their weight and they are arranged in increasing order. Their end pixels are merged into a single region as long as a cycle is not caused in the graph, and while the pixels are 'similar' to the nearby regions' pixels. Cycles can be detected with the aid of a disjoint-set data structure.[2] This detection can be done in near constant time. Pixel similarity is arrived at by examining and comparing the weight to a per-segment threshold. The algorithm gives multiple disjoint MSTs as an output, i.e. a forest where each tree corresponds to a segment in the image. The complexity of the algorithm is quasi-linear as sorting edges is possible in linear time using the counting sort.

In this paper we have extended the work of Felzenszwalb[1] by using Prim's algorithm.

It is a greedy algorithm that finds a MST for an undirected weighted connected graph. The above statement means the algorithm finds a subset of the edges of the tree such that it includes all the edges and the total weight of all the edges in the tree is minimized. The Prims algorithm in contrast to the Kruskal's algorithm, finds the shortest spanning tree in an arbitrary manner. The minimum edge in the graph is removed and continuously added to the spanning tree till all the edges connect two vertices in the graph and all the vertices in the original graph is connected in the spanning tree.

As mentioned earlier, within the scope of this paper we will be proposing the use of prims algorithm for solving image segmentation and edge detection problems. We will be seconding our proposition with the help of mathematical models and simulations. We will also implement the proposed algorithm using programming tools like mat lab and octave and perform image segmentation on sample images. Finally, the paper will show a statistical and comparative analysis of the results that obtained

2 TERMS AND DEFINITION

2.1 Graph

A graph is an ordered pair G = (V, E) comprising a set V of vertices or nodes together with a set E of edges or lines, which are 2-element subsets of V (i.e., an edge is related to two vertices, and the vertices forming the edge are called the endpoints, or end vertices of the edge. But at the same time it is possible that a vertex in a graph may not belong to any edge. V and E are usually considered to be finite, and many of the well-known results are not true for infinite graphs because many of the arguments that are taken in the case of finite graphs fail in the infinite case. The order of a graph is $|V|$ (the number of vertices). A graph's size is $|E|$ the number of edges. For an edge {u, v}, graph theorists usually use the somewhat shorter notation uv the relation is represented as an unordered pair of the given vertices with respect to the corresponding edge.

3.2 Undirected graph

It is one in which the edges have no particular direction. The edge (a, b) is the same as the edge (b, a), that is they are not ordered pairs, but a set {u, v} (or 2-multisets) of vertices. Complete graphs are ones in which there exists an edge connecting every pair of vertices. Bipartite graphs are ones in which the the vertex set can be partitioned into two sets, W and X, such that no two vertices in W are adjacent and no two vertices in X are adjacent. We can also say that it is a graph having a chromatic number of 2.

3.3 Planar graph

It is a graph that can be embedded in the plane, that is, it can be drawn on a plane in such a way that its edges do not intersect anywhere except for at their endpoints. In other words, it is a graph where no two edges cross or overlap each other.[3] Such a drawing is called a plane graph or planar embedding of the graph. A plane graph can be defined as a planar graph with a mapping from every node to a point on a region of the plane, and from every edge to a curve on that plane, such that the outer most points of each curve are the points mapped from its end nodes, and all curves are disjoint except on their outer most points.

3.4 Tree

It is an undirected graph in which any two vertices are connected by exactly one simple path and the graph has no cycles. A forest is an undirected graph, which consists of a set of trees; We can also say that, the graph consists of a disjoint union of trees. Equivalently, a forest is an undirected cycle-free graph.

A tree is called a rooted tree if one vertex has been designated the root, in which case the edges have a natural orientation, towards or away from the root

3.5 Spanning tree

A spanning tree T of a connected, undirected graph G is a tree composed of all the vertices and some (or perhaps all) of the edges of G. Informally, a spanning tree of G is a selection of edges of G that form a tree spanning every vertex. That is, every vertex lies in the tree, but no cycles (or loops) are formed. Since a tree can have many spanning tree therefore here it becomes necessary to introduce Minimum Spanning tree. A minimum spanning tree (MST) is a spanning tree with the least weight possible. In other word, the spanning tree will have weight less than or equal to other spanning trees of the graph. Prim's algorithm for minimum spanning tree constructing.

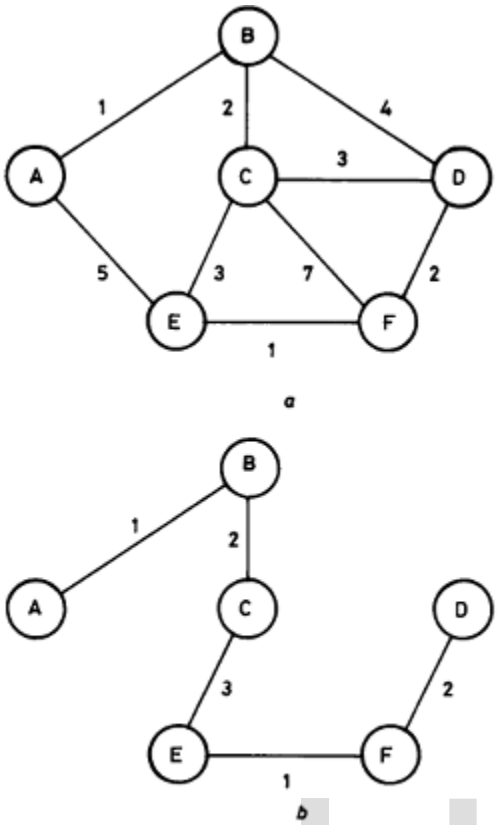

Fig. 1 *Example of a graph and its SST*
a Example graph
b SST of graph

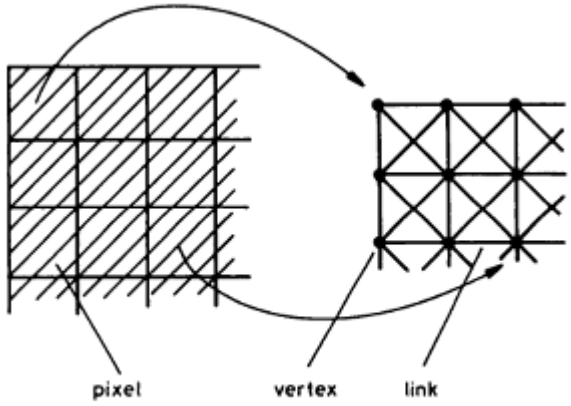

Fig. 2 *Mapping an image onto an eight-way-connected graph*

.

4 SPANNING TREES

4.1 Mapping images onto graphs

In order to analyze images using graph theory, the original image must be mapped onto a graph ready for processing. There are a number of possible mappings, but the most obvious one is to map each pixel in the image onto a vertex in the graph. The vertex weights can be made equal to the corresponding pixel intensities, or a more complex weight function can be used. Thus, if the image intensity at a point (x, y) is 7^, then the corresponding vertex weight is

Vi=Ix,y

where x, y is mapped onto i in a one-to-one mapping.

If the link weights are defined as the absolute value of the difference between the vertex weights that they join, i.e. the link weight is a measure of similarity between the two vertices, and hence between the corresponding two pixels. A vertex could be linked to any other, but a useful simplification is only to link vertices to their nearest neighbors. Either four or eight neighbors may be considered. Fig. 2 shows the mapping of an image onto an eight-way connected graph

4.2 Obtaining shortest spanning trees

A formal description of the SST is given in [4]. A more informal description is given here. For any tree in a forest, if an link with the lowest weight which connects that tree to another is added, then the link will be in the SST. This is true even if there are links in the forest not in the SST (8, 9). If the forest forms part of an SST then the complete SST can be found by successively adding these new links to it. The SST, therefore, tends to include links with low weights. More costly links are only included when there is no alternative. Algorithms for finding an SST of a graph are well known (4-6). As the graphs to be considered in this paper are very sparse, there being only a few links per vertex, prim's algorithm is used. This algorithm has an execution time proportional to O(£ log (V)), where E is the number of links and V is the number of vertices in the graph. Faster algorithms have been proposed [4] which would be useful if speed were a limiting factor.

4.3 Forming segmentations from spanning trees

A spanning tree of a graph can be used to form a partition by cutting its links. This is because the forest so produced consists of a set of disjoint trees which are used to define each partition. Vertices in each tree of the forest describe the pixels in that partition. The forest still spans the graph, so that no vertices are left out of the segmentation. If T is a tree in a forest, then it defines a partition P(T)

$$P(T)_i = \begin{cases} 1, \text{ if } V_i \in T \\ 0, \text{ otherwise} \end{cases} \quad (1)$$

Once a partition of the graph has been obtained, the reverse problem must be considered; that is the mapping of the graph back onto an image. A method of doing this to generate a segmentation image

Sx,y, such that each pixel in any one region is assigned a constant intensity value which is the mean of all the pixels within that region. Thus, let p(T) be the partition weight, such that

$$p(T) = \sum_i P(T)_i \cdot v_i \Big/ \sum_i P(T)_i \qquad \forall i \tag{2}$$

If Mx,y is the mapping from (x, y) to T, then

$$S_{x,y} = P(M_{x,y})$$

Now that all the necessary principles have been introduced, we proceed to the explanation of the SST segmentation, give an example and discuss its properties.

5 SST SEGMENTATION

5.1 Principles of SST segmentation

The object of the segmentation method is to group together pixels that are similar in some sense, and to separate pixels that are dissimilar according to the same measure. A simple measure of similarity between pixel values is to take the absolute value of the difference used in eqn. 2. In the graph the most similar vertices are linked by locating the lowest link weights. This is done for the whole graph by finding an SST. As the link weight function is to be minimized, any monotonically increasing distance function can be used, but the function of eqn. 2 is a simple one to implement. Once the SST has been found it can be used to form a segmentation by cutting it at its highest link weights, so forming partitions that differ from their neighbors by the maximum amount. This guarantees that each pixel has neighbors in its own region which have values closer to its own than any neighboring pixel in a different region. If a further regions required by making a further cut in the SST, then the next most contrasting region is found. This means that the segmentation is hierarchical, providing regions in order of maximum contrast. A simple segmentation algorithm can now be envisaged to form N regions:

1. Map the image onto a weighted graph.
2. Find an SST of the graph.
3. Cut the SST at the N - 1 most costly links.
4. Map the forest onto a segmentation image.

Improved segmentation methods based on the SST

Three distinct methods for improving the SST segmentation are now described which make more complex use of the SST in order to use the global information in the image. These are:

Recursive SST segmentation.

Minimax SST segmentation.

Local averaging SST segmentation.

5.2 Recursive SST segmentation

The recursive SST segmentation presents a powerful solution to the problem of incorporating global information into the segmentation method. If an image has M pixels, then a simple algorithm to find N regions is:

1. Copy the image into an image buffer.
2. For / = M - 1 down to N, do:
3. Generate a /-region segmentation using the SST segmentation from the image buffer.
4. Copy the segmentation image into the image buffer.

The algorithm expressed in this form is inefficient, but it nevertheless illustrates an important principle. At each stage of the recursion a segmentation image is generated with one fewer region than the stage before. Pixel values for each region in the segmentation image are constant and equal to the mean of the pixels that the region covers in the original image. As the pixel values are modified according to all the pixels in the region, it is possible for one pixel to affect another which is not necessarily its nearest neighbor. The link weights between regions will also be altered to take account of all the pixels in both the regions that the link connects, so that global region information is incorporated into the link weights. Initially, only local information is used as each region consists of only one pixel, but as the recursion progresses and the regions enlarge in size the graph contains more global information. As every pixel in each region is the same, the graph can be simplified so that a vertex represents a whole region. When the SST is cut into a forest to describe the partition, each tree can now be thought of as a single entity, which becomes a single vertex. The graph-to-image mapping is now a one-to-many mapping, and the partition P(i) defined by a single vertex V is redefined as:

$$P(i)_{x,y} = \begin{cases} 1, \text{ if } V_i \text{ maps onto } (x, y) \\ 0, \text{ otherwise} \end{cases} \tag{3}$$

The vertex weight function is now

$$v_i = \sum_{x,y} P(i)_{x,y} I_{x,y} \Big/ \sum_{x,y} P(i)_{x,y} \qquad \forall x, y \tag{4}$$

The initial mapping from the image to the graph is the same as before, but the structure of the graph is altered during the recursion. Improved recursive SST segmentation can now be envisaged using the new graph. Initially, each pixel is considered to be a separate region, and each iteration of the recursion merges two regions together. At each stage the two most similar vertices, Vi and V, are merged across the shortest link. The new vertex Vki that is formed is assigned the mean value of the new partition:

$$v_k = \frac{\sum_{x,y} (P(i)_{x,y} I_{x,y} + P(j)_{x,y} I_{x,y})}{\sum_{x,y} (P(i)_{x,y} + P(j)_{x,y})} \qquad \forall x, y, \tag{5}$$

and the new partition P(k) is the union of two most similar partitions:

$$P(k)_{x,y} = P(i)_{x,y} + P(j)_{x,y} \tag{6}$$

The link that joined vertices V and Vj is removed and saved for use when mapping the graph back to the image domain. All the other links that had vertices Vi and Vj at their ends have their weights re-evaluated using the new vertex weight. If there are now any links which are identical then the redundant ones are removed. This stage is repeated until there are no links left. Since the terminal vertices of each of the saved links are merged at each stage of the recursion, the next stored link can never form a circuit with links previously saved. These links thus form a spanning tree of the original graph, although not necessarily an SST. They can be used to obtain a recursive SST segmentation of the image rapidly for any number of regions N by cutting the tree at the last N — 1 links found by the recursion. The recursive SST segmentation algorithm for obtaining N regions is therefore:

1. Map the image onto a weighted graph.
2. While there is more than one vertex in the graph.
3. Find the next least weighted link.
4. Save the link.
5. Merge the two vertices joined by this link.
6. Recalculate the new vertex weight and link weights.
7. Remove duplicated links.
8. Form a spanning tree with the saved links.
9. Cut the spanning tree at the N — 1 most costly l inks to form a spanning forest.
10. Map the forest onto a segmentation image.

The most important improvement over the SST segmentation is that global information about the image is used more and more as the regions are merged. The result is that if segmentation with many regions is required, then regions are found which are locally significant. Segmentations with fewer regions will provide more globally significant regions. Segmentations are produced that find equally contrasting regions for a given number of regions N. The 'scale of interest' is automatically adjusted to the required number of regions. The recursive SST segmentation is therefore hierarchical with respect to the 'scale of interest' of the image, as well as to the contrast between regions as in the SST segmentation. Globally contrasting regions take precedence over locally contrasting regions as the number of regions is reduced. At each stage of the recursion the boundary common to the two regions to be merged is removed, but the rest of the segmentation boundaries remain unaltered. As the vertex weight is an average of all the pixel intensities in the partition, noise becomes less of a problem as the segmentation progresses.

5.3 Minimax SST segmentation

The recursive SST segmentation is one way that global information may be incorporated into the segmentation. Another is to partition the graph into sub-graphs, such that a suitable objective function evaluated on each of the resulting sub-graphs is minimized. Unfortunately, algorithms for the solutions of this problem require evaluation of all possible partitions of the graph [5]. This is impracticable for a graph representing an image of any reasonable size. The minimax SST segmentation finds a suboptimal solution to this problem in a computationally efficient way. If the SST is cut into a spanning forest F of trees T, then a cost function for each tree, c(T), is defined:

$$c(T) = \max [\,|v_i - v_j|\,] \qquad \forall i, j \in T \tag{7}$$

The minimax SST segmentation algorithm for obtaining a N-region segmentation is:

1. Map the image onto a weighted graph.
2. Find an SST of the graph.
3. Repeat N times:

4. Find the tree T max with the largest cost c(T), such that:

$$c(T_{max}) = \max [c(T)] \qquad \forall T \in F \tag{8}$$

5. Cut T max at the link E which minimizes the maximum value of the costs of T the two sub-trees so created:

$$\min\ [\max\ [c(T_i), c(T_j)]] \qquad \forall E_{i,j} \in T_{max} \tag{9}$$

5.4 Local averaging SST segmentation

In previous Sections global information was introduced into the algorithms, either by modifying the graph as the segmentation progressed in the recursive SST segmentation, or by cutting the original SST so as to optimize a cost function defined over the resultant trees in the minimax SST segmentation. Regional information can be built into the original graph directly by making the vertex weights depend on an area around the corresponding pixel in the image. A consequence of this is that the algorithm is more stable when the image is noisy. The only difference between this method and the SST segmentation or the recursive SST segmentation is in the assignment of vertex weights in the graph. The new vertex weight function is

$$v_i = \sum_{x=x-N}^{x+N} \sum_{y=y-N}^{y+N} I_{x,y}/(2N+1)^2 \tag{10}$$

so that Vi is the local mean of the region around pixel (x, y) for an area of 2N + 1 pixel square. Fig. 6 uses the original image with white noise added (signal/noise ratio of 10 dB) to compare the local averaging SST segmentation (with N = 1) with the SST segmentation. The local averaging method has the disadvantage that it may produce spurious regions along real boundaries, because the edge is spread out by the averaging. This means that for noiseless images it is not as good as taking the difference between pixel values. However, when the image is noisy, it performs considerably better than the original method, as the segmentation image is much less speckled. 4.4 Comparison of segmentation methods The three improvements to the SST segmentation method proposed in Sections 4.1 to 4.3 are not unrelated. The local averaging SST segmentation method differs only in the mapping from an image to a graph, and so could be incorporated into the other two methods. The minimax SST segmentation varies in its method of cutting the spanning tree, and so could also be applied to the recursive SST segmentation method, which is the only method to find a different spanning tree from the basic SST segmentation. Fig. 5 shows how the minimax SST segmentation gives results comparable to those of the recursive SST segmentation shown in Fig. 4. The results are not as good, but the method is much simpler and faster to run. It illustrates the importance of cutting the SST in an intelligent way.

Table 1 lists the main properties of the four SST Segmentation methods

Table 1: Comparison of SST segmentation method properties

	SST	Recursive SST	Minimax SST	Local average SST
Number of regions definable	Yes	Yes	Yes	Yes
Noise immunity	Poor	OK	OK	Good
Region accuracy	Good	Excellent	Good	Blurred
Speed	Fast	Slow	OK	Fast
Neighbourhood of information	Local	Global	Global	Mask

Each region is split using histograms of a number of its properties such as intensity or variance. The partitions to be made in the histograms are found using a set of six heuristics. The regions so produced are themselves split recursively until an end condition is met. The main properties of this segmentation are:

1. When splitting a region no use is made of the relationship between the region and its neighbors, and so no global information outside the region is used.
2. Large features are found roughly, but smaller features are not located accurately. The histograms are a poorer estimate of the probability density functions as the regions become smaller.
3. At each stage of the segmentation the method produces a variable number of regions which cannot be specified beforehand. The number of regions produced cannot be controlled directly.
4. Noise blurs the region histograms, resulting in less accurate region splitting. The method therefore has a low noise immunity.

Fig. 1b is a split-merge segmentation [2, 7] of the same image. The image is described by quarters, which are themselves made up of quarters in a recursive fashion. These quarters are represented by a quad-tree in which nodes are split or merged depending on which best would satisfy a minimax cost function. Properties of the split-merge method are given here for comparison:

The segmentation is made up of block-like regions because of the rigidly defined structure of the quad tree.

The method has a higher noise immunity, because the weight function is a minimax function defined over the total area of each region.

The number of regions required can be defined.

6 CONCLUSIONS

Methods of segmentation and edge detection based on graph-theoretic descriptions of images have been developed in this paper. The main features of these methods are that region boundaries are found very accurately, and that the description of images is hierarchical. Comparisons of these methods are made with others, and some implications for other areas of research have been outlined. The paper illustrates the importance of prims algorithm in case of larger images. Prims, being a vertex centric algorithm take much less time and resource than any other technique.

References

[1] *P. Felzenszwalb, D. Huttenlocher: Efficient Graph-Based Image Segmentation. IJCV 59(2) (September 2004)*

[2] *G. Harfst, E. Reingold: A Potential-Based Amortized Analysis of the Union-Find Data Structure. SIGACT 31 (September 2000) pp. 86–95*

[3] *Trudeau, Richard J. (1993). Introduction to Graph Theory(Corrected, enlarged republication. ed.). New York: Dover Pub. p. 64. ISBN 978-0-486-67870-2. Retrieved 8 August 2012. "Thus a planar graph, when drawn on a flat surface, either has no edge-crossings or can be redrawn without them."*

[4] *Cheriton, D., and Tarjan, R.E.: 'Finding minimum spanning trees', SI AM J. Comput., 1976, 5, pp. 724-42*

[5] *Bellmore, M., and Jensen, P.A.: 'An implicit enumeration scheme for proper cut generation', Technometrics, 1970, 12, pp. 775-788*