CVGIP: IMAGE UNDERSTANDING
Vol. 59, No. 2, March, pp. 171–182, 1994

Extracting Topographic Terrain Features from Elevation Maps

IN SO KWEON

Department of Automation and Design Engineering, KAIST, Seoul, 130-012 Korea

AND

TAKEO KANADE

The Robotics Institute, Carnegie–Mellon University, Pittsburgh, Pennsylvania 15213

Received **July 8.** 1991; accepted September 24, 1993

Some applications such as the autonomous navigation in natural terrain and the automation of map making process require high-level scene descriptions **as** well **as** geometrical representation of the natural terrain environments. In this paper, we present methods for building high level terrain descriptions, referred to **as** topographic maps, by extracting terrain features like "peaks," "pits," "ridges," and "ravines" from the contour map. The resulting topographic map contains the location and type of terrain features **as** well as the ground topography. We present new algorithms for extracting topographic maps consisting of topographic features (peaks, pits, ravines, and ridges) and contour maps. We develop new definitions for those topographic features based on the contour map. We build a contour map from an elevation map and generate the connectivity tree of all regions separated by the contours. We use this connectivity tree, called **a *topographic change tree,*** to extract the topographic features. Experimental results on a digital elevation model (DEM) supports our definitions for topographic features and the approach.

1. INTRODUCTION

Extracting topographic features from elevation maps has traditionally been studied in two research areas: in robotics, autonomous underwater vehicle **(AUV)** route planning and self-localization, [11] and in cartography, the automation of the map making process **[6,** 10, 14].

Recently, Seemuller [14] presented a method for producing drainage networks from terrain elevation data. In his method, he determined drainage points by finding grids located at a local minimum in elevation for the horizontal or vertical direction in a **3 x 3** neighborhood. He then traced and linked drainage points into a list of drainage nets. **As** he described in his paper, it suffers with the locality of the method. Moreover, he did not justify why a local minimum in elevation only for the horizontal or vertical direction can be a drainage point.

There also has been a variety of techniques to detect pits, peaks, ridges, and ravines as a means of characterizing topographic structures in a digital image. Many of the methods, however, use various heuristics without rigorous mathematical justification [7].

Haralick ***et al.*** [5], Ponce [3], and Besl [2] defined and extracted topographic features using concepts from differential geometry. There exists a problem: If we take a peak and then rotate it about a horizontal axis, it soon stops being a peak, even if curvature, being intrinsic to the shape, does not change. In this paper, we will present a novel method to overcome those problems found in previous research works.

We first examine the definitions of topographic features. We then present our basic representation, the contour tree, from which topographic features will be extracted. In Section **5,** we describe an algorithm for extracting peaks and pits from the contour tree. This is followed by the extraction of ridges and ravines from the contour tree in Section **6.** Last, we present experimental results from application of the algorithms to a DEM.

2. DEFINITIONS OF TOPOGRAPHIC FEATURES

Many topographic feature definitions are ambiguous. Traditional natural language definitions of topographic features have the substantial drawback that such definitions either use terms which are not exactly defined but are assumed to be generally understood, or they end up in circular definitions. For example, geographers have tried to define these features, but the definitions are often incomplete or contradictory. In geography, most terms are defined by natural language (e.g., English).

However, a **3D** vision system cannot rely on these definitions for extracting the topographic features. In the following two sections, we examine the two alternative defi-

1049-9660/94 $6.00

nitions which overcome the problems of the natural language definitions.

2.1. Definitions Based on Elevation Data

To avoid the problem of dependency on generally accepted terms or circular definitions, there has been a wide variety of techniques to define features based on a procedure for the identification of a feature [5].

Other researchers defined ravines and pits as all the locations at which water would flow or collect if water were poured on the terrain [14]. According to this definition, ravines or pits correspond to points lying at a local minimum in elevation. Ridges and peaks are duals to ravines and pits, respectively. In other words, ridges or peaks correspond to points lying at a local maximum in elevation.

More recently, many researchers have used concepts from differential geometry to classify the points on a surface into basic classes of several features [2, 3]. From the theory of differential geometry, combinations of values of the two principal curvatures uniquely define features. Classifying terrain features in terms of curvature has a problem. If we take a peak and then rotate it about a horizontal axis, it soon stops being a peak, even if curvature, being intrinsic to the shape, does not change. Gravity clearly plays a role, and curvature has nothing to do with it.

There are a couple of problems in finding topographic features based on these definitions: differential geometry, local maximum, and minimum. First, due to the locality of the methods, it is very noise sensitive and produces many false features. Second, it is easy to extract special points based on pure shape descriptors, such as high curvature or local maximum. Identifying those as ridges or ravines requires another procedure (e.g., grouping) and more context, such as the shape of the surrounding terrain.

2.2. Definitions Based on Contour Lines

Those definitions described in the previous section may have a sound mathematical justification. However, human beings are capable of recognizing topographic features from a contour map without knowing the detailed elevation of the terrain. Based on this observation, topographic features can be qualitatively defined using the shape and relationship of contour lines on contour maps [9].

Let an elevation map be a function $z(x, y)$, where (x, y) denotes a location in a geographic domain. Peaks or pits can be defined as a series of closed contours. From the elevations of the contour lines shown, a peak or pit is defined as having either ascending or descending elevations. In terms of elevations, peaks and pits are corresponding to local maxima and minima, respectively. We define peaks and pits as

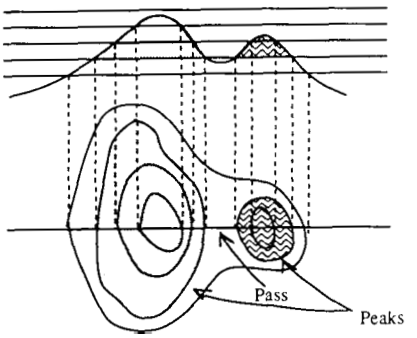


FIG. 1. Contour lines fur a peak.

$$\begin{aligned} \text{Peak:} &\quad \max z(x, y) \\ \text{Pit:} &\quad \min z(x, y). \end{aligned} \tag{1}$$

Figure 1 shows contour lines for two peaks and a pass.

Ravines can be defined as the contour lines forming modified V's pointing upstream. Similarly, ridges can be defined as the contour lines forming modified V's pointing downstream.

A ravine is shown in Fig. 2 whose contour lines form smooth and rounded V shapes. The same contours indicate a ridge if we reverse the elevations. We use these definitions for detecting topographic features.

The contour-based definitions for ridges and ravines are still very heuristic and qualitative. We present a new definition for ridges and ravines and show that the new definition is equivalent to the contour-based definition.

Let us assume that we are at a point on a ridge (at point *A* in the left figure of Fig. 3). Along the tangential direction, denoted as t, we have sudden changes in elevations. Along the normal direction, denoted as n, we have small changes in elevations.

The right figure of Fig. 3 shows the cross section of the terrain by a vertical plane through t. If a ball is displaced slightly from this point, it rolls away along the steepest direction. Similarly, if water were poured at the point, water would naturally flow along the same direction.

Now, we are ready to give a formal definition of ridges

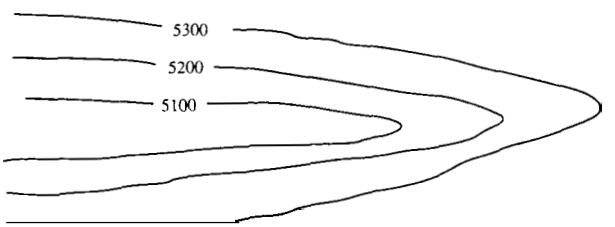


FIG. 2. Contour lines for a ravine

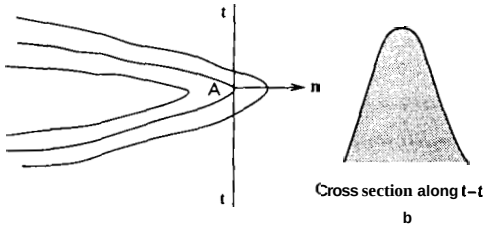


FIG. 3. Contours of ridges and a cross section along the direction of gravity.

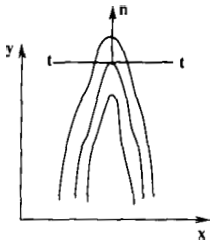


FIG. 4. The rotated contour line and tangent plane at a ridge

and ravines, based on the above observations. We first note that for every point of an analytic function of two variables there is a direction along which the first derivative is zero: this is the direction tangent to the iso-elevation contour.

Let C be the iso-elevation contour. We denote the directional derivative of z at the point (x, y) in the direction t by $z'_t(x, y)$. We define the second derivative of L along the direction tangent to the contour, t, at (x, y) on C as

$$z''_t = \partial^2 z/\partial t^2. \tag{2}$$

Similarly, we define the first derivative of z along the direction normal to the contour, n, at (x, y) on C as

$$z'_n = \partial z/\partial n. \tag{3}$$

Using Eqs. (2) and (3) we can formally define ridges and ravines as

$$\text{Ridge:} \quad \max_{(x,y)\ \text{on}\ C} \frac{z''_t}{z'_n}$$
$$\text{Ravine:} \quad \min_{(x,y)\ \text{on}\ C} \frac{z''_t}{z'_n}. \tag{4}$$

Equation **(4)** states that a ridge point has the maximum change in the gradient of elevation in the tangential direction (z''_t) and has the minimum elevation change in the normal direction (z'_n) along the contour.

Now we want to show that the local extrema in z''_t/z'_n occur at the point on the contour where the shape of contour forms a **V.** We first note that the shape of a **V** corresponds to the local maxima in curvature along the contour. We then assume the following two conditions:

1. $z(x, y)$ is a twice differentiable function, and
2. $(\partial z/\partial x, \partial z/\partial y) = (z'_x, z'_y) \neq (0, 0)$, z'_x and z'_y cannot be zero at the same point (we do not consider peaks or pits).

Without **loss** of generality, we can always rotate (x, y) axes so that the tangent plane t is horizontal as shown in Fig. **4.** In the new reference frame, our definition of ridge can then be stated as

$$\max_{(x,y)\ \text{on}\ C} \frac{z''_t}{z'_n} = \max \frac{\partial^2 z}{\partial x^2} \Big/ \frac{\partial z}{\partial y} = \max \frac{z''_{xx}}{z'_y}. \tag{5}$$

As shown in Fig. **4,** we can represent the contour as $y = f(x)$ and find

$$f' = 0 \tag{6}$$

in the neighborhood of the point A. The iso-elevation contour is represented as

$$z(x, y) = z_0, \tag{7}$$

where z is assumed to have first partial derivative.

Using Eq. **(7)** we can apply the implicit function theorem and a fundamental result of differential geometry to obtain [8, 4]

$$\max_{(x,y)\ \text{on}\ C} \frac{z''_t}{z'_n} = \max \frac{z''_{xx}}{z'_y} = \max |f''| = \max K. \tag{8}$$

Equation **(8)** means that the local extrema in curvature along the contour occurs at the point where the local extrema in z''_t/z'_n occurs.

3. CONSTRUCTING CONTOUR MAPS FROM ELEVATION MAPS

Building a contour map from an elevation map is trivial: imagine a series of parallel cuts which intersect the profile of the terrain. Next, define successive plateaus of constant elevation by planar cuts located at a particular distance below **or** above a reference plane.

Starting from a reference elevation H_c, the algorithm for building a contour map works in four steps:

1. Create a binary image B_{ij} by assigning

$$B_{ij} = \begin{cases} 1 & \text{if } H_{ij} \geq H_c \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

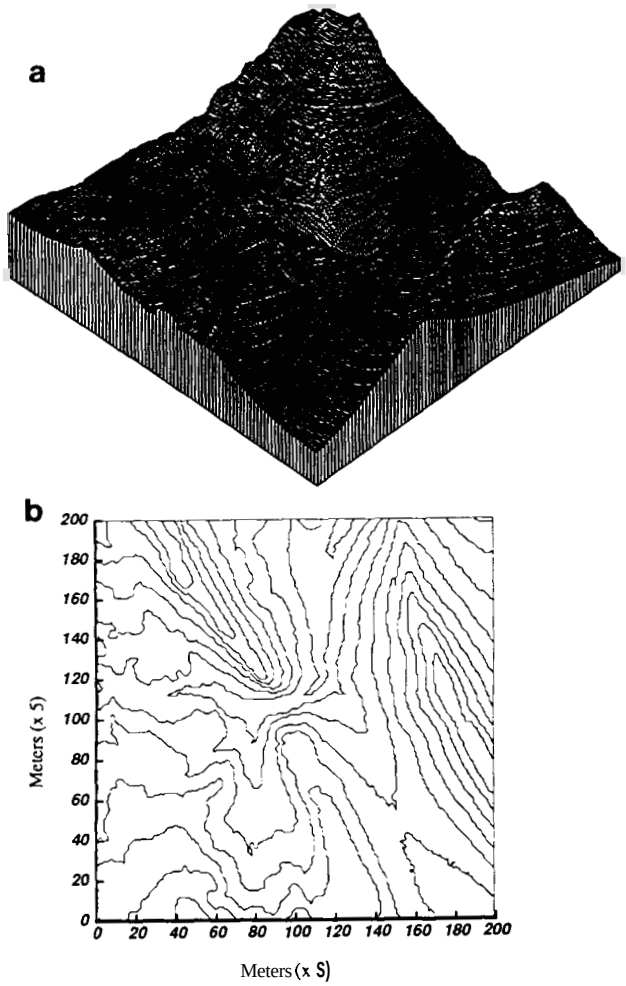


FIG. 5. Result of constructing a contour map from an elevation map: (a) a DEM; (b) extracted contour map.

where H_{ij} is an elevation at each grid point ij of the elevation map;

2. Extract the connected components by using blob-coloring algorithm;

3. Compute the contour lines by fitting polygons to the connected components;

4. Repeat steps from (1) to **(3)** at a new elevation

$$H_{,.} = H_{c} + \Delta H. \tag{10}$$

where ΔH is an increment parameter.

Figure 5 shows an elevation map and corresponding contour map.

4. CONSTRUCTING THE CONTOUR TREE FROM A CONTOUR MAP

In the previous section, we described a simple algorithm for building a contour map from an elevation map. In this section, we present a tree data structure, the contour tree, which can represent the nesting of contour lines on a continuous topographic surface [13]. The contour tree represents the relationships among contour lines on a contour map. We call this contour tree representation a ***topographic change free*** (TC tree).

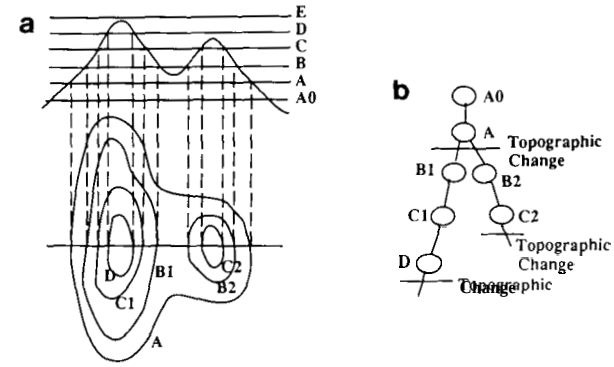


FIG. 6. An example of a topographic change tree for ID.

In a TC tree, each node represents a cross-sectional area of terrain intersected by a planar cut, and links between nodes represent parent–child relationships (i.e., a path from one area to another). Every node has a list of descendants, its corresponding elevation value, a list of edge points approximated by a polygon, and its parent.

In general, we can have three links connecting the two nodes. First, single parent–child relationship between two nodes indicates no topographic change, noted as no-change (e.g., between the planar cuts $A0$ and **A** in Fig. **6).** Second, multiple connections among nodes or a ***branch*** indicates topographic changes among them (e.g., between the planar cuts A and B in Fig. **6).** Third, no child at a particular node also means a topographic change has occurred between the two elevations (e.g., between the planar cuts C and D in Fig. **6).**

Starting from the root node with minimal elevation, we recursively create the contour tree in a depth-first fashion. **A** depth-first tree generation expands the most recently generated node first. When it reaches a node that has no descendants, it visits an unexplored node at the nearest depth. We have tested this algorithm on a real terrain data, a DEM. Figure **7** shows the TC tree of the contour map shown in Fig. **5.**

5. EXTRACTING PEAKS AND PITS FROM THE CONTOUR TREE

Peaks and pits are extracted by finding a series of closed contours from the TC tree, described in Section **4.** The TC tree is recursively traversed for extracting a series of closed contours. The algorithm is a graph traversal procedure to label each region encircled by a contour and is expressed in pseudocode at Table 1.

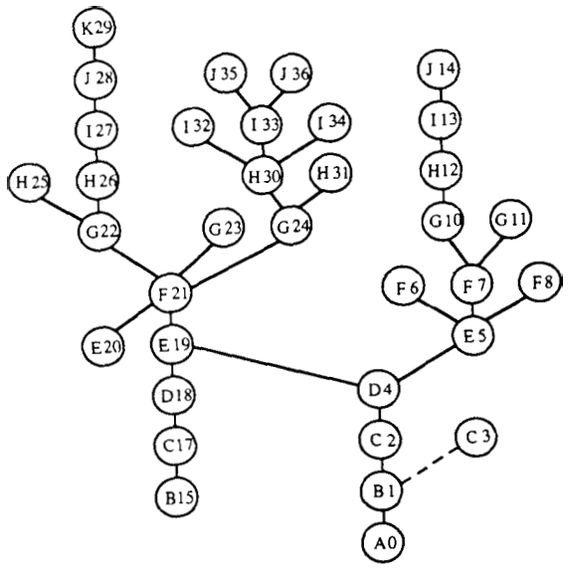


FIG. 7. TC **tree representing the contour map.**

In addition to peaks and pits, we can detect passes between two peaks from two or more contours of the same level enclosed by common lower neighbors. Figure 8a shows a contour map and the corresponding contour tree for peaks and a pass. The areas (or nodes in the contour tree) shaded with dots and wave patterns indicate the resulting peaks to be extracted when we apply the above algorithm starting the trace at the elevation AO. Similarly, Fig. 8b shows the same results for extracting pits. In this case, the terrain for extracting peaks is flipped with respect to the horizontal axis. However, all these examples are relatively simple and cannot cover all possible combinations of contour lines in a 3D real world.

In 3D, the relationship between contour lines of the same elevation can be divided into two cases: (1) two contours are completely separated from each other as shown in Fig. 8; (2) one contour is enclosed by the other contour (for example, we can observe this case with a volcano in real terrain).

We explain the second case using a simple example of a volcano. Figure 9a shows the contour lines and the contour tree for extracting peaks, created by

- starting the cross section cut from the minimum elevation A,
- creating blobs of the map where the elevations are greater than A,
- extracting the contour lines by following the border of the blobs, and
- repeating the proceeding steps at elevations E, C, and D.

At elevation A, we have one contour line which contains every point in the map. At elevation B, however. we have

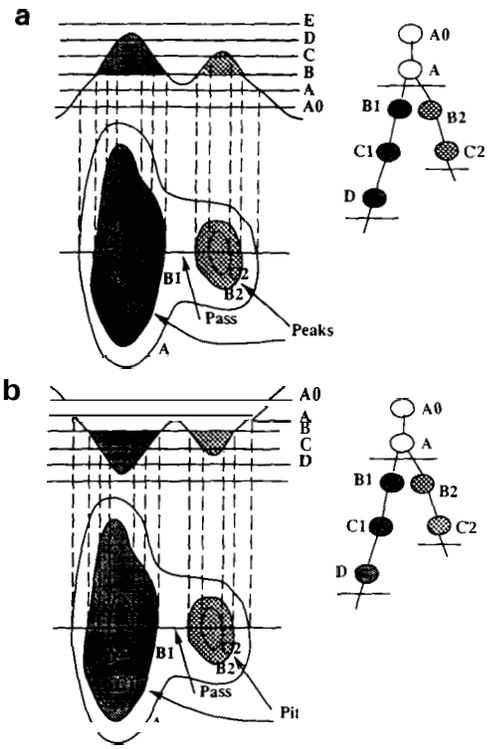


FIG. 8. Examples of a contour map and the corresponding contour tree: (a) for peaks and a pass; (b) for pits and a pass.

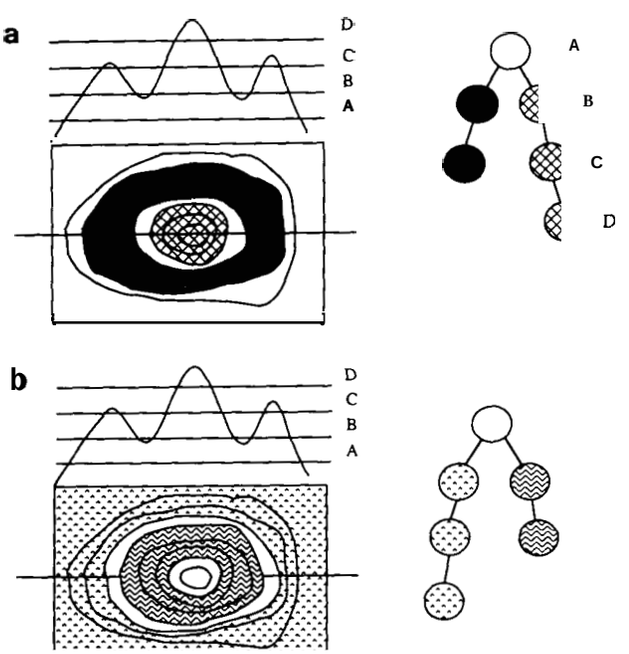


FIG. 9. Contour trees, extracted peaks. and pits for a **volcano:** (a) **peaks;** (b) **pits.**

TABLE 1
An Algorithm for Extracting a Series of Closed Contours

```
extract-connected-closed-contours (origin-node, current-node)
    M ← the number of children of current-node
    if M > 1                    /* if there is a branch at the node */
                                /* assign a new origin-node */
    then
        origin-node ← current-node
    else if M = 0               /* no child at the current node */
        get contours from the nodes between current-node and origin-node
    end if
    for j = 1 to M do           /* recursive call for each child node */
        current-node ← child of current-node
        extract-connected-closed-contours (origin-node, current-node)
    endfor j
```

two contour lines: one corresponding to a peak at the center represented by a diamond shade in the contour tree at elevation B, and the other from the outer boundary at the same elevation and shaded with dots in the contour tree. By using the algorithm described in Table 1, we extract a series of closed contours, shown as shaded areas with dots and diamonds in Fig. 9a.

In a similar way, we create another contour tree for extracting pits by

- starting the cross section cut from the maximum elevation **D,**
- creating blobs of the map where the elevations are smaller than D,
- extracting the contour lines by following the border of the blobs, and
- repeating the proceeding steps at elevations C, E, and **A.**

From the resulting contour tree, we observe one topographic change between elevations C and **D.** This topographic change between contour lines defines two pits as shown in Fig. 9b.

From the result of this example, the correct labels for the self-included contours can be obtained by creating two contour trees: one from downward; the other from the upward direction.

To examine the above observations, we apply the algorithm to real terrain data, a DEM shown in Fig. 5a. Figure 10 shows extracted topographic features, peaks and pits and the corresponding contour tree. From this real data, we observe the second case of the contour relation—one contour is enclosed by another contour even though the

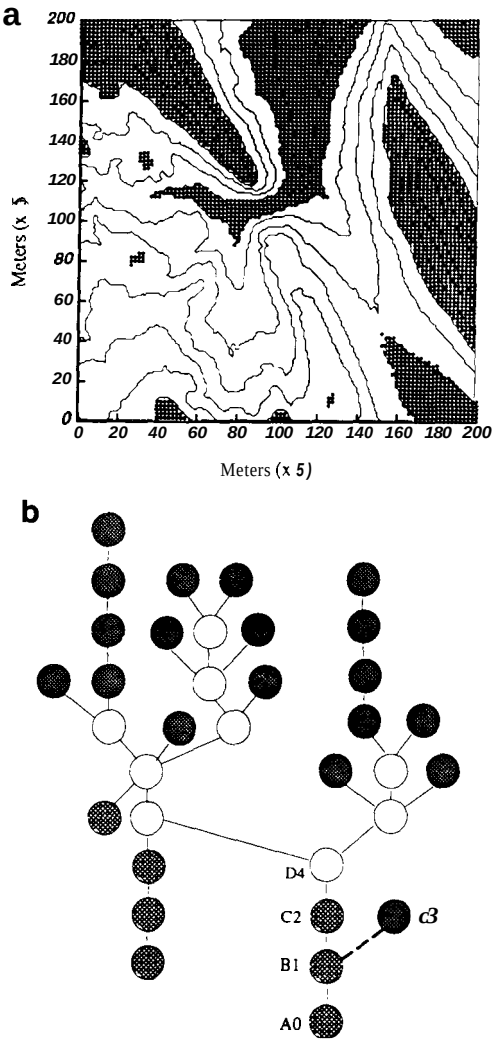


FIG. 10. Results for **the extraction** of **peaks and pits** from a DEM: **(a) peaks and pits;** (b) **the contour tree.**

contours have the same elevations. For example, two contours, C_2 and C_3, have the same elevations and C, is enclosed by C_2. In other words, the area encircled by the contour C_3 is an island surrounded by an area defined by the contour C_2. When we create the contour tree for pits (or in a downward direction starting from the maximum elevation), we do not have a node C3. **As** a result of the extraction of a series of closed contours, we have a pit consisting of nodes, shaded with diamonds, $A,,, B_1$, and C_2. However, a node C_3 is created from the contour tree which is generated by starting the cut at the minimum elevation in upward direction. Since we observe a topographic change between the two elevations E and C, a peak corresponding to a node C_3 is extracted.

The extracted peaks and pits at least qualitatively (or visually) correspond to what we observe from the original elevation map. We conclude that the results of experi-

ments support the feasibility of applying the algorithms to real data.

6. EXTRACTING RIDGES AND RAVINES FROM THECONTOURTREE

According to the definitions of ridges and ravines based on contour lines, the contour lines consisting of ridges or ravines have a modified V shape. We also showed that the point whose contour shape forms **V** has the local extrema in z_t''/z_n' along the tangential direction on the contour. Also note that the shape of V corresponds to the local maximum in curvature on the contour. Therefore, ridges and ravines can be extracted if we extract those local maxima (or minima) of the curvatures from the contour, and group them together.

There exists a couple of problems in applying this method to real terrain. One difficulty is that the shape of V changes from a very sharp to a smooth and round one depending upon the underlying materials. Another problem is how to reliably group each feature point into a ridge or ravine. We present our novel method which can gracefully handle these problems.

In this section, we first present a method for reliably extracting a **V** shape from a contour line. We represent the contour lines by chain codes. Each element in a chain code represents the curvatures of their respective curve segments. We extract local maxima of the curvatures by using the scale space approach.

We then describe an algorithm to group those extracted features into ridge- and ravine-lines by using the topological relationship among the features, represented by the contour tree.

6.1. Extracting Extrema Points

We present an algorithm for extracting the local extrema in curvature along contour lines. The algorithm works in two steps: (1) the contours are represented by eight-directional chain-codes; (2) we then divide the contours into line segments by detecting the extrema curvature points on the contours.

The chain-code representation is calculated by computing the angle change at a particular point along the contour. We use the arm method, introduced by Rosenfeld [12], to compute the angle change.[1]

To detect the extreme points at different scales, we analyze the contours in the scale space. The scale space approach was applied to the contour analysis by Brady and Asada [1]. They traced the features of the contour across the different scales in the scale space.

We find the proper scales (arm lengths) for each local contour shape and extract the local extrema in curvature along the contour lines in the meaningful part of the scale space.

[1] **The angle change can be considered to be the curvature because the definition of curvature is given as** $\kappa = \delta\theta/\delta s$.

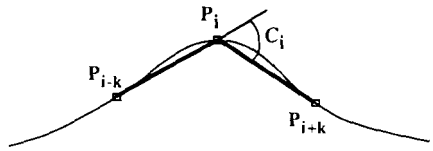


FIG. 11. Definition of curvature at P_i with arms.

6.1.1. *Calculation of the Curvature*

Curvature is defined to be the derivative of the tangential slope with respect to arc length. To calculate the curvature, we use the simple arm method which can directly compute the derivative of the tangential slope [12]. Curvature at a point P_i on a contour is defined by angle C_i as shown in Fig. 11.

Let V_{ij} be a vector from a point P_i to a point P_j. Given points P_{i-k} and P_{i+k} which are k chains away from P_i, the angle C_i is then given as

$$C_i = \arccos \frac{V_{i-k,i} \cdot V_{i,i+k}}{|V_{i-k,i}| \times |V_{i,i+k}|} \tag{11}$$

The arm length, k, controls the amount of smoothing. For example, short arms are too sensitive to the digitized direction and other noise sources. Longer arms are better for smoothing the noisy contours. However, long arms do not always show the proper curvature if the arms are too long to follow the contour shape. In other words, arm length should be dynamically and locally selected by checking how well the arms follow the local contour shape. If some points on the contour are far away from the arms, the arm length is not appropriate for the local point.

We use multiple arm lengths and compute the multiple curvature curves ($\theta' - s$ curves). At each scale, we extract the candidate areas of the local extrema in curvature as a rough estimate of feature positions. The candidate areas are selected only when both arms fit the contour and the curvature is above a specified threshold. A coarse-to-fine strategy is then applied to these candidate areas in the multiple curvature curves. During this coarse-to-fine tracing, the arm length becomes short and the candidate area is narrowed down. Peaks in the resulting corner areas are picked up as high curvature points.

Tracing the area instead of the point has several merits: (1) We do not need a strict thresholding for detecting high curvature points in the presence of the noise peaks. (2) The ambiguity of the feature point positions in different scales does not make the tracing problem difficult.

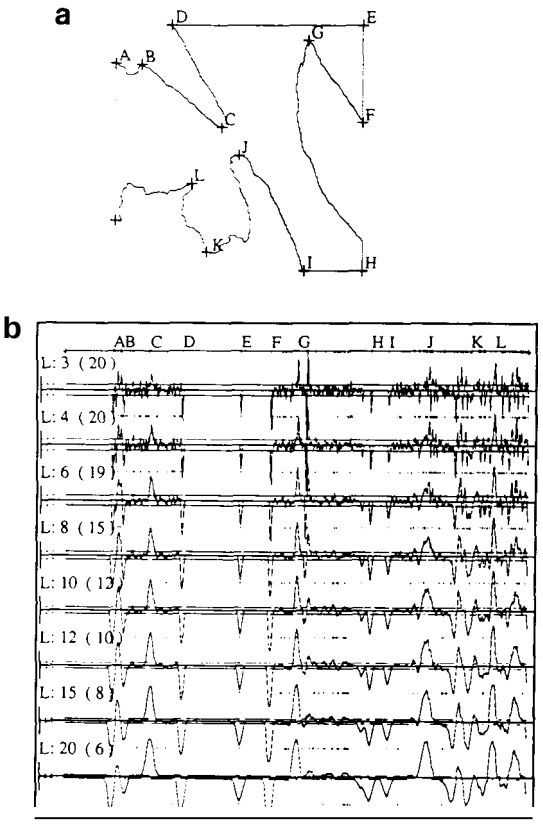


FIG. 12. Results for contour analysis by the multi-arm method: (a) a contour line; (b) the curvatures at multiple scales.

6.1.2. Example of Contour Analysis

We applied the multiple-arm method to contour analysis for contour maps. Contour lines, represented by an eight-direction chain-code, are obtained from a contour map as described in the above section.

Here we present an example of the multiple-arm method for a contour line. Figure 12a shows the analyzed contour line with feature points (or high curvature points), denoted from *A* to *L*. The eight curvature curves, calculated with eight different arm lengths from 20 to 3, are shown in Fig. 12b with the coarse-level at the bottom and the fine-level at the top. The unit of the horizontal axis is the arc length. The dotted lines to the left of each curve show the threshold for each arm length. To the right of the dotted lines, the two lines indicate whether the two arms fit to the contour; the upper (or lower) line shows the area where the left (or right) arm fits the contour. A blank indicates that the arm does not fit the contour. Small dots below each curvature line indicate the position of the high curvature point for the corresponding arm length. Those points satisfy two conditions: (1) both left- and right-arm fit the contour; (2) the resulting curvature is above the threshold. The topmost diagram shows the detected high curvature points along the contour line.

Figure 13a shows the contour map with the contour interval 50 feet and the isoplot of the corresponding DEM. Figure 13b shows the resulting high curvature points, denoted as +, overlayed on the contour map. These results demonstrate that the multi-arm method provides accurate and reliable high curvature points on the contour lines.

6.2. Tracing and Linking Ridge and Ravine Points

We described how to extract ridge points in the above section. The next important step is to convert the extracted ridge points into a list of lines. One can start tracing at the point of lowest elevation and recursively look for neighbor points which have characteristics similar to the current ridge point. This approach has several disadvantages: (1) searching only local neighbor points produces many false lines due to noise sensitivity; (2) the neighbor points may not have a direct topological relationship with the current point, even though they are spatially adjacent to each other.

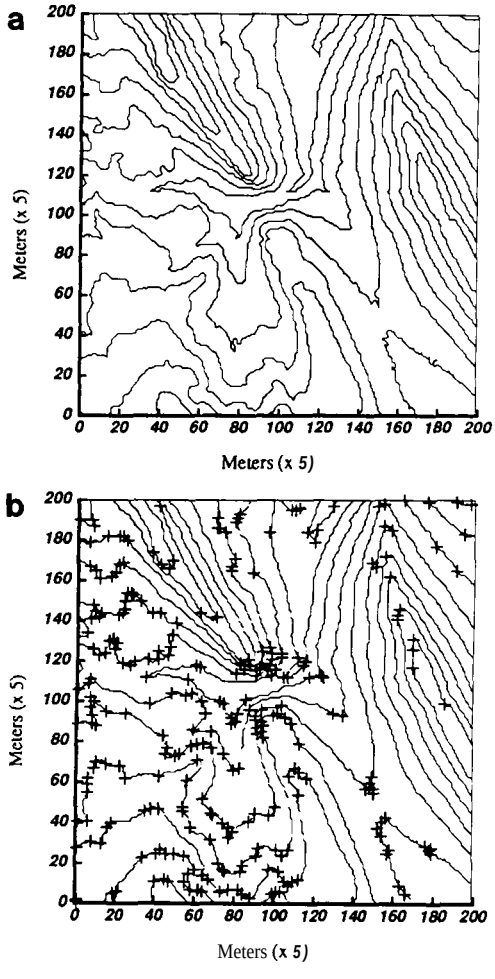


FIG. 13. Results for extracting the local extrema in curvature: (a) a contour map; (b) the local extrema in curvature.

We present an alternative method for tracing and linking ridge (or ravine) points by using the contour tree. As we explained in Section **4,** the contour tree provides the topological relationship among contour lines on a contour map. Assume that we have a known (a starting) ridge point in the current node of the contour tree. Then we first look for the ridge points in the children nodes of the current node because the topological relationship embedded in the contour tree representation indicates that those points in the children nodes are the only topologically possible connections to the current point. If no points are found in the children nodes, it recursively visits the children of its children.

The traversal of the contour tree is done in a breadth-first fashion. A breadth-first search visits all the children of a node before visiting any of the children of its children.

For each ridge point of each child node, we compute some attributes which are used to search for the best ridge point connecting to the current ridge point. The attributes to be computed include:

- The current global direction of a ridge, determined by the slope of the line drawn from the previous ridge point to the current point (*a* in Fig. **14);**
- The current local direction of a ridge, determined by the slope of the line drawn from the center of the curvature to the current point (β in Fig. **14);**
- The Euclidean distance from the previous ridge point to the current point (ρ in Fig. 14).

Only points within $\pm 90°$ of the global and local directions are considered candidates for the next point. These restrictions eliminate radical direction changes in the ridge. The candidates are further reduced by finding those having the minimum distance from the current point. If there is still more than one, the points resulting in the least angle change in the ridge are found.

In addition to angle and distance constraints, we use the similarity of the shape to find the ridges. The idea is to choose the feature point with the most similar curvature value with the known ridge point. To solve the ambiguity of the feature point positions in different contours, we use the neighbor points of the local extreme point to compute the correlation of curvature values between the two high curvature points. We determine the size of the neighbor by using the most appropriate arm length which is determined by the contour analysis. We then define the curvature similarity between two feature points, C_i and P_j, by

$$S_{ij} = \sum_{k=-l}^{k=l} \sqrt{(C_{i+k} - P_{j+k})^2/2l}, \tag{12}$$

where l refers to the arm length.

To find out the best matching feature points from the previous contour with the known feature point from the current contour, we use four constraints, such as the angle change between the current and previous local directions ($\Delta\beta = \|\beta_C - \beta_P\|$), the angle change between the local and global directions ($\Delta\alpha_C = \|\alpha - \beta_C\|$, $\Delta\alpha_P = \|\alpha - \beta_P\|$), the distance between two feature points ***(p),*** and the curvature similarity (S). The subscript ***P*** and C refer to the previous and current feature points, respectively.

We present a heuristic function to measure how well a feature point satisfies each constraint. We use the distance between the value of each constraint and the corresponding threshold. If the value of a constraint of a feature is very close to the threshold, the likelihood of the feature being the best corresponding feature point should be very small and vice versa.

Based on this intuition, the simplest heuristic function is defined by

$$V = \frac{\Delta\beta}{\Delta\beta_{\text{thres}} - \Delta\beta} + \frac{\Delta\alpha_C}{\Delta\alpha_{C\,\text{thres}} - \Delta\alpha_C} + \frac{\Delta\alpha_P}{\Delta\alpha_{P\,\text{thres}} - \Delta\alpha_P} + \frac{\rho}{\rho_{\text{thres}} - \rho} + \frac{S}{S_{\text{thres}} - S}. \tag{13}$$

We do not have to consider degenerate cases for the above formula since we first remove the features which do not satisfy the threshold.

By following the same procedure for all other feature points along the contours, we can obtain the ridge and ravine lines. Now, we present the results of tracing and linking ridge and ravine points using the whole DEM which is shown in Fig. 13a.

Figure 15 shows the extracted topographic features

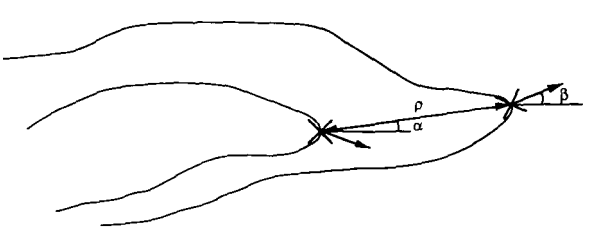
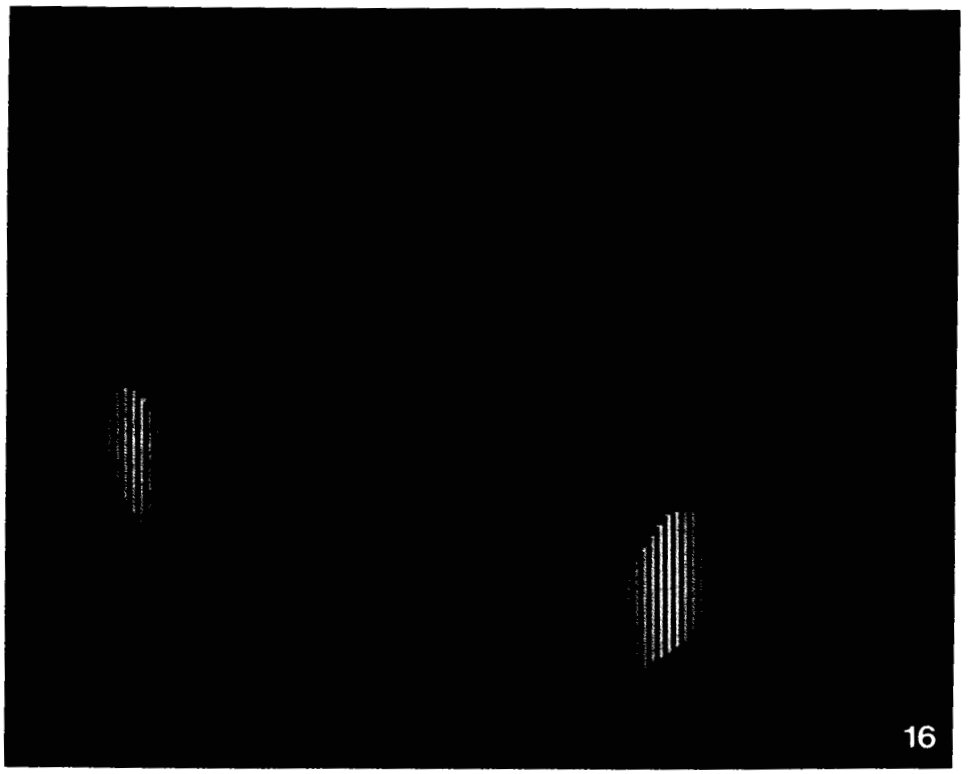


FIG. 14. The attributes for linking process.

FIG. 15. Extracted topographic features from a DEM



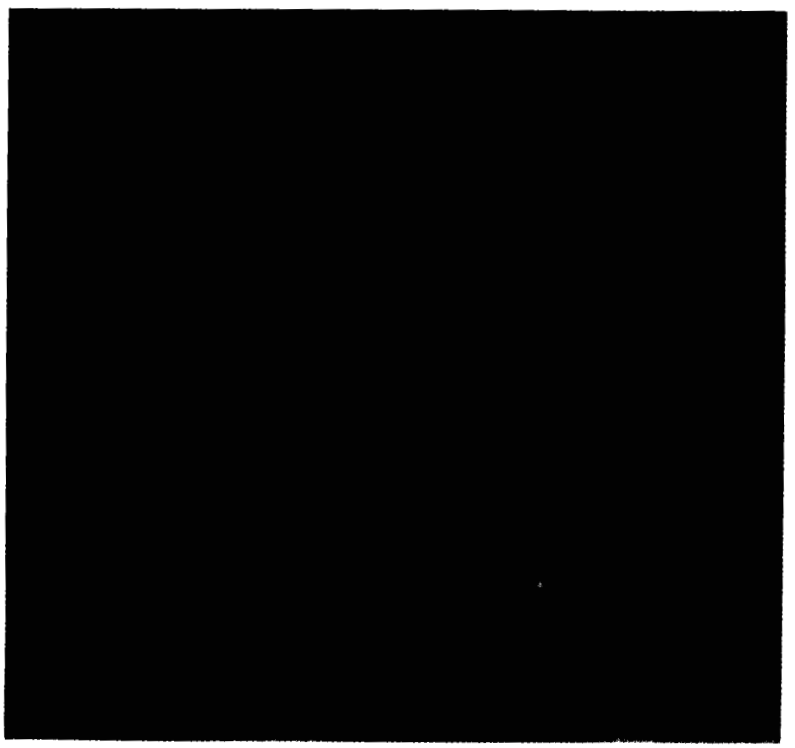


FIG. *16.* Perspective view of the extracted topographic features.

FIG. 17. Extracted topographic features from **a** DEM.

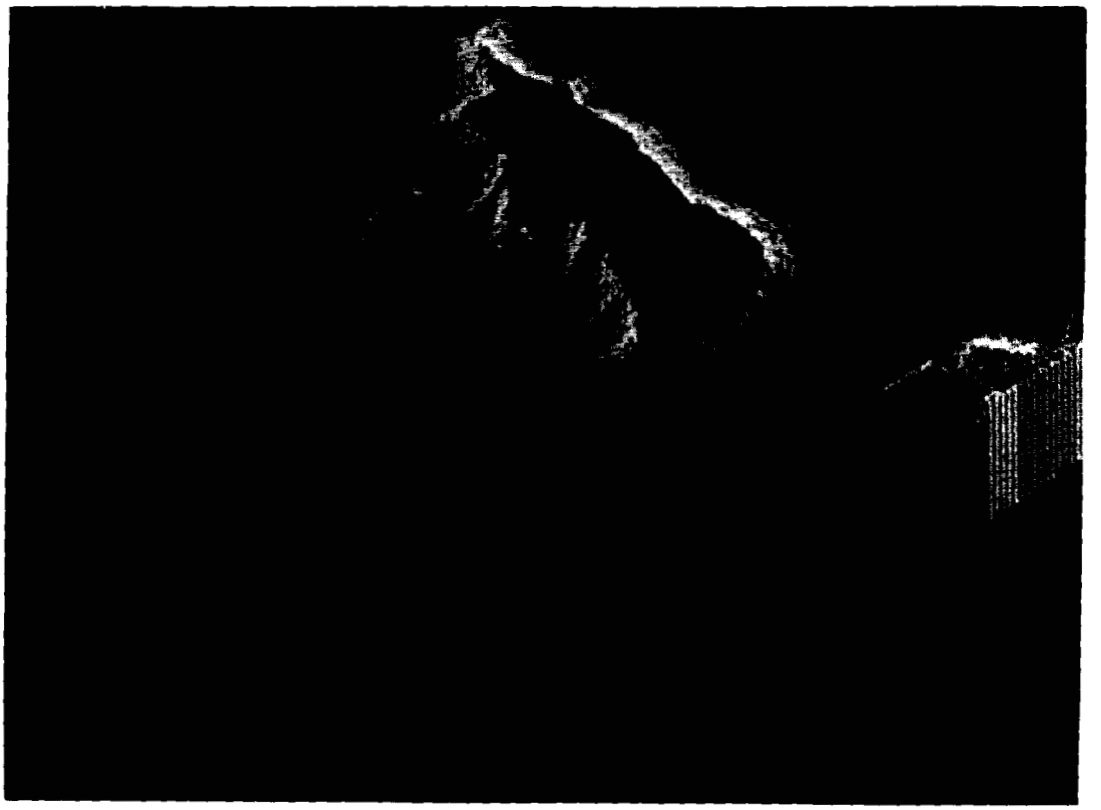


FIG. 18. Perspective view of the extracted topographic features

from the DEM. Black lines indicate ridges; white lines indicate ravines. Black and white regions correspond to peaks and pits, respectively. The features are overlaid on the DEM in which the lighter intensity indicates the higher elevation.

Figure 16 shows the perspective view of the extracted topographic features. White lines and regions represent the extracted ravines and pits. Black lines and regions indicate the extracted ridges and peaks.

Figure 17 and Fig. 18 show the extracted topographic features from another DEM. White lines and regions show the extracted ravines and pits. Black lines and regions correspond to the extracted ridges and peaks.

7. CONCLUSIONS

In this paper, we presented algorithms for extracting topographic terrain features from elevation maps, represented as the contour tree. First, we developed the definitions of topographic features and have shown that a point with a local extreme in curvature on the contour has a local extreme in z_t''/z_n' on the contour, which is our definition of ridges and ravines.

Extracting reliable high curvature points is a very important intermediate stage to obtain correct high-level scene descriptions of rugged terrain. The multi-arm method working in the multi-scale space provided noise-insensitive high curvature points that correspond to ridge or ravine points in the terrain.

Using the contour tree representation. we were able to reduce the complexity of the grouping process in converting ridge (or ravine) points into a list of ridge (or ravine) lines and even to remove any grouping process for peaks and pits.

In processing a 200 $\times$ 200 DEM, it took approximately 90 CPU seconds on a Sparc machine. In a series of experiments, an increment parameter, AH, was set to 50 feet for DEMs which have approximately 500 feet of total height change.

The method presented in this paper uses an increment parameter, AH, in order to generate contour maps at various height intervals. We observe that a too small value of the parameter produces a noisy contour map due to a large number of contours. However, the linking process from ridge (or ravine) points *to* lines becomes a much easier task since the distance between successive contour lines is small. A large value of AH results in poor localization of the features. Work is currently underway to improve the robustness of the method by using a multi-resolution approach.

ACKNOWLEDGMENTS

The authors wish to thank Martial Hebert for his ideas and valuable discussions on this work. This paper describes research done at the Robotics Institute of Carnegie Mellon University.

REFERENCES

I. H. Asada and M. Brady. The curvature primal sketch, *IEEE Trans. PAMI* **8,** 1986.

2. P. J. Besl and R. C. Jain, Segmentation through symbolic surface descriptions, in ***CVPR. May 1986.***

3. M. Brady. J. Ponce. A. Yuille, and H. Asada, Describing surfaces, *Comput Vision Graphics Image Process. 32,* 1985.

4. I. D. Faux and M. J. Pratt. ***Computational Geometry for Design and Manufacture***, Ellis Horwood, Chichester, UK, 1979.

5. R. Haralick, L. Winston, and T. Laffey, The topographic primal sketch, *Int. J. Rob. Res. 2,* 1983.

6. S. K. Jenson, Automated derivation of hydrologic basin characteristics from digital elevation model data, in ***ASP/ACSM Conference,*** *March 1985.*

7. E. G. Johnston and A. Rosenfeld, Digital detection of pits. peaks. ridges, and ravines, *IEEE Trans. Systems Man Cybern.* July 1975, 472.

8. I. S. Kweon, ***Modeling Rugged Terrain by a Mobile Robot with Multiple Sensors,*** Ph.D. thesis, The Robotics Institute. Carnegie–Mellon University, August 1990.

9. F. H. Moffitt and H. Bouchard, ***Surveying,*** Harper & Row. New York, 1982.

10. J. F. O'Callaghan and D. M. Mark, The extraction of drainage networks from digital elevation data. *Comput. Vision Graphics Image process. 28,* 1984.

11. D. J. Orser and M. Roche, The extraction of topographic features in support of autonomous uncerwater vehicle navigation. in ***Fifth International Symposium on Unmanned Untethered Submersible. March 1987.***

12. A. Rosenfeld, Digital straight line segments, ***IEEE Truns.*** *Comput.* ***23,*** 1974.

13. J. Roubal and T. K. Poiker. Automated contour labelling and the contour tree, in *Auto-Carto, March 1985*

14. W. W. Seemuller, The extraction of ordered vector drainage networks from elevation data, *Comput. Vision Graphics Image Process.* **47,** 1989.