CONF-9611123--1

A Segmentation Algorithm for Noisy Images*

Ying Xu, Victor Olman, and Edward C. Uberbacher

Informatics Group
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6364

Submitted to the *IEEE Symposium on Image, Speech, Natural Language Systems,* Washington, D.C., November 4-6, 1996.

*Research was supported by the Office of Health and Environmental Research, U.S. Department of Energy under contract No. DE-AC05-960R22464 with Lockheed Martin Energy Research Corporation.

DISCLAIMER

A Segmentation Algorithm for Noisy Images

Ying Xu, Victor Olman, and Edward C. Uberbacher

Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6364

Abstract

This paper presents a 2-D image segmentation algorithm and addresses issues related to its performance on noisy images. The algorithm segments an image by first constructing a minimum spanning tree representation of the image and then partitioning the spanning tree into subtrees representing different homogeneous regions. The spanning tree is partitioned in such a way that the sum of gray-level variations over all partitioned subtrees is minimized under the constraints that each subtree has at least a specified number of pixels and two adjacent subtrees have significantly different "average" gray-levels. Two types of noise, transmission errors and Gaussian additive noise, are considered and their effects on the segmentation algorithm are studied. Evaluation results have shown that the segmentation algorithm is robust in the presence of these two types of noise.

Introduction

Image segmentation is one of the most fundamental problems in low-level image processing. The problem is to partition (segment) an image into connected regions of similar textures or similar colors/gray-levels, with adjacent regions having significant dissimilarity. Many algorithms have been proposed to solve this problem (see surveys [3, 4, 5, 7]). Most of these algorithms fit into two classes: (1) boundary detection-based approaches, which partition an image by discovering closed boundary contours, and (2) region growing and clustering-based approaches, which group "similar" neighboring pixels into clusters. Region-based approaches can be further divided into two classes, those using global information in segmentation and those using only local information. Many of the global information-based segmentation algorithms are very computationally time-consuming, while the local information-based approaches are usually sensitive to noise.

In this paper, we present an efficient region-based segmentation algorithm based on a minimum spanning tree representation of a gray-level image and a tree partitioning algorithm. With a tree representation, the algorithm partitions an image into a number (not predetermined) of arbitrarily-shaped regions (represented as subtrees) by globally minimizing the sum of gray-level variations over all regions under constraints which prevent partitioning a homogeneous region (a region with similar gray levels) into smaller regions. A tree structure is simple enough to facilitate a fast segmentation algorithm, and is also general enough to represent all the hierarchical relationships among objects present in an image. By combining the tree representation and global optimization, our segmentation algorithm is robust even on very noisy images.

The basic idea of the algorithm can be described as follows. It first builds a weighted planar graph representation of the given gray-level image, and constructs a minimum spanning tree of the graph in such a way that a connected homogeneous region corresponds to one subtree of the spanning tree. Then the algorithm partitions the spanning tree into a set of subtrees, whose vertices have similar gray levels. The tree-partitioning is done by a fast dynamic programming algorithm. If we use N, $\mathcal{I}$ and L to represent the number of pixels in the image, the number of gray levels used, and the size of a smallest partitioned region, respectively, the algorithm segments an image in $O(\max\{(N-L),1\}\mathcal{I}(\log(\mathcal{I})+L^2))$ time and $O(NL\mathcal{I})$ space. Hence in applications where $\mathcal{I}$ is a small constant, the algorithm runs in $O(\max\{(N-L),1\}L^2)$ time and $O(NL)$ space.

We have studied how two types of noise, transmission errors and Gaussian additive noise, affect the performance of this segmentation algorithm. We have shown that (1) both types of noise have very little effect on the minimum spanning tree construction algorithm, i.e., the property that an originally homogeneous region corresponds to one subtree of the spanning tree will generally not be affected by noise; (2) transmission errors, in general, have less effect on the performance of the tree-partitioning algorithm than Gaussian additive noise does; (3) averaging improves the performance of the segmentation algorithm when having Gaussian additive noise while does the opposite in the presence of

transmission errors.

The development of this algorithm is a part of the visGRAIL project at Oak Ridge National Lab [8]. visGRAIL is a multi-sensor/neural network based satellite imagery processing/pattern recognition system. The system recognizes areas of significant structure in satellite imagery by recognizing regular features from edge detection and geometric shapes and by combining the features using a neural network system. In visGRAIL, the region segmentation algorithm serves as a front-end of the shape recognition subsystem and also serves as a robust edge-detection subsystem (boundaries of partitioned regions).

Segmentation Algorithm

This section first presents a minimum spanning tree representation of a gray-level image, and then gives a fast tree-partitioning algorithm[1].

Spanning tree representation of an image

We assume that an image is given as a $p \times q$ array and each pixel has an integral gray level $\in [0, \mathcal{K}]$. For a given image I, we define a weighted (undirected) planar graph $G(I) = (V, E)$ as follows: The vertex set $V = \{$ all pixels of $I \}$ and the edge set $E = \{(u, v) | u, v \in V$ and $distance(u, v) \leq \sqrt{2} \}$, with $distance(u, v)$ representing the Euclidean distance in terms of the coordinates of the image array; Each edge $(u, v) \in E$ has a weight $w(u, v) = |\mathcal{G}(u) - \mathcal{G}(v)|$, with $\mathcal{G}(x) \in [0, \mathcal{K}]$ representing the gray level of a pixel $x \in I$.

A *spanning tree* T of a connected graph $G(I)$ (note that $G(I)$ is connected) is a connected subgraph of $G(I)$ such that (1) T contains every vertex of $G(I)$, and (2) T does not contain cycles. A *minimum* spanning tree is a spanning tree with a minimum total weight.

A minimum spanning tree of a weighted graph can be found using greedy methods [1], as illustrated by the following strategy: the initial solution is a singleton set containing an edge with the smallest weight, and then the current partial solution is repeatedly expanded by adding the next smallest weighted edge (from the unconsidered edges) under the constraint that no cycles are formed until no more edges can be added. For the above defined planar graph $G(I)$, a minimum spanning tree can be constructed in $O(\|V\| \log(\|V\|))$ time and in $O(\|V\|)$ space, where $\|\cdot\|$ denotes the cardinality of a set.

We have used Kruskal's algorithm [6] to construct a minimum spanning tree. An important property of a minimum spanning tree representation obtained by Kruskal's algorithm is that it tends to group neighboring pixels of similar gray levels together, which is summarized by the following lemma and Theorem 2 in Section 3.

[1]For a more complete description of the algorithm, we refer the reader to [9].

Lemma 1 *If each object (background) is represented by a connected region and each object (background) has a uniform gray-level (assuming that adjacent objects have different gray levels), the pixels representing each object form one (connected) subtree of the minimum spanning tree obtained by Kruskal's algorithm.* □

Tree partitioning algorithm

The goal of tree-partitioning is to partition a tree into subtrees so that each homogeneous region (possibly corresponding to an object) of an image is represented by a partitioned subtree. One way to achieve this is to partition the tree into subtrees so that the sum of gray-level variations over all partitioned subtrees is minimized under the constraint that two adjacent partitioned subtrees have significantly different average gray-levels (and also that the smallest subtree has at least a specified number of vertices). But it is computationally intractable to directly implement this strategy since it involves explicitly calculating averages of partitioned subtrees. To overcome this, we have used the following approximation method. Instead of calculating averages explicitly, we divide the $\mathcal{K}$ gray levels into $\mathcal{I}$ equally-sized intervals, and use the centers $\{c_1, ..., c_\mathcal{I}\}$ of those intervals to approximate the averages[2].

Now we give the definition of the tree partitioning problem. For a given tree T and an attached gray level $\mathcal{G}(v) \in [0, \mathcal{K}]$ for each vertex v of T, we want to find a partition $T = \bigcup_i T_i$ and a mapping g from T's subtrees to $\{c_1, ..., c_\mathcal{I}\}$ (called g-mapping), so the following function is minimized[3]:

$$\begin{array}{lll} \text{minimize} & \sum_i \sum_j (g(T_i) - \mathcal{G}(x_i^j))^2 & \\ \text{subject to:} & (1) \quad \|T_i\| \geq L, \text{ for each } T_i, & (P) \\ & (2) \quad |g(T_i) - g(T_{i'})| \geq D, \text{ for} & \\ & \text{all adjacent subtrees } T_i \text{ and } T_{i'}. & \end{array}$$

where x_i^j is the j^{th} vertex of T_i, and L and D are two (application-dependent) parameters, both of which are positive numbers.

A dynamic programming algorithm is developed to solve this optimization problem. The algorithm first converts the given tree into a *rooted* tree by selecting an arbitrary vertex as the root. Hence the *parent-children* relation is defined. For each tree vertex, the algorithm constructs a minimum solution (to (P)) on the subtree rooted at the vertex based on the minimum solutions to the subtrees rooted at its children. It extends a partial solution in such a bottom-up fashion and stops when it reaches the root.

[2]Note that the larger $\mathcal{I}$ is, the more accurate the approximation and the longer time it takes to compute.

[3]Our algorithm is general enough to deal with any objective function in the form of $\sum_i \sum_j F(g(T_i), \mathcal{G}(x_i^j))$, where $F()$ is a function computable in $O(1)$ time.

We assume that the vertices of T are labeled consecutively from 1 to $\|T\|$ with the tree root labeled as 1. We use T^i to denote the subtree rooted at vertex i. Note the difference between T^i and T_i, the latter denoting one of the partitioned subtrees. Let $score(i,k,g)$ denote the minimum value of the objective function of (P) among all possible partitions and all possible g-mappings on subtree T^i which satisfy both constraints (1) and (2) except that the partitioned subtree containing i has at least k vertices and is mapped to a fixed g, where $k \in [0, L]$ and $g \in \{c_1, ..., c_{\mathcal{I}}\}$. If we define $scores()$ to be $+\infty$ for all undefined triples (i,k,g) (note $scores()$ is not defined everywhere by the above definition) we have the following lemma, which essentially describes how an optimum partition on a subtree rooted at a vertex is related to the optimum partitions on subtrees rooted at this vertex's children.

Lemma 2 *(a) If $i_1, i_2, ..., i_n$ are the children of vertex i, $n \leq 8$ and $1 \leq k \leq L$, we have*

$$score(i,k,g) = \min \sum_{j=1}^{n} score(i_j, k_j, g) + (g - \mathcal{G}(i))^2,$$
$$\text{for } k = \sum_{j=1}^{n} k_j, k_j \geq 0, \quad \text{when } \|T^i\| \geq L$$
$$scores(i,k,g) = \begin{cases} \sum_{p \in \mathcal{D}(i)} (g - \mathcal{G}(p))^2, & \|T^i\| = k \\ +\infty, & \|T^i\| \neq k \end{cases}$$
$$\text{when } \|T^i\| < L$$

where $\mathcal{D}(i)$ is the set of all i's descendants, i is defined to be $\in \mathcal{D}(i)$ and $score(i_j, 0, g)$ is defined to be

$$\min_{|g'-g| \geq D} score(i_j, L, g').$$

(b) $\min_g score(1, L, g)$ is a minimum solution of (P), where 1 represents the tree root. □

Based on Lemma 2, we can solve the optimization problem (P) by calculating $score()$ for each tree vertex in a bottom-up fashion using the recurrence from Lemma 2(a), and stopping at the tree root.

The pseudo-code to calculate $score()$'s is given below. The input to the algorithm is the minimum spanning tree T obtained by Kruskal's algorithm. To efficiently calculate the recurrence of Lemma 2(a), we need to introduce an auxiliary quantity $score_{temp}(i,k,g)$ for each (i,k,g).

Procedure *cal_scores*(i)
1. **if** vertex i has at least L descendants **then**
2. **begin**
3. **for** each of the n children, i_j, of i **do call** *cal_scores*(i_j);
4. set $score_{temp}(i,k,g) \leftarrow 0$, for all $k \in [0, L]$, and $g \in \{c_1, ..., c_{\mathcal{I}}\}$;
5. **for** $j = 1$ **step** $+1$ **until** n **do**
6. **for** $g = 1$ **step** $+1$ **until** $\mathcal{I}$ **do**
7. **begin**
8. **for** $k = 1$ **step** $+1$ **until** L **do**
9. $score_{temp}(i,k,g) \leftarrow \min\{score_{temp}(i,k_1,g) + score(i_j,k_2,g)\}$, for $k = k_1 + k_2, k_1, k_2 \geq 0$;
10. **for** $k = 1$ **step** $+1$ **until** L **do**
11. $score(i,k,g) \leftarrow \min_{j \geq k}\{score_{temp}(i,j,g) + (g - \mathcal{G}(i))^2\}$;
12. **for** $g = 1$ **step** $+1$ **until** $\mathcal{I}$ **do**
13. set $score(i,0,g) \leftarrow \min score(i,L,g')$, for $|g' - g| \geq D, g' \in \{c_1, ..., c_{\mathcal{I}}\}$.
14. **end**
15. **end**
16. **else**
17. **begin**
18. **for** $g = 1$ **step** $+1$ **until** $\mathcal{I}$ **do**
19. **for** $k = 1$ **step** $+1$ **until** L **do**
20. **if** $k = \|T^i\|$ **then** set $score(i,k,g) \leftarrow \sum_{p \in \mathcal{D}(i)} (g - \mathcal{G}(p))^2$;
21. **else** set $score(i,k,g) \leftarrow +\infty$.
22. **end**

Theorem 1 summarizes the computational efficiency of the tree partitioning algorithm. To get the tree partition corresponding to $\min_g score(1, L, g)$, some simple bookkeeping needs to be done while calculating $score()$'s. This can be done within the asymptotic time and space bounds of Theorem 1.

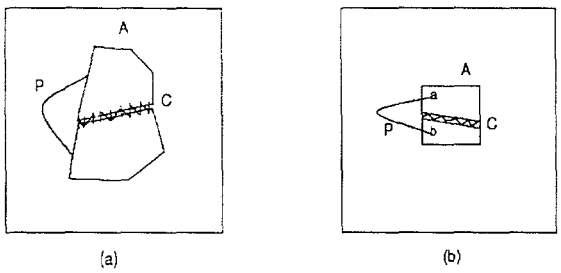
Theorem 1 *$\min_g score(1, L, g)$ can be correctly calculated by the above algorithm in $O(\max\{(\|T\| - L), 1\}\mathcal{I}(\log(\mathcal{I}) + L^2))$ time and in $O(\|T\|L\mathcal{I})$ space.* □

For many practical applications, $\mathcal{I}$ is a small constant. Hence the computational resources used by the algorithm become $O(\max\{(\|T\| - L), 1\}L^2\})$ in time and $O(\|T\|L)$ in space.

Segmentation on Noisy Images

This section evaluates how noise affects the performance of our image segmentation algorithm. The goals are two-fold: (1) to study how well the segmentation algorithm performs on real-life images, which can be considered as ideal images (each object has a uniform gray-level) with variations on each pixel's gray level, and (2) to understand why different types of noise affect the performance of the algorithm differently as we have observed in our tests. Potentially noise affects the algorithm's performance in both stages of the algorithm: spanning tree construction and tree partitioning. We will show that noise has greater effects on the performance in the tree partitioning stage than in the spanning tree construction stage. In this study, we consider two types of noise: transmission errors and Gaussian additive noise. Studies on other types of noise are under way.

The model for generating *transmission errors* is defined as follows: each pixel of the image has a probability $\mathcal{P}$ to keep its original gray level during transmission and the probability $1 - \mathcal{P}$ to randomly change to another gray level $\in [0, \mathcal{K}]$. *Gaussian additive noise* adds



(a) (b)

Figure 1: A schematic of an image, an object A in the image, a cutset C and a by-pass P.

to each pixel independently a random normal value (using the floor function for real-to-integer conversion) according to a normal distribution $N(0, \sigma^2)$ censored to $[-\mathcal{K}/2, \mathcal{K}/2]$.

Spanning tree construction on noisy images

One basic premise for our image segmentation algorithm to be effective is that each object, given as a homogeneous region in an image, is represented as one subtree of the spanning tree representation. Lemma 1 has shown that this is true in the ideal situation. In the following, we show how noise affects this property.

Consider an object A in a given image I. Let $G(I)$ be the planar graph representation of I (as defined in Section 2.1) and T be its minimum spanning tree obtained by Kruskal's algorithm. We use $G(A)$ to denote the subgraph of $G(I)$ induced by the pixels of A. An edge set C of $G(A)$ is called a *cutset* if deleting C divides $G(A)$ into two unconnected parts. Theorem 2 generalizes Lemma 1 on noisy images.

Theorem 2 *In the presence of possible noise, A is represented by more than one subtree[4] in T if and only if there exist a cutset C of $G(A)$ and a path P that has its two end vertices on two sides of the cut of $G(A)$ and has its remaining vertices outside of $G(A)$ such that every edge of P has smaller[5] weight than every edge of C.* □

Figure 1(a) illustrates the idea of Theorem 2. Note that Theorem 2 is a true generalization of Lemma 1 since the if-and-only-if condition can never be true in a situation of no noise, and hence A is always represented as one subtree of T in the ideal situation.

To estimate how probable the if-and-only-if condition in Theorem 2 is we have conducted the following computer simulation. The experiment is done on a 256-gray-level image I having one object A in the center of the image (see Figure 1(b)). I is a 256×256 image and A is a 30×30 square. The background has a uniform gray level 100 and A has a uniform gray level 150. We add transmission errors and Gaussian additive noise, respectively, to I as follows. When adding transmission errors, each pixel of I has a probability 0.3 to keep its original gray level and the probability 0.7 to randomly and uniformly change to another gray level $\in [0, 255]$. When adding Gaussian additive noise, each pixel of I is added by a value $[\delta + 1/2]$ (modulo 256), where δ is random number generated according to the normal distribution $N(0, \sigma^2)$ censored to [-128, 128] with $\sigma = 40$.

For each type of noise, we estimated the probability that there exist a path P connecting two pixels a and b, and a cutset C of A separating a and b (see Figure 1(b)) such that every edge of P has smaller weight than every edge of C, where a and b are two randomly chosen pixels both of which are 5-pixels from the left boundary of A and are at least 5 pixels from the lower and upper boundaries of A[6], and P has at least 20 edges.

We have observed, for this particular experiment, that the probability that there exist such a P and a cutset C is very small ($< 10^{-3}$), for both types of noise. This experiment suggested that both types of noise have very little effect on the property that a homogeneous region corresponds to one subtree of the minimum spanning tree constructed by Kruskal's algorithm.

Tree partitioning on noisy images

Though both types of noise have little effect on the property that a homogeneous region corresponds to one subtree of the spanning tree representation they could affect the tree partitioning result in a form we call *corrosions*. Consider an object A in a given image and its representing subtree T_A. With noise, T_A may contain a subtree (or subtrees) that has a (significantly) different average gray level than the rest of T_A, and contains more than enough vertices ($\geq L$. See the definition of tree-partitioning problem (P)) to be partitioned into a separate region (see Figure 2). This subsection presents a comparative study on how the two types of noise affect the formation of corrosions.

Let $g(A)$ be the (uniform) gray level of A before noise is added. We compare the probabilities, P_1 and P_2, that a connected region A' of A will have its gray level changed to the same value $g(A)+k$, for any $k \neq 0$, when transmission errors and Gaussian additive noise are added, respectively. Let p_k denote the probability that one pixel of A' changes its gray level from $g(A)$ to $g(A) + k$ when Gaussian additive noise is added. For the simplicity of discussion, we assume that $g(A) = 0$,

[4] Note that pixels of A corresponds to one subtree of T if and only if for any pair of pixels of A, the path between them in T only contains pixels of A.

[5] We ignore the case of equality for the simplicity of discussion.

[6] The reason for choosing a and b inside A instead of on the boundary of A is to avoid errors caused boundary "corrosions".

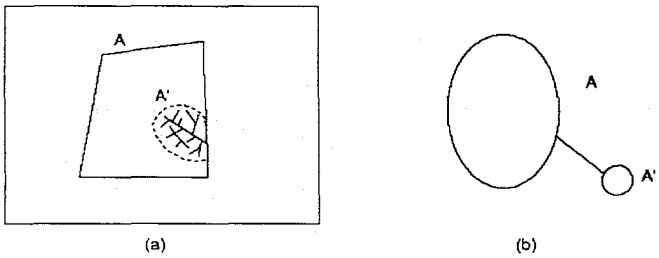


Figure 2: A schematic of "corrosion". (a) Object A in an image with its "corroded" subregion A'. (b) A's subtree representation and A''s subtree representation.

hence $k \in [1, \mathcal{K}]$. We define $p_0 = 1 - \sum_{k=1}^{\mathcal{K}} p_k$. Recall $\mathcal{P}$ denotes the probability that a pixel keeps its original gray level in the presence of transmission errors. It can be shown by a simple calculation that

$$P_1 = \left(\frac{1-\mathcal{P}}{\mathcal{K}-1}\right)^n (\mathcal{K}-1), \qquad \text{and} \qquad P_2 = \sum_{k=1}^{\mathcal{K}} p_k^n,$$

where $n = \|A'\|$ (note that $P_2 = \sum_{k=1}^{\mathcal{K}} p_k^n$ is true for any type of noise). Theorem 3 shows the relationship between P_1 and P_2, which can be proved using Jensen's inequality [2] (page 433).

Theorem 3 *For any $\mathcal{N} \in [2, \mathcal{K}]$ and $n > 0$, when $\sum_{k=0}^{\mathcal{N}} p_k = 1$ and $p_0 = \mathcal{P}$,*

$$\sum_{k=1}^{\mathcal{N}} p_k^n \geq \left(\frac{1-\mathcal{P}}{\mathcal{N}-1}\right)^n (\mathcal{N}-1).$$

□

Theorem 3 implies that transmission errors are the least possible to form corrosions among all possible forms of noises (including Gaussian additive noise) when $\mathcal{P}$ or p_0 is fixed. Computer simulations have shown that when the noise levels generated by the two noise models are visually "similar",

$$\sum_{k=1}^{\mathcal{N}} p_k^n >> \left(\frac{1-\mathcal{P}}{\mathcal{N}-1}\right)^n (\mathcal{N}-1).$$

Test results on real images confirm this inequality.

Computer simulation results have shown that the probability of keeping a pixel's original gray level is the dominating factor in the formation of corrosions in the presence of any type of noise. Note that in the case of Gaussian additive noise,

$$p_0(\sigma) = \frac{1}{\sqrt{2\pi}\sigma} \int_{-1/2}^{1/2} e^{-\frac{t^2}{2\sigma^2}} dt,$$

represented explicitly as a function of the standard deviation σ, is a decreasing function of σ. Hence the averaging operation, which decreases σ when done on a small neighborhood (e.g. a 3×3 square around each pixel), will improve the performance of the tree partitioning algorithm when having Gaussian additive noise.

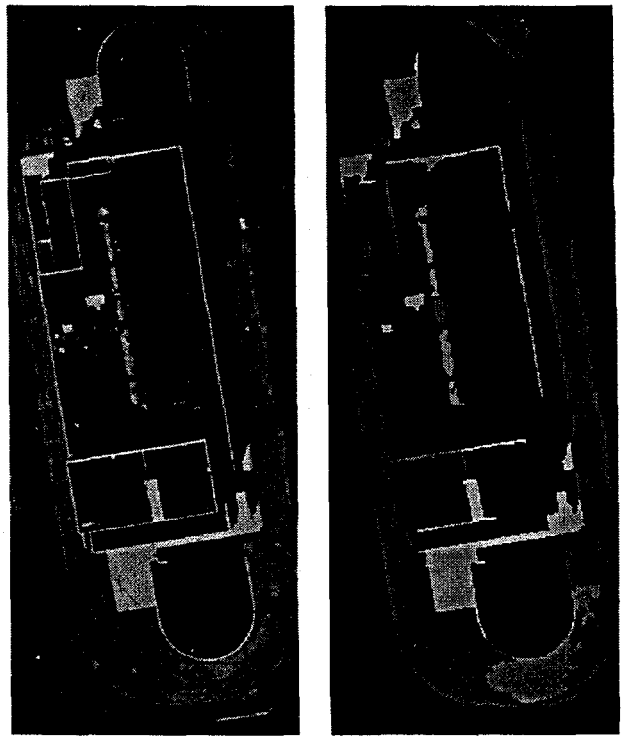


Figure 3: An aerial image of 202x503 pixels and its segmentation result.

The averaging operation does just the opposite when having transmission errors as test results have shown. The main reason for this behavior is that averaging decreases $\mathcal{P}$ and hence increases the probability

$$\left(\frac{1-\mathcal{P}}{\mathcal{K}-1}\right)^n (\mathcal{K}-1)$$

of forming corrosions.

The following gives three examples of the segmentation results of our image segmentation algorithm. Segmentations are done on a SPARC-20 workstation, and none of the three images take more than 1.5 CPU minutes. Figure 3 is an aerial image along with its segmentation. Figure 4 gives the segmentation result on an image with transmission errors added, and Figure 5 shows the segmentation result on the same image with Gaussian additive noise added. Corrosions can be seen from the segmentation results of Figures 4 and 5.

Summary

We have developed an efficient and effective computer algorithm for 2-D image segmentation. Segmentations are done by essentially dividing an image into regions so that the sum of gray-level variations over all divided

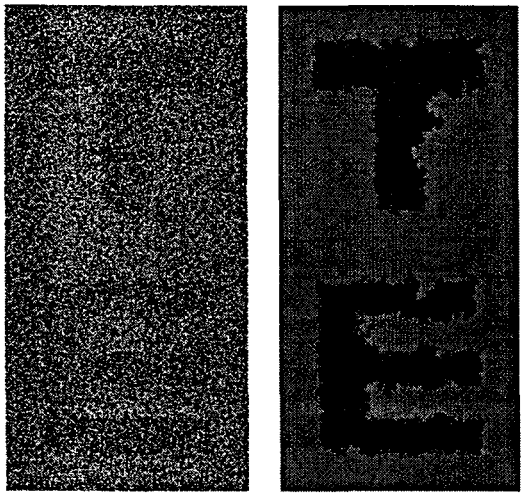


Figure 4: An image of letters "T" and "E" with transmission errors added and its segmentation result. Each pixel has a probability 0.3 to keep its original value and the probability 0.7 to randomly change to another gray level $\in [0, 255]$. The image has 256×256 pixels.

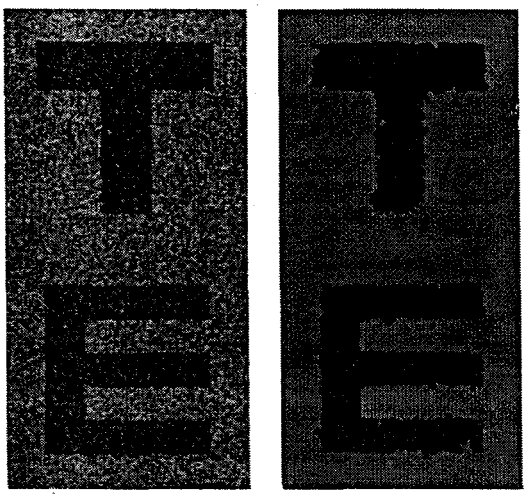


Figure 5: An image of letters "T" and "E" with Gaussian additive noise with $\sigma = 40$, and its segmentation result. The image has 256×512 pixels and consists of 256 gray levels.

regions is minimized under the constraint that adjacent regions have significantly different "average" gray levels. The computational efficiency is achieved based on three key ideas: (1) a tree representation of an image which keeps the topological relationships among objects in the image, (2) an approximation method to avoid calculating averages explicitly, and (3) recurrence relationships among partial optimum solutions defined on a tree. Both computer simulations and tests on real images have shown that the algorithm is robust even on very noisy images, which is further confirmed by our mathematical analysis.

Acknowledgements

This research was supported by the United States Department of Energy, under contract DE-AC05-84OR21400 with Lockheed Martin Energy Systems, Inc.

The authors would like to thank Dr. Reinhold C. Mann for many helpful discussions related to the work presented in this paper.

References

[1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman (1974), *The Design and Analysis of Computer Algorithms*, Readings, MA, Addison-Wesley Publishing Company.

[2] P. J. Bickel and K. A. Doksum (1977), *Mathematical Statistics: Basic Ideas and Selected Topics*, Holden-Day Inc.

[3] R. O. Duda and P. E. Hart (1973), *Pattern Classification and Scene Analysis*, New York, John Wiley and Sons.

[4] R. C. Gonzalez and P. Wintz (1987), *Digital Image Processing (second edition)*, Readings, MA, Addison-Wesley Publishing Company.

[5] R. M. Haralick and L. G. Shapiro (1985), "Image Segmentation Techniques", *Computer Vision, Graphics, and Image Processing*, Vol. 29, pp. 100 - 132.

[6] J. B. Jr. Kruskal (1956), "On the shortest spanning subtree of a graph and the traveling salesman problem", *Proc. Amer. Math Soc*, Vol. 7, No. 1, pp. 48 - 50.

[7] N. Pal and S. Pal (1993), "A review on image segmentation techniques", *Pattern Recognition*, Vol. 26, No. 9, pp. 1277 - 1294.

[8] E. C. Uberbacher, Y. Xu, R. W. Lee, C. W. Glover, M. Beckerman, and R. C. Mann (1995), "visGRAIL – Image Exploitation Using Multi-sensor/Neural Network Systems", *Lockheed Martin Technical Report K/NSP-338-95.*

[9] Y. Xu and E. C. Uberbacher (1996), "2-D Image Segmentation Using Minimum Spanning Trees", *Image and Vision Computing*, in press.