A 1 $\log N$ Parallel Algorithm for Detecting Convex Hulls on Image Boards

Ja-Chen Lin and Jenn-Yih Lin

***Abstract*— By finding the maximum and minimum of $\{y_i - mx_i \mid 1 \le i \le N\}$ for certain slopes m, we propose here a simple and fast parallel algorithm to obtain the convex hull of N arbitrarily given points on an image board. The mathematical theory needed is included, and computation time is $1 \log N$.**

***Index Terms*—Bottom path, convex hull, parallel, slopes, top path.**

I. Introduction

Finding the convex hull (CH) given planer points is an interesting combinatorial problem, the result of which can be applied to pattern recognition [1], computer graphics and image processing [2], statistics [3], etc. When a sequential computer is used to find the CH of N given points, the time used is $\Omega(N \log N)$. (Several $O(N \log N)$ time algorithms have been developed [3].) When a parallel computing machine is used, more specifically, when a concurrent-read exclusive-write (CREW) parallel random access machine (PRAM) is used, the fastest existing algorithms use $O(\log N)$ computation time [4]–[5]. However, none of these $O(\log N)$ algorithms can have the computation time as low as $1 \log N$ (if each arithmetic or logic operation takes one unit time).

When the CH problem is encountered in image processing, the problem is somewhat different. Because the resolution of the image board is known in advance, say 512 × 512, the points in the image board will have integer coordinates that belong to a known finite set (e.g., $\{(x, y) \mid x = 1, 2, \cdots, 512; y = 1, 2, \cdots, 512\}$). Finding the CH is therefore easier. At least two parallel algorithms [6], [7] have been proposed to solve the CH problem in image boards, and their computation time is, if the image board is a 512 × 512 board, at least $\log(512 \times 512)$. In this correspondence, we propose a fast parallel algorithm whose computation time for finding the CH of an N-point set is $1 \log N$ (assuming these N points are N pixels taken from an image board). Since $N \le 512 \times 512$, the computation time is shorter than that of [6] and [7].

The proposed method is described in Section II below. The corresponding parallel CREW algorithm is included in Section III. Section IV is the conclusion.

II. The Proposal Method

Let $E = \{(x_n, y_n)\}_{n=1}^{N}$ be an input point set of which the CH is to be found. It is well known that every vertex of CH boundary belongs to the input point set [8]. We therefore try to identify those input points that should be vertices and arrange these points in a suitable order to form the boundary of the CH. In general, the CH boundary encloses a closed region called the CH region. The boundary can

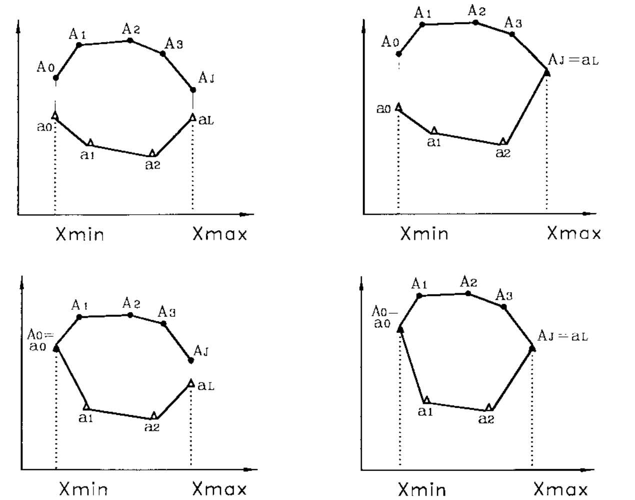

Fig. 1. Four possible cases of decomposition of the boundary of a convex hull. There are J and L line segments in the top path and bottom path, respectively. In this example, $J = 4$ and $L = 3$.

be decomposed into (at most) four parts, called the top path, the bottom path, the left edge, and the right edge [see Fig. 1, top left]. The left edge is the shortest vertical line segment containing input points $\{(x_n, y_n)\}$ of which the x coordinates are all equivalent to $x_{\min} = \mathrm{Min}\{x_n \mid 1 \le n \le N\}$. The right edge can be defined likewise, with $x_{\min}$ replaced by $x_{\max} = \mathrm{Max}\{x_n \mid 1 \le n \le N\}$. When there is only one input point of which the x coordinate is $x_{\min}$ ($x_{\max}$), then the left (right) edge degenerates into a single point called the left (right) vertex [see Fig. 1, top right and bottom left and right]. The top path and bottom path are defined by the two-dimensional (2-D) point sets $\{(x, f(x)) \mid x_{\min} \le x \le x_{\max}\}$ and $\{(x, g(x)) \mid x_{\min} \le x \le x_{\max}\}$, respectively, if the CH region is defined by $\{(x, y) \mid g(x) \le y \le f(x), x_{\min} \le x \le x_{\max}\}$ for some piecewise linear functions $f(x)$ and $g(x)$ defined on the interval $x_{\min} \le x \le x_{\max}$. In other words, $f(x) = \mathrm{Max}\{y \mid (x, y) \in$ CH Region$\}$ and $g(x) = \mathrm{Min}\{y \mid (x, y) \in$ CH Region$\}$.

Obviously, the CH is completely determined if we can obtain the top path $y = f(x)$ and the bottom path $y = g(x)$, because the left and right edges can be generated by connecting the left and right ends of the top and bottom paths. Without loss of generality, we show how to obtain the top path. Note that both the top and bottom paths are piecewise linear (i.e., formed of several line segments) because the CH region is a convex polygon. To obtain the top path $\overline{A_0A_1} \cup \overline{A_1A_2} \cup \cdots \cup \overline{A_{J-1}A_J}$ (reading from the leftmost vertex A_0 through the rightmost vertex A_J along the top path), we assume that each two line segments $\overline{A_{i-1}A_i}$ and $\overline{A_iA_{i+1}}$ can never have the same slope (otherwise, these two line segments could have been combined to reduce the value of J by 1). Also note that the top path can be completely determined if we can find the $J+1$ vertices $A_0, A_1, \cdots, A_J$. Before introducing how to find these vertices, we need the following notations. Let $\theta_1 = \mathrm{Angle}(\overrightarrow{A_0A_1}) = \tan^{-1}(\text{Slope } \overrightarrow{A_0A_1})$, $\theta_2 = \mathrm{Angle}(\overrightarrow{A_1A_2}) = \tan^{-1}(\text{Slope } \overrightarrow{A_1A_2}), \cdots, \theta_J = \mathrm{Angle}(\overrightarrow{A_{J-1}A_J}) = \tan^{-1}(\text{Slope } \overrightarrow{A_{J-1}A_J})$ be the directional angles of each (directed) line segments of the top path. The range of the angle function is the range of the arc-tangent function, namely, $-90^\circ < \theta < 90^\circ$. Also, define $\theta_0 = 90^\circ$ and $\theta_{J+1} = -90^\circ$ for convenience. We can use the convex property of the CH region, and the fact that the top path is

Manuscript received November 21, 1994; revised June 13, 1997. This work was supported by the National Science Council, Taiwan, R.O.C., under Contract NSC84-2213-E-009-042. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Rama Chellappa.

J.-C. Lin is with the Department of Computer and Information Science, National Chiao Tung University, Hsinchu 300, Taiwan, R.O.C. (e-mail: jclin@cis.nctu.edu.tw).

J.-Y. Lin is with the Electrical Engineering Department, Ming-Hsin Institute of Technology, Hsinchu Shiann, Taiwan, R.O.C.

Publisher Item Identifier S 1057-7149(98)03988-8.

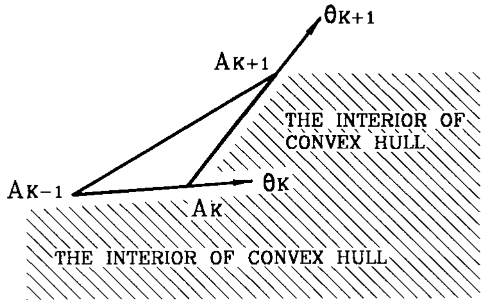

Fig. 2. For the top path, $\theta_k < \theta_{k+1}$ is impossible. Because if $\theta_k < \theta_{k+1}$, then the line segment $\overline{A_{k-1}A_{k+1}}$ is "above" the top path (the bald line segments), and hence, not contained in the convex hull, although A_{k-1} and A_{k+1} are two points of the convex hull.

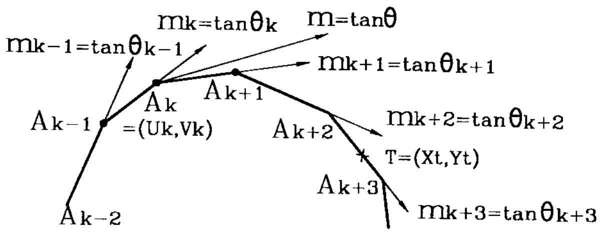

Fig. 3. Illustration describing Lemma 1.

the top boundary of the CH region to prove (see Fig. 2) that

$$\theta_0 > \theta_1 > \theta_2 > \cdots > \theta_J > \theta_{J+1}. \tag{1}$$

Theorem 1 below is fundamental to the theory of the proposed method. The proof of Theorem 1 uses Lemma 1. To save space, both proofs are omitted.

Lemma 1: Let $A_k = (u_k, v_k)$ be an arbitrary vertex of the top path, and θ an arbitrary angle satisfying $\theta_k > \theta > \theta_{k+1}$. Let $m = \tan\theta$, then $(y - mx)|_{A_k} = v_k - mu_k > (y - mx)|_T$ for any other point $T = (x_t, y_t)$ that is on the top path (see Fig. 3).

Theorem 1: Define A_k, θ, and m as in Lemma 1. If $(x_i, y_i) \in \{(x_n, y_n)\}_{n=1}^N$ is an arbitrary point in the input set, and $(x_i, y_i) \neq A_k$, then $(y - mx)|_{A_k} > y_i - mx_i$.

Because the set of all vertices of the CH is a subset of the input set (this well-known fact was used by many existing algorithms [8] to find the CH vertices); we may say that the A_k in Theorem 1 is one of the input points, and $(y - mx)|_{A_k}$ is the unique maximum among the N values $\{y_n - mx_n\}_{n=1}^N$. Also note that in Lemma 1 and Theorem 1, θ is not allowed to be identical to θ_k or θ_{k+1} for reasons explained below. If θ happens to coincide with one of the directional angles of the line segments that form the CH, say, $\theta = \theta_k$, then $(y - mx)|_{A_k} = (y - mx)|_{A_{k-1}}$ because $\overline{A_{k-1}A_k}$ is a line segment with slope $\tan\theta_k = \tan\theta = m$. In fact, we can also prove that if $\theta = \theta_k$, then all the input points in $\overline{A_{k-1}A_k}$, including A_{k-1} and A_k, will maximize $\{y_n - mx_n\}_{n=1}^N$, i.e., the point that maximizes $\{y_n - mx_n\}_{n=1}^N$ is no longer unique if $\theta = \theta_k$. The nonuniqueness is an undesired property for the following reason. Since any point $z \in \overline{A_{k-1}A_k}$ (z needs not be A_{k-1} or A_k) will maximize $\{y_n - mx_n\}_{n=1}^N$ when $m = \tan\theta = \tan\theta_k =$ Slope($\overline{A_{k-1}A_k}$), we cannot use this kind of θ to find the CH vertex if we plan to identify those input points that maximize $\{y_n - mx_n\}_{n=1}^N$ as the CH vertices in the top path. Otherwise, the middle point z of the line segment $\overline{A_{k-1}A_k}$ might be mistreated as a vertex, although z is not a vertex.

All the discussions made in the previous paragraph suggest that each "vertex" A_k of the top path can be found by identifying it as

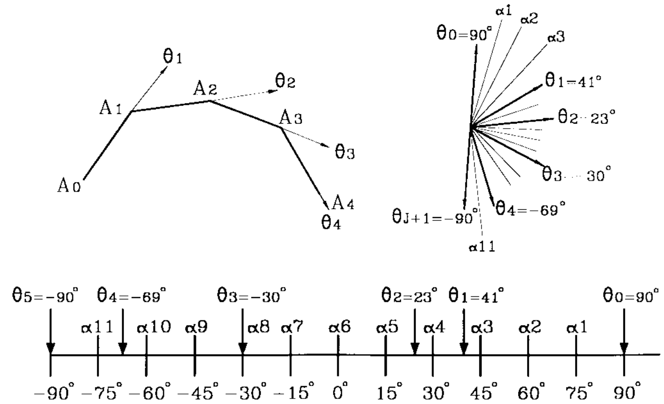

Fig. 4. Example that illustrates the application of Theorem 1. See the text above (2).

the "unique" input point that maximizes $\{y_n - mx_n\}_{n=1}^N$ for some $m = \tan\theta$ that is carefully chosen to meet the $\theta_k > \theta > \theta_{k+1}$ requirement stated in Lemma 1. Of course, finding this θ is not easy because θ_k and θ_{k+1} are not known in advance. Let us first inspect the example given below before we discuss how to find the suitable value of θ. Assume that the top path is $\overline{A_0A_1} \cup \overline{A_1A_2} \cup \overline{A_2A_3} \cup \overline{A_3A_4}$ (see Fig. 4, top left) with $\theta_1 = 41^\circ$, $\theta_2 = 23^\circ$, $\theta_3 = -30^\circ$, and $\theta_4 = -69^\circ$, respectively (see Fig. 4, top right). Therefore, $J = 4$. If we uniformly partition the angular interval $\{\alpha \mid -90^\circ \leq \alpha \leq 90^\circ\}$ into many subintervals so that the width $\Delta\alpha$ of each subinterval is, say, $\Delta\alpha = 15^\circ$, then we will have a set $W = \{\alpha_1 = 75^\circ, \alpha_2 = 60^\circ, \alpha_3 = 45^\circ, \cdots, \alpha_{10} = -60^\circ, \alpha_{11} = -75^\circ\}$ which is a collection of $I = (\frac{180^\circ}{\Delta\alpha}) - 1 = 11$ angles (see Fig. 4, bottom). Note that W has the following property:

$$(\theta_{j+1}, \theta_j) \cap W = \{\theta \mid \theta_{j+1} < \theta < \theta_j\} \cap W \neq \phi \quad \text{for all } j = 0, 1, \cdots, J. \tag{2}$$

In other words, for each open interval (θ_{j+1}, θ_j), where $j \in \{0, 1, 2, \cdots, J\}$, there exists at least one $\alpha_i \in W$ that satisfies $\theta_j > \alpha_i > \theta_{j+1}$.

If we compare the N values $\{y_n - x_n \tan\alpha\}_{n=1}^N$, where $\alpha = \alpha_1$, we find that a unique maximum value can be obtained at A_0 using Theorem 1. If α is set to α_2 or α_3, we can still obtain A_0. If α is set to α_4, however, we obtain A_1. If α is set to α_5, α_6 or α_7, we obtain A_2. If α is set to $\alpha_8 = -30^\circ = \theta_3$, a unique maximum is not obtained (any input point in the line segment $\overline{A_2A_3}$, including A_2 and A_3, will give maximum). We therefore bypass α_8 to avoid any of the ambiguity that leads to the problems discussed earlier. If α is set to α_9 or α_{10}, we obtain A_3. If α is set to α_{11}, we obtain A_4. Note that $A_0A_1A_2A_3A_4$ appear in the exact sequence needed to create the top path.

The remaining question is how to decide the value of $\Delta\alpha$ so that property (2) holds. Obviously, if $\theta_1 = 41^\circ$ and $\theta_2 = 40.9^\circ$, as in the previous example, then $\Delta\alpha = 15^\circ$ is not small enough to validate property (2). In order to find a universal $\Delta\alpha$ suitable for all given input data set when the resolution of the image board is fixed at, say, 512 × 512, we let $A = (u_a, v_a)$, $B = (u_b, v_b)$, and $C = (u_c, v_c)$ be any three points that are not on the same line, and try to find the smallest positive directional angle change

$$(\Delta\theta)_{\min} = \mathrm{Min}\{|\tan^{-1}(\text{Slope}(\overrightarrow{AB})) - \tan^{-1}(\text{Slope}(\overrightarrow{BC}))|\} \tag{3}$$

where u_a, v_a, u_b, v_b, u_c, and v_c range through $\{1, 2, 3, \cdots, 511, 512\}$. We use a computer to evaluate this $(\Delta\theta)_{\min}$ and list in Table I

TABLE I
$(\Delta\theta)_{\min}$ FOR SEVERAL COMMON IMAGE BOARDS

Resolution	$(\Delta\theta)_{min}$
128 × 128	0.007217914°
256 × 256	0.001776181°
512 × 512	0.000489519°
1024 × 1024	0.000109709°

TABLE II
$\Delta\alpha$ AND I FOR SEVERAL COMMON IMAGE BOARDS

Resolution	$\Delta\alpha$	I= no. of α_i in W.
128 × 128	0.007217900°	24937
256 × 256	0.001776163°	101341
512 × 512	0.000489518°	367707
1024 × 1024	0.000109708°	1640704

the $(\Delta\theta)_{\min}$ for several common resolutions. For example, when the resolution is 512 × 512, then $A = (1, 1)$, $B = (255,\ 256)$, $C = (510,\ 512)$ gives $\overrightarrow{AB} = \langle 254,\ 255\rangle$ and $\overrightarrow{BC} = \langle 255,\ 256\rangle$, which yields the smallest positive directional-angle-change $(\Delta\theta)_{\min} = \tan^{-1}\frac{255}{254} - \tan^{-1}\frac{256}{255} = 0.000\,489\,519°$. Note that the two vectors $\overrightarrow{AB}$ and $\overrightarrow{BC}$ are almost staying on the same line. If we let $\Delta\alpha$ be a little smaller than $(\Delta\theta)_{\min}$, for example, let

$$\Delta\alpha = \frac{180°}{\lfloor 180°/(\Delta\theta)_{\min}\rfloor + 1} < (\Delta\theta)_{\min} \tag{4}$$

and let

$$I = \frac{180°}{\Delta\alpha} - 1 = \lfloor 180°/(\Delta\theta)_{\min}\rfloor \tag{5}$$

then the W defined by $W = \{\alpha_i = 90° - i(\Delta\alpha)\}_{i=1}^{I}$ will satisfy property (2), because

$$\begin{aligned}\theta_j - \theta_{j+1} &= \text{Angle}(\overrightarrow{A_{j-1}A_j}) - \text{Angle}(\overrightarrow{A_jA_{j+1}})\\ &\geq (\Delta\theta)_{\min} > \Delta\alpha\end{aligned}$$

by the definitions of $(\Delta\theta)_{\min}$ and $\Delta\alpha$. (The logic here is simple: if there is no $\alpha_i \in W$ that satisfies $\theta_j > \alpha_i > \theta_{j+1}$, then there must be two adjacent α, say, α_5 and α_6, such that $\alpha_5 \geq \theta_j > \theta_{j+1} \geq \alpha_6$. It follows that $\Delta\alpha = \alpha_5 - \alpha_6 \geq \theta_j - \theta_{j+1} \geq (\Delta\theta)_{\min}$, which is a contradiction to the fact that $(\Delta\theta)_{\min} > \Delta\alpha$.) Note that setting $\Delta\alpha$ a little smaller than $(\Delta\theta)_{\min}$ will guarantee property (2), whereas setting $\Delta\alpha$ to be identical with $(\Delta\theta)_{\min}$ would render Theorem 1 inapplicable. We therefore use the $\Delta\alpha$ defined in (4) hereafter. The corresponding $\Delta\alpha$ and I are listed in Table II. Note that if the image board is rectangular instead of square, say, 512 × 480 instead of 512 × 512, then we can either derive $(\Delta\theta)_{\min}$, and hence $\Delta\alpha$, by a computer and the definitions (3) and (4), or, we may just use the $\Delta\alpha$ of the 512 × 512 resolution provided in Table II, because $\{(x, y) \mid x = 1, 2, \cdots, 512$ and $y = 1, 2, \cdots, 480\}$ is a subset of $\{(x, y) \mid x = 1, 2, \cdots, 512$ and $y = 1, 2, \cdots, 512\}$.

III. USING THE PARALLEL CREW ALGORITHM TO FIND THE TOP PATH

The parallel CREW–PRAM algorithm to obtain the top path is as follows (assume that each bank has a copy of the input data set $\{(x_n, y_n)\}_{n=1}^{N}$).

Step 1: Read the image board resolution and use Table II to determine the corresponding $\Delta\alpha$ and I.

Step 2: For each bank $i = 1, 2, \cdots, I$, do:

- Set $m = \tan(90° - i(\Delta\alpha))$.
- Set (a_i, b_i) to the unique (x_n, y_n) that maximizes $\{y_n - mx_n\}_{n=1}^{N}$.
- If more than one (x_n, y_n) has this maximal property, then set (a_i, b_i) to (∞, ∞) to mean "bypass this bank because of nonuniqueness."

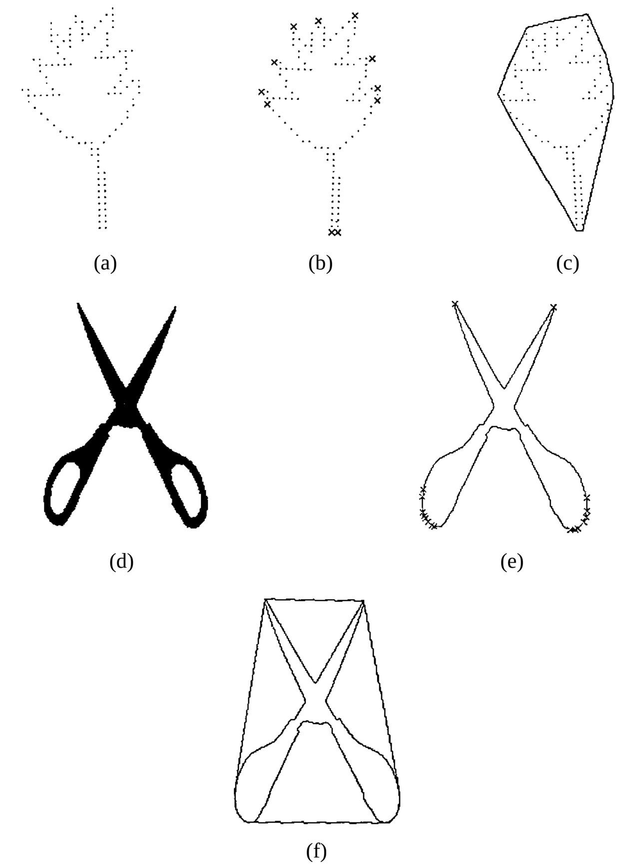

Fig. 5. Experimental results. (a) Leaf shape. (b) Detected vertices (marked by X). (c) Detected convex hull. (d) Scissors shape. (e) Detected vertices (marked by X). (f) Detected convex hull.

Step 3: For each bank $i = 2, 3, \cdots, I$ that generates an $(a_i, b_i) \neq (\infty, \infty)$, we output the (a_i, b_i) only if $(a_i, b_i) \neq (a_{i-1}, b_{i-1})$. (Remark: Bank 1 always output (a_1, b_1) if $(a_1, b_1) \neq (\infty, \infty)$.)

Note that most of the banks will not make any report because Step 3 is used to cancel out the redundancy when adjacent banks find the same vertex on the top path. Also note that the communication requirements between the adjacent banks are very simple (see Step 3). As for the computation time, we assume that each bank is equipped with $N/2$ processors, then the comparisons made in Step 2 need $1 \log N$ time-units. The time used in Steps 1 and 3 are constants. We therefore consider our algorithm a fast parallel algorithm which uses $1 \log N$ processing time and $I \times N/2$ processors. Here, I is the number of banks, and is proportional to the resolutions of the image board used, whereas N is the number of input points.

Although we have only shown how to find the top path, the bottom path can be found (simultaneously) in an identical way using the same $\Delta\alpha$ and I. The only difference is that minimization instead of the maximization is used in the algorithm. Indeed, the top and bottom paths give the expected convex hull. To convince the reader, the authors have simulated the proposed method using a personal computer. Many experiments were done, two of which are shown in Fig. 5. (Fig. 5(a) and (d) are the two input point sets.) Note that the scissors in Fig. 5(d) is a real photo image.

IV. Conclusion

A fast parallel algorithm that used simple operations was proposed to detect the CH of the N given points of the image plane. Several mathematical properties were provided to support the proposed idea. An illustrative example describing the idea and two experimental examples showing the simulation results were included in Sections II and III, respectively. The processing time is $1 \log N$, instead of $O(\log N)$, which is usually $C \times \log N$ with $C > 1$ described in many papers. Increased speed is achieved at the expense of using many processors. Note that the number of processors can be reduced further, because in most real-world applications, we seldom need a CH whose two consecutive edges differ in their directional angles by less than, say, 0.5°. Therefore, instead of using the small values listed in Table II for $\Delta\alpha$, we can just use $\Delta\alpha$ =0.5°, and, hence, $I = (180/0.5) - 1 = 359$ banks of processors are enough.

Also note that we did not use either the divide-and-conquer or data-sorting technique, although these two techniques are commonly employed in most of the existing algorithms that are used to find CH. Moreover, each processor only did very simple operations, and the flow of the data was also quite simple and obvious. Therefore, the VLSI design for implementing this special-purpose algorithm is quite easy.

References

[1] S. G. Akl and G. T. Toussaint, "Efficient convex hull algorithms for pattern recognition applications," in *Proc. 4th Int. Joint Conf. Pattern Recognition*, Kyoto, Japan, 1978, pp. 483–487.

[2] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[3] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*. New York: Springer-Verlag, 1985.

[4] A. Aggarwal *et al.*, "Parallel computational geometry," *Algorithmica*, vol. 3, pp. 293–327, 1988.

[5] M. J. Atallah and M. T. Goodrich, "Efficient parallel solutions to some geometric problems," *J. Parallel Distrib. Comput.*, vol. 3, pp. 492–507, 1986.

[6] R. Miller and Q. F. Stout, "Geometric algorithms for digitized pictures on a mesh-connected computer," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-7, pp. 216–228, 1985.

[7] F. T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes*. San Mateo, CA: Morgan Kaufmann, 1992.

[8] R. A. Jarvis, "On the identification of the convex hull of a finite set of points in the plane," *Inform. Processing Lett.*, vol. 2, pp. 18–21, 1973.

Texture Synthesis via a Noncausal Nonparametric Multiscale Markov Random Field

Rupert Paget and I. Dennis Longstaff

***Abstract*—Our noncausal, nonparametric, multiscale, Markov random field (MRF) model is capable of synthesizing and capturing the characteristics of a wide variety of textures, from the highly structured to the stochastic. We use a multiscale synthesis algorithm incorporating *local annealing* to obtain larger realizations of texture visually indistinguishable from the training texture.**

***Index Terms*—Local annealing, Markov random fields, multiresolution, nonparametric estimation, texture synthesis.**

I. Introduction

We present here a method of modeling texture that enables synthesis of texture visually indistinguishable from training textures. Our noncausal, nonparametric multiscale Markov random field (MRF) model captures the high-order statistical characteristics of textures. We propose that if a model is capable of synthesizing texture visually indistinguishable from its training texture, then it has captured *all* the visual characteristics of that texture and must therefore be unique to that particular texture. Given a set of unique statistical models for a set of training textures, it may then be possible to use these models to segment and classify textures in images that contain many textures, including unmodeled textures. Classification could be achieved by using these unique statistical models to determine the statistical similarity of a region in the image to a training texture. Any region where there was no statistical similarity could then be labeled as an *unknown texture*.

The conventional approach to classifying texture is to choose the most discriminatory features from the training textures via a feature selection process such as linear discriminatory analysis [17]. However, when a new texture is added to the training set, the features selected may *not* be those appropriate for distinguishing the new texture from the previously modeled textures. Therefore, for each new texture, the selection process has to be repeated to obtain a new set of discriminatory features. Another limitation of conventional models is they cannot be applied to images containing textures other than those in the training sets. Therefore, they cannot be used to classify complex images such as synthetic aperture radar (SAR) images of Earth's terrain, which contain a myriad of textures.

Current texture models such as fractal models, auto-models, autoregressive models, moving average models, and autoregressive moving average models, do not realistically reproduce natural textures [9], such as those in the Brodatz album [4]. This implies the models do not capture *all* the visual characteristics. Julesz [11] hypothesized that third- or higher-order models were required to model natural textures. The MRF model has the required statistical order [1], but the parametric versions are inherently inaccurate for modeling high-order statistical characteristics over a data-sparse multidimensional feature

Manuscript received April 3, 1996; revised August 4, 1997. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Maria Petrou.

The authors are with the Department of Electrical and Computer Engineering, and the Cooperative Research Centre for Sensor, Signal, and Information Processing, University of Queensland, Brisbane, QLD 4072, Australia (e-mail: paget@elec.uq.edu.au; idl@elec.uq.edu.au).

Publisher Item Identifier S 1057-7149(98)03987-6.