RC 21879 (98446) 7 November 2000 **Computer Science**

IBM Research Report

Semi-Automatic Segmentation Using Simulated Annealing

Lisa M. Brown
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - T. J. Watson - Tokyo - Zurich

Semi-Automatic Segmentation Using Simulated Annealing

Lisa M. Brown
I.B.M. T.J. Watson Research Center
Hawthorne, NY 10532
lisabr@us.ibm.com

Abstract

We describe an interactive color segmentation scheme based on simulated annealing. This method allows a user to very quickly identify an arbitrary but particular region and returns the region more accurately segmented and appropriately measured. This technique could be used in a range of applications such as image sub-selection in graphic arts, tumor identification in medicine, or as an input mechanism for training samples in industrial tracking. We have implemented this technique in an educational software tool that allows students to take measurements on images and videos. Using the region identification routine, students can compare the land areas of the continents, the growth of the ozone hole, or the size of an enormous iceberg that recently broke off an ice shelf in Antarctica.

Introduction

The method developed here arose from a need to improve semi-automatic segmentation for an educational software tool called Visual Venture. Visual Venture was designed to allow middle school students the ability to take measurements on images and videos[1]. In addition to measuring the length, coordinates and angles of two-dimensional features in arbitrary images, the software allows a student to manually or semi-automatically measure the area of 2D features such as the area of color-coded map regions or geometric patterns. The problem faced was that the segmentation technique needed to be general purpose. Visual Venture was explicitly created to allow curriculum specialists, teachers, and even students, to use images/videos from the Internet. After the software is initialized with a measurement of the scale of a given image, the student can manually draw around a region of interest to measure its perimeter or area. In addition, students can manually specify the region of interest, but allow the software to use the image information to improve the accuracy of this measurement. By utilizing an initial estimate of the region, we could hopefully identify the region of interest, even though, the wide range of content area and image types would clearly challenge the system.

Initially, an interactive segmentation routine was developed using intelligent color histogram segmentation followed by connected components. However, this approach was limited in two fundamental ways. Although the initial region specified by the user, was segmented and exploited by the histogram computation, it did not spatially constrain the final segmentation, which was performed using thresholding. Secondly, the connected component selected to most nearly represent the region of interest could be much larger or smaller than the desired region.

A simulated annealing approach has several advantages:

(1) It can fully exploit the initial estimate information.

(2) The approach allowed us to design a global optimization routine for a range of objective functions. This is a significant advantage since a realistic part of the problem we face is to determine the ideal objective function.

(3) A smoothness constraint is not required. For our application, the value of such a constraint is

not known *a priori*, may vary along the border, and is part of the problem we are trying to solve.
(4) Each update is based only on a local computation.
(5) The objective function can incorporate the range of image information available to us: distance from the initial boundary, boundary-based, region-based, and pixel-based metrics.

Related Work

Unlike the broad class of segmentation methods which perform global segmentation, the method described here identifies a single connected region based on a preliminary region supplied by the user. Global segmentation or region labeling methods typically involve the segmentation of an entire image into distinct regions such that every pixel in the original image belongs to a connected region in the segmented image. See Pal et.al. [2] for a survey of image segmentation methods. In the general case of global segmentation, there is little or no reliance on input from the user. The technique described here differs from these methods in that only a single connected region is identified and this region is selected based on the information supplied by the user.

A related area of investigation in computer vision is the development of methods, which determine a contour or boundary from an image given a set of input points. The pioneer work in this area is based on a variational approach in which a cost function composed of image and contour forces are minimized [3]. This work is often referred to as the "active contour" model or more graphically, as "snakes." The image forces are typically based on gradient information and the contour forces are computed from constraints such as smoothness and elasticity. These methods have several inherent differences from our approach. First of all, the problem to be solved is somewhat different. In the active contour model, the input is a number of fairly accurate points and the desired result is a contour fitted to the image passing near these points. The objective is to find the contour rather than a closed region that best fits the user's input. Exceptions include the work of [4, 5]. In [4], global segmentation is performed. In [5] region descriptions based on clustering are used.

In the standard contour model, region information is not used. The resulting curve may follow maximal gradients that lie along different regional boundaries. Secondly, the input and output of the active contour is based upon a fixed number of points. Therefore, the accuracy or resolution of the result, and the associated speed of the computation, is directly dependent on the number of points. Related to this issue, is the use of constraints on the smoothness or elasticity of the curve. In our case, we are assuming the user is not drawing the contour at the desired smoothness, and this smoothness is not known *a priori*. Rather, we want to determine this contour based on image information, particularly, the information about the enclosed region of interest and the proximity to the original input.

Lastly, there are some noteworthy computational aspects of the contour model that have lead us to use a different global optimization technique. In the variational approach, energy values result from integrating along the entire contour; every part of the solution is dependent on the entire configuration. This becomes an expensive procedure that employs matrix computations that are more cumbersome as the complexity of the energy function increases. Since a configuration change in our 'Metropolis' version of the simulated annealing process is a local computation, the evaluation of each value of the objective function is very fast.

The use of simulated annealing as an optimization process in image processing and computer vision is widespread. Several authors have developed edge detection [6] and global image segmentation methods [7] using simulated annealing. These methods differ from our approach since they are again designed to solve a significantly different objective. Lundervold et. al. [8] has developed a methodology to match medical image information to an atlas. In this work, models of specific *a priori* information are used to solve the task. For example, the number of tissues to identify is known, as is, some of the statistical differences in tissue types.

The optimization may also be implemented using a genetic algorithm [9]. Research performed by [10] uses a genetic approach to semi-automatically segment medical images. In their work, the user supplies two (or more) 2D training samples that are used to segment the region of interest in 3D. Although their approach is similar to ours, their objective is to use the 2D information to derive the 3D region. One sample is used as seed information and the others are used to evaluate the fitness function. The fitness function does not use region information; it is based on connectivity, edge similarity and proximity. New edge points are found using a

genetically evolving detector. Unlike our algorithm, their method evaluates points that do not define a connected boundary and requires a later stage to perform interpolation and contour extraction. For the latter, they propose a differential elastic model.

Method

Our region identification method is based on a modified Metropolis Algorithm. Starting with our initial estimate of the boundary, we determine which pixels are inside and outside of the initial estimate of the region. Our configuration space is the set of all possible boundaries, which enclose a single eight-connected region. To generate a change in the configuration space, we can either add or delete a single pixel or a single "layer" of pixels. A change is selected at random and the displacement it causes in the objective function is measured. Using the Boltzmann probability distribution and a simple annealing schedule, we test if the current configuration change should be accepted. If the change is accepted, we update the configuration. The procedure is repeated until the change in the objective function is smaller than a critical threshold for a given number of configuration changes.

Given an arbitrary initial contour, composed of an array of mouse positions, we compute which pixels are inside the region using the following algorithm. Adding the first point to the end of the array closes the contour. The array is interpolated so that every row in the image is traversed as we go from one point to the next. The array is then filtered so that all consecutive points, which are on the same row, are removed. In addition, all points that are local vertical peaks are removed. The points are then reordered such that for each row there is an array of points that lie along this row. The points along each row are ordered from left to right. Finally, the original image is segmented to identify a single connected region based on the user's specified points, by traversing each row list using a scanline filling process as follows. All rows without points are not included in the final region. For each row list, all points preceding the first point in the list, are not included in the final region. Starting at the first point, all points are included until the next point is reached. All points after the second point are excluded until the next point is reached. We continue by alternating inclusion until the row is entirely traversed. We call this process *scan-line filling*. The final region is then specified by a binary image in which each point that is included is marked on, while each point not included is not marked off. We claim that the region identified by this process is a single 8-connected region, which may have holes and is arbitrarily complex.

Once the initial region is identified, a color histogram of this region is computed. We compute our histogram in a similar manner to [11] with some modifications based on [12]. We convert our color image to 3 bytes in Hue/Saturation/Intensity (HSI) space and compute the histogram using a 5x5 array for hue/saturation and a 5x1 array for intensity. For each pixel, we determine whether it is chromatic based on the psychophysics of human color perception. These results basically say humans have trouble seeing color if the intensity is too low or too high, and as the intensity increases, humans require decreasing amounts of saturation in order to differentiate hue. If it is chromatic, it is added to the hue/saturation histogram; otherwise it is added to the intensity histogram. We also increment each histogram value in hue/saturation space using a 3x3 Gaussian filter, and in intensity space using a 3x1 Gaussian filter.

Before annealing, we also compute a distance map, i.e., the distance of each pixel in the image from the nearest initial boundary point. We perform this using a chamfer approximation since accuracy here is not paramount [13]. Lastly, we compute the initial value of the gradient across the boundary for each point along the border. Using a 3x3 window, we compute the direction of the gradient at each border point, based on the average direction of pixels outside the region. The gradient value is then computed as the magnitude of the absolute color difference between the border pixel and the exterior pixel most nearly located in the direction of the gradient.

To apply the simulated annealing, we must ensure that we can randomly select an option from the set of all possible configurations. Therefore we need to keep track of the points along the border. A list of these points is maintained and called the *delete list.* . The number of points that can be deleted from a particular configuration is referred to as $\mathbf{N^d}$. We also need to keep track of the points adjacent to the border but not in the region, which we refer to as the *add list*. The number of points on this list is $\mathbf{N^a}$. The configuration change generator

will select a point to add if a random number between 0 and 1 is greater than $N^a / (N^a + N^d)$. If a pixel is added, pixels in its 3x3 neighborhood are evaluated in order to update the *add* and *delete* lists. When a pixel is added,
(1) it is added to the delete list, removed from the add list,
(2) pixels are tested to see if they are no longer are on the border and should be removed from the delete list, and
(3) pixels are tested to see if they are now adjacent to the border, not in the region, not already in the add list - these pixels are added to the add list.
Similarly, when a pixel is deleted,
(1) it is removed from the delete list, added to the add list,
(2) pixels are tested to see if they are now on the border and should be added to the delete list,
(3) pixels are tested to see if they are on the add list but they are no longer adjacent to the border, these pixels are removed from the add list.

However, since the boundary is required to represent a single connected region, we must ensure that deleting a point will not disconnect the region. We do this by not allowing the removal of a point P which disconnects two pixels in the 3x3 (or larger) window centered at P. We call this constraint, *deletability*. A point is *deletable* if and only if removing it will not disconnect two points in the window centered at this point. Similarly, we do not allow a point Q to be added to region, if, it is not possible to delete this point on the next iteration. In this way, all changes to the configuration are still equally likely. This is an important condition that must be satisfied so that the search space is explored appropriately by the simulated annealing.

Furthermore, in order to ensure convergence, assuming adequately slow cooling, it is sufficient, but not necessary, to satisfy the requirement that the system is *locally balanced* [14]. This requirement can be satisfied by ensuring that,

$$P_f(A,B) \; / \; P_f(B,A) = \exp(E_A - E_B)/T \quad [1]$$

where $P_f(A,B)$ is the probability of generating and accepting a change from configuration A to configuration B. $P_f(B,A)$ is the probability of performing the reverse update, E_A , E_B are the values of the objective function for state A and state B respectively, and **T** is the temperature.

In physics, this is known as the *detailed balance* equation. More simply, we need to compensate for the fact that the proposed transitions are not unbiased, viz., the probability of generating a change from configuration state A to configuration state B is not equivalent to the probability of generating a change from configuration state B to configuration state A. If $P_i(A,B)$ is the probability of generating a *proposed* change from A to B and we propose to add a point with this probability, then if it is accepted, the reverse probability that we propose to delete this point, $P_i(B,A)$, will not necessarily be equivalent to $P_i(A,B)$. This is because the total number of points $(N^a + N^d)$ changes after every update. To correct for this phenomenon, we define,

$$r_A = T \ln (P_i(A,B))$$
$$r_B = T \ln (P_i(B,A)). \quad [2]$$

We now define a modified Metropolis acceptance rule as follows:

if $(E_A - r_A) > (E_B - r_B)$, accept
if $(E_A - r_A) <= (E_B - r_B)$, accept with probability $\exp(((E_A - r_A) - (E_B - r_B)) / T)$. [3]

We refer to this rule as $P_a(A,B)$, or the probability of accepting a configuration change from state A to state B. Since $P_f(A,B) = P_i(A,B) * P_a(A,B)$, equation (1) can be rewritten as,

$$(P_i(A,B) * P_a(A,B)) / (P_i(B,A) * P_a(B,A)) = \exp(E_A - E_B)/T.$$

Using the terms r_A, r_B defined in (2),

$$P_i(A,B)/P_i(B,A) = \exp((r_A - r_B) / T),$$

and by the acceptance rule (3),

$$P_a(A,B)/P_a(B,A) = \exp(((E_A - r_A) - (E_B - r_B)) / T),$$

we see that our modified rule satisfies equation (1). Therefore, if we cool down slowly enough, the algorithm will converge to an optimal solution.

For our system, the probability of a proposed update, where an update is a single pixel added or deleted from the boundary, is $P_i(A,B) = 1/(N^a_A + N^d_a)$ where the subscript denotes the count for configuration A. Similarly, $P_i(B,A) = 1/(N^a_B + N^d_B)$ refers to the probability for an update from B to A. Therefore,

$$r_A = - T \ln(N^a_A + N^d_a)$$

$$r_B = - T \ln(N^a_B + N^d_B)$$

and the modified Metropolis accept/reject step is well-defined and straightforward to implement. For a particular configuration change from A to B, r_A is computed from the counts prior to the change and r_B is computed from the counts after the change. The neighboring pixels that are changed from lying on the border (delete points) or adjacent but outside of the border (add points) are already determined in order to measure the change in the objective function.

The objective function **E**, that we wish to minimize, is composed of a linear sum of weighted terms:

$$E = w_1 E_A + w_2 E_D + w_3 E_H + w_4 E_G + w_5 E_S$$

where $w_1, w_2 .., w_5$ are the weights, and E_A, E_D, E_H, E_G, and E_S are the area, distance map, region histogram, gradient and gradient similarity energy terms, respectively The area term reflects how the area of the region given by the current configuration differs from the original. The distance map term reflects the overall distance that the current boundary has deflected from the original boundary. For each pixel on the boundary, we compute the weighted cost of its absolute distance from the original boundary using the distance map computed earlier. We make no penalty for small distances and penalize heavily as a point moves very far way. The distance map term is the sum of the distances for each point on the boundary.

The region histogram energy term, E_H, reflects the degree to which the pixels included in the region of the current configuration improve or disturb the integrity of the original histogram. If a pixel is added to the original region, we assess its likelihood of belonging to the region specified by the user. Its probability is greatest if it falls on, or near, a color value with a large histogram value. Similarly, if it falls far from any color value, which is represented in the original histogram, its probability is low. If it falls on a color value only sparsely represented, it is neutral. In this way, nearby pixels of the same or similar color to the original region are good candidates to add, pixels of questionable colors are added or not based on other criteria.

The fourth energy term, E_G, adds a cost based on the extent to which the boundary points lie along a strong gradient. For each point on the boundary, we measure the gradient in the direction normal to the boundary. The gradient term is the inverse of the sum of the gradients for all points on the boundary. The gradient may be computed as the magnitude of the color or intensity difference or it may be computed from the difference in the probabilistic likelihood between the pixels on each side of the boundary. By using the latter metric, particular differences in color changes are not differentiated. This is useful, since the true boundary is not necessarily the strongest color change, but the local consistent one.

The last term is the gradient similarity term, E_S. This term is added to decrease the likelihood that the contour moves from one boundary to the boundary of a different region. For this term we compute the sum of the differences in the gradient direction along contiguous boundary points.

The objective function is computed by evaluating the change in the energy value based on the pixel/layer that was added or deleted from the configuration change generator at each iteration. To compute the energy change, which is region, based (E_A or E_H), we simply measure the change introduced by the added or deleted pixel. On the other hand, to measure the change resulting from boundary-based metrics (E_D, E_G, E_S), we need to consider how the boundary has changed, i.e., which points have been added or deleted from the boundary. This can be done in conjunction with the bookkeeping already performed to update the add and delete lists.

Results

We have implemented this method and integrated it into Visual Venture, to be used by students to measure the perimeter and area of regions in images. In particular, the method is used to measure the area of regions of a map or geometric patterns. Several maps related to middle school science curricula were downloaded from the Internet for testing. Three examples are shown in Figures 1-3. In each image the final segmentation is shown outlined and also marked by an interior grid.

Although it is difficult to discern the color segmentation clearly in these gray-level images, several advantages and disadvantages of the method can be noted. Because the method is constrained by the distance map, it will never deviate radically. Secondly, the problem of interior colors only sparsely represented, such as interior text or lines, does not, in general pose a problem since the segmentation only considers displacements from the initial boundary estimate.

Figure 3 is an example for which a connected component or region growing type method will fail. On the other hand, as seen in Figures 2 and 3, where extraneous high gradients or sparsely represented color pixels exist near the boundary, the final boundary will stray erroneously.

In order to measure the accuracy of this approach, we have begun to compare the water and land area measurements of images at given scales with their geographic reference values. The results of this effort are still in progress and will be included in the final version of this paper.

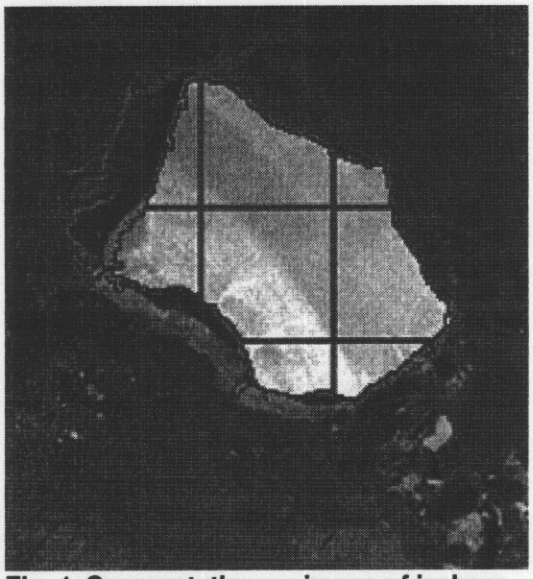

Fig. 1. Segmentation on image of iceberg.

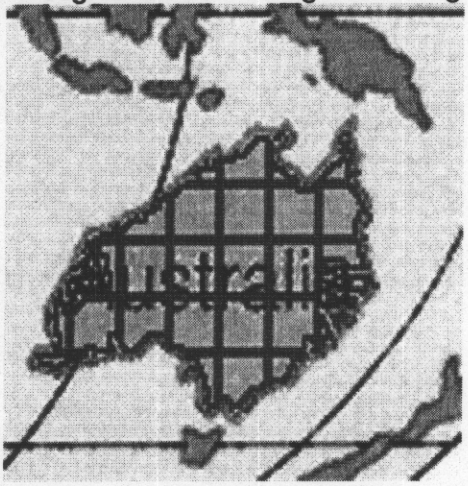

Fig. 2. Segmentation of Australia.

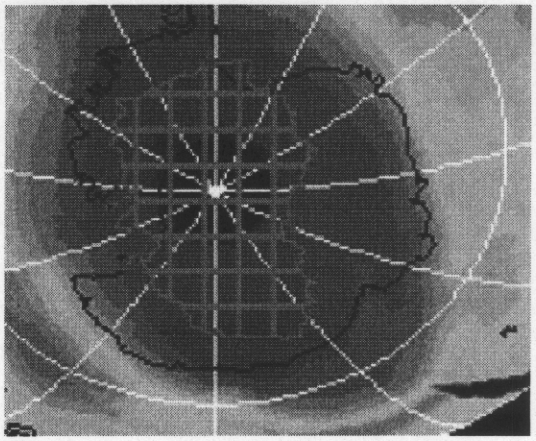

Fig. 3. Segmentation result on image of ozone hole from video sequence.

References

[1] Brown, L.M. and Crayne, S., "Visual Venture: Investigations with Images and Videos," *Int'l Conf. on Mathematics/Science Education and Technology*, Feb 5-8, 2000, San Diego, Ca., pp. 89-94.

[2] Pal, N.R., Pal, S.K., "A Review of Image Segmentation Techniques," *Pattern Recognition*, Vol. 26, No. 9, 1993, pp1277-1294.

[3] Kass, M., Witkin, A., Terzopoulos, D., "Snakes: Active Contour Models," *IEEE Proc. Int'l Conf. Computer Vision*, ICCV87, London 1987, pp259-268.

[4] Zhu, S.C., Yuille, A., "Region Competition: Unifying Snakes, Region Growing, and Bayes/MDL for Multiband Image Segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol.18, No. 9, September, 1996, pp884-900.

[5] Etoh, M., "Active Contour Extraction Based on Region Descriptions Obtained from Clustering," *Systems and Computers in Japan*, Vol. 25, No. 11, 1993, pp1111-1119.

[6] Tan, H.L., Gelfand, S.B., & Delp, E.J., "A Cost Minimization Approach to Edge Detection Using Simulated Annealing," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol.14, No. 1, January 1991, pp3-18.

[7] Bhandarkar, S.M., Zhang, H., "Image Segmentation Using Evolutionary Computation," *IEEE Trans. on Evolutionary Computation*, Vol. 3, No. 1, April 1999, pp1-21.

[8] Lundervold, A., Storvik, G., "Segmentation of Brain Parenchyma and Cerebrospinal Fluid in Multispectral Magnetic Resonance Images," *IEEE Trans. on Medical Imaging*, Vol. 14, No. 2, June 1995, pp339-349.

[9] Gudmunddsson, M. El-Kwae, E.A., Kabuka, M.R., "Edge Detection in Medical Images Using a Genetic Algorithm," *IEEE Trans. on Medical Imaging*, Vol. 17, No. 3, June 1998, pp469-474.

[10] Cagnoni, S., Dobrzeniecki, A.B., Yanch, J.C., & Poli, R., "Interactive Segmentation of Multi-

Dimensional Medical Data with Contour-Based Application of Genetic Algorithms," IEEE 1994.
[11] Kjeldsen, R. and Kender, J., "Finding Skin in Color Images," *Int'l Conf. on Automatic Face and Gesture Recognition*," October 14-16, 1996, pp.312-317.
[12] Tseng D. and Chang C., "Color Segmentation using UCS Perceptual Attributes," *Proc. Natl. Sci. Counc.* ROC(A) Vol 18, No. 3, 1994, pp.305-314.
[13] Barrow, H.,Tenenbaum, J.,Bolles, R., Wolf, H., "Parametric Correspondence and Chamfer matching: Two New Techniques for Image Matching," *IJCAI*, 1977, pp659-663.
[14] van Laarhoven, P., & Aarts, E., Simulated Annealing: Theory and Applications, D. Reidel Publishing Co. Dordrecht, Netherlands, 1987.