
PERGAMON

Pattern Recognition 36 (2003) 1783–1797

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Ant colony search algorithms for optimal polygonal approximation of plane curves

Peng-Yeng Yin*

Department of Computer Science and Information Engineering, Ming Chuan University, 5 Teh-Ming Road, Taoyuan 333, Taiwan

Received 29 June 2000; received in revised form 11 July 2002; accepted 19 September 2002

Abstract

This paper presents a new polygonal approximation method using ant colony search algorithm. The problem is represented by a directed graph such that the objective of the original problem becomes to find the shortest closed circuit on the graph under the problem-specific constraints. A number of artificial ants are distributed on the graph and communicate with one another through the pheromone trails which are a form of the long-term memory guiding the future exploration of the graph. The important properties of the proposed method are thoroughly investigated. The performance of the proposed method as compared to those of the genetic-based and the tabu search-based approaches is very promising.

Keywords: Polygonal approximation; Genetic algorithm; Tabu search; Ant colony search algorithm; Local optimal solution; Global optimal solution

1. Introduction

Planar curve approximation is a very important topic because digital curves often appear as region boundaries and object contours in an image. Since the main information of the digital curves is preserved at the corner points which have the maximal measure of significance, e.g., the curvature, in their neighborhood, it is desired to approximate a curve with the corner points to reduce the memory storage and the processing time for subsequent procedures. A polygonal approximation technique is one of the approaches which can accomplish this task and has attracted the attention of many researchers. The idea behind is to approximate a given curve by an "optimal polygon" with the minimal number of line segments such that the approximation error between the original curve and the corresponding line segments is no more than a pre-specified tolerance.

Most existing methods provide local optimal approximation results due to limited computational time resources. They can be divided into three groups: (1) sequential approaches, (2) split-and-merge approaches, and (3) dominant point-detection approaches. For sequential approaches, Sklansky and Gonzales [1] proposed a scan-along procedure which starts from a point and tries to find the longest line segments sequentially. Kurozumi and Davis [2] proposed a minimax method which derives the approximating segments by minimizing the maximum distance between a given set of points and the corresponding segment. Ray and Ray [3] proposed a method which determines the longest possible line segments with the minimum possible error. Most of the sequential approaches are simple and fast, but the quality of their approximating results is dependent on the location of the point where they start the scan-along process. For split-and-merge approaches, Ramer [4] presented a recursive method starting with an initial boundary segmentation. At each iteration, the segment is split at the point which has the farthest distance from the corresponding segment unless the approximation error is no more than the pre-specified error tolerance. Ansari and Delp [5] proposed another

* Tel.: +886-33507001x3415; fax: +886-33349889.
E-mail address: pyyin@mcu.edu.tw (P.-Y. Yin).

0031-3203/03/$30.00
PII: S0031-3203(02)00321-7

technique which first uses Guassian smoothing to smooth the boundary and then takes the maximal curvature points as break points to which the split-and-merge process is applied. Ray and Ray [6] proposed an orientation-invariant and scale-invariant method by introducing the use of ranks of points and the normalized distances. The approximation results of the split-and-merge approaches may be far from the optimal one if a poor initial segmentation is used. For dominant point-detection approaches, Teh and Chin [7] determine the region of support for each point based on its local property to evaluate the curvature. The dominant points are then detected by a nonmaxima suppression process. Wu and Wang [8] combined the corner detection and the iterative partition methods to detect the dominant points. Held et al. [9] proposed a two-stage method. In the first stage, a coarse-to-fine smoothing scheme is applied to detect dominant points. Then, in the second stage, a hierarchical approximation is produced by a criterion of perceptual significance. The performance of most dominant point-detection methods depends on the accuracy of the curvature evaluation.

All of the local optimal methods are very fast but their results may be very far from the optimal one. However, an exhaustive search for the vertices of the optimal polygon from the given set of data points will result in an exponential complexity. Dunham [10] and Sato [11] used dynamic programming to find the optimal approximating polygon. However, when the starting point is not specified, these methods require a worst-case complexity of $O(n^4)$ where n is the number of data points. Fortunately, there exist some global search heuristics which can output solutions very close to the global optimal one in a relative short time. Approaches based on genetic algorithms (GA) [12,13] and tabu search (TS) [14] have been proposed to solve the polygonal approximation problem and they obtain better results than most of those due to the local optimal methods. In this paper, we develop a more effective and efficient global search algorithm based on a heuristic called the ant colony search (ACS) method [15]. The properties of the proposed algorithm are thoroughly analyzed, and the approximation results are encouraging as compared to those of the works using GA and TS.

The remainder of this paper is organized as follows. Section 2 reviews the basic principles of the ACS algorithm. Section 3 renders the details of the proposed method. In Section 4, we present the experimental results and the discussions. Finally, a summary is given in Section 5.

2. ACS algorithms

The ACS algorithm was proposed by Dorigo [15] in 1992, and had been used to solve many complex problems successfully such as the traveling salesman problem [16] quadratic assignment problem [17], and combined heat and power economic dispatch problem [18]. The ACS is inspired by the research on the real ant behavior. Ethologists observed that ants can construct the shortest path from their colony to the feeding source and back through the use of pheromone trails. An ant leaves some quantities of pheromone on the ground and marks the path by a trail of this substance. The next ant will sense the pheromone laid on different paths and choose one with a probability proportional to the amount of pheromone on it. The ant then traverses the chosen path and leaves its own pheromone. This is an autocatalytic (positive feedback) process which favors the path along which more ants previously traversed.

The general principles for the ACS simulation of real ant behavior are as follows.

(1) *Initialization*. The initialization of the ACS includes two parts: the problem graph representation and the initial ant distribution. *First*, the underlying problem should be represented in terms of a graph, $G = \langle N, E\rangle$, where N denotes the set of nodes, and E the set of edges. The graph is connected, but not necessarily complete, such that the feasible solutions to the original problem correspond to paths on the graph which satisfy problem-domain constraints. *Second*, a number of ants are arbitrarily placed on the nodes chosen randomly. Then each of the distributed ants will perform a tour on the graph by constructing a path according to the node transition rule described next.

(2) *Node transition rule*. The ants move from node to node based on a node transition rule. According to the problem-domain constraints, some nodes could be marked as inaccessible for a walking ant. For example, in the traveling salesman problem, we can mark the nodes which an ant has visited as inaccessible to let the ant visit every node of the graph in finite steps. The node transition rule is probabilistic. For the kth ant on node i, the selection of the next node j to follow is according to the node transition probability,

$$p_{ij}^{k} = \begin{cases} \dfrac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{h \notin tabu_k}(\tau_{ih})^{\alpha}(\eta_{ih})^{\beta}} & \text{if } j \notin tabu_k, \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where τ_{ij} is the intensity of pheromone laid on edge (i,j), η_{ij} is the value of visibility of edge (i,j), α and β are control parameters, and $tabu_k$ means the set of currently inaccessible nodes for the kth ant according to the problem-domain constraints. The intensity of pheromone laid on edge (i,j) reflecting the previous experiences of the ants about this edge is a shared memory which provides indirect communication between the ants, i.e., the ants "communicate" with one another by sensing the strength of pheromone laid on edges. Pheromone information provides a global view about the selection of the edge based on a quality measure of the solution constructed afterward. On the other hand, the value of visibility is determined by a greedy heuristic for the original problem which considers only the local information on edge (i,j) such as the length of it. Parameters α and β control the relative contribution between the two types of

measures mentioned above. Therefore, the node transition rule given by Eq. (1) is a trade-off between the search of intensification and diversification.

(3) *Pheromone updating rule*. The ant keeps walking through edges to different nodes by iteratively applying the node transition rule until a solution to the original problem is constructed. For example, in the traveling salesman problem, a solution is obtained when an ant has visited every node in its tour. We define that a cycle of the ACS algorithm is completed when every ant has constructed a solution. At the end of each cycle, the intensity of pheromone trails on each edge is updated by the pheromone updating rule,

$$\tau_{ij} \leftarrow \rho\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}, \tag{2}$$

where $\rho \in (0,1)$ is the persistence rate of previous trails, $\Delta\tau_{ij}^{k}$ is the amount of pheromone laid on edge (i,j) by the kth ant at the current cycle, and m is the number of distributed ants. In a real ant system, shorter paths will retain more quantities of pheromone; analogously, in the ACS, the paths corresponding to fitter solutions should receive more pheromone quantities and become more attractive in the next cycle. Hence, if we define by f_k the fitness value of the solution found by the kth ant that is inverse proportional to some cost of the constructed path, then $\Delta\tau_{ij}^{k}$ can be given by

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{f_k}{Q} & \text{if edge } (i,j) \text{ is traversed by the } k\text{th ant at the current cycle,} \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where Q is a constant.

(4) *Stopping criterion*. The stopping criterion of the ACS algorithm could be the maximal number of running cycles, the CPU time limit, or the maximal number of cycles between two improvements of the global best solution.

It is interesting that Wagner et al. [19,20] also proposed a series of trace-oriented algorithms called the Ant-Robots. In the algorithm, the ants walk along the edges of a graph, leave pheromone traces at edges, and use those traces to guide their exploration of the graph. Ant-Robots is synonymous with ACS in those mechanisms such as they both use traces as means of navigation and indirect communication between ants. But the two algorithms still differ in the following aspects. *First*, each time an edge is traversed, Ant-Robots immediately overrides the previous trace on the edge, while ACS defers the trace update until every ant has completed its tour on the graph. *Second*, Ant-Robots uses a deterministic transition rule based on local information for determining the next node to move, while ACS applies a probabilistic transition rule based on measures of global solution quality and a local heuristic. There is no theoretical proof yet that either one is superior to the other. In the next section, we focus our discussion on the ACS paradigm and propose the proper modifications that enable ACS to solve the polygonal approximation problem.

3. The proposed method

In this section, we first give the problem definition of a polygonal approximation, and then describe how we modify the ACS algorithm and apply it to the problem.

3.1. Problem definition

Given a curve in an image with n_1 clockwise-ordered points, denoted $S = \{x_1, x_2, \ldots, x_{n_1}\}$. The objective of the polygonal approximation problem is to find the minimal ordered subset $S^* = \{x_{v(1)}, x_{v(2)}, \ldots, x_{v(n_2)}\}$, where $v(i) \in [1, n_1]$ is a monotonically increasing function for $1 \leqslant i \leqslant n_2$ and $n_2 \leqslant n_1$, and the set of the n_2 line segments, $P = \{\overline{x_{v(1)}x_{v(2)}}, \overline{x_{v(2)}x_{v(3)}}, \ldots, \overline{x_{v(n_2-1)}x_{v(n_2)}}, \overline{x_{v(n_2)}x_{v(1)}}\}$, composes an approximating polygon to the point set S such that the error norm between S and P is no more than a pre-specified tolerance ε (we shall refer to this condition as the ε-bound constraint). For convenience of presentation, we consider S as a circular list (x_1 is considered as the succeeding point of x_{n_1}) and define arc $\widehat{x_i x_j}$ as the collection of points x_i, x_j and all the intermediate points, i.e., $\widehat{x_i x_j} = \{x_i, x_{i+1}, \ldots, x_j\}$. Fig. 1 gives an illustrative example where arc $\widehat{x_i x_j}$ is the collection of six points $\{x_i, x_{i+1}, x_{i+2}, x_{i+3}, x_{i+4}, x_j\}$. The error norm between S and P, denoted $E(S,P)$, is then defined as the sum of the approximation errors between the n_2 arcs $\{\widehat{x_{v(1)}x_{v(2)}}, \widehat{x_{v(2)}x_{v(3)}}, \ldots, \widehat{x_{v(n_2-1)}x_{v(n_2)}}, \widehat{x_{v(n_2)}x_{v(1)}}\}$ and the corresponding n_2 line segments $\{\overline{x_{v(1)}x_{v(2)}}, \overline{x_{v(2)}x_{v(3)}}, \ldots, \overline{x_{v(n_2-1)}x_{v(n_2)}}, \overline{x_{v(n_2)}x_{v(1)}}\}$. We have

$$E(S,P) = \sum_{i=1}^{n_2} e(x_{v(i)}\widehat{x_{v(i+1)}}, \overline{x_{v(i)}x_{v(i+1)}}), \tag{4}$$

where $v(n_2+1) \equiv v(1)$ and $e(x_{v(i)}\widehat{x_{v(i+1)}}, \overline{x_{v(i)}x_{v(i+1)}})$ is the approximation error between arc $x_{v(i)}\widehat{x_{v(i+1)}}$ and segment $\overline{x_{v(i)}x_{v(i+1)}}$. The value of $e(x_{v(i)}\widehat{x_{v(i+1)}}, \overline{x_{v(i)}x_{v(i+1)}})$ can be estimated in a number of ways. Here, we adopt the integral square error (ISE), i.e., the sum of the squared perpendicular distance from every point on the arc to the corresponding line segment. Thus, we have

$$e(x_{v(i)}\widehat{x_{v(i+1)}}, \overline{x_{v(i)}x_{v(i+1)}}) = \sum_{x_j \in x_{v(i)}\widehat{x_{v(i+1)}}} d^2(x_j, \overline{x_{v(i)}x_{v(i+1)}}), \tag{5}$$

where $d(x_j, \overline{x_{v(i)}x_{v(i+1)}})$ is the perpendicular distance from point x_j to the corresponding line segment $\overline{x_{v(i)}x_{v(i+1)}}$. For example, as Fig. 1 shows, the value of $e(\widehat{x_i x_j}, \overline{x_i x_j})$ is computed as $d^2(x_i, \overline{x_i x_j}) + d^2(x_{i+1}, \overline{x_i x_j}) + d^2(x_{i+2}, \overline{x_i x_j}) + d^2(x_{i+3}, \overline{x_i x_j}) + d^2(x_{i+4}, \overline{x_i x_j}) + d^2(x_j, \overline{x_i x_j})$.

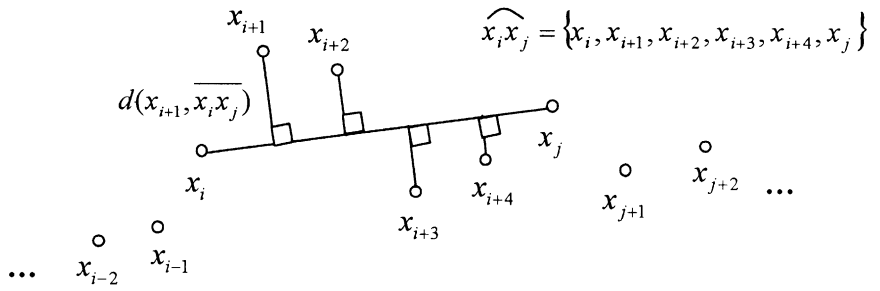

Fig. 1. Illustration of arc $\widehat{x_i x_j}$ and computation of $e(\widehat{x_i x_j}, \overline{x_i x_j})$.

3.2. Graph representation

To apply the ACS, the underlying problem should be represented in terms of a graph, $G = \langle N, E \rangle$. Apparently, for the polygonal approximation problem, each point on the curve should be represented as a node of the graph, i.e., $N = S$. An ideal edge set E^* will be the one which has the desired property that any closed circuit which originates and ends at the same node represents a feasible solution to the original problem, i.e., the approximating polygon consisting of the edges and nodes along the closed circuit should satisfy the ε-bound constraint. However, it is very hard to generate an ideal edge set, or in most cases the ideal edge set does not exist. A more practical way is to generate a pseudo-ideal edge set $\hat{E}$, such that, $E^* \subseteq \hat{E}$. For those circuits which do not satisfy the ε-bound constraint, we can decrease the intensity of pheromone trails on the circuits to make them less attractive. The way we generate $\hat{E}$ is as follows. First, an empty edge set is created, i.e., $\hat{E} = \emptyset$. For every node $x_i \in S$, we examine each of the remaining nodes, $x_j \in S$, in clockwise order. The directed edge $\vec{x_i x_j}$ is added to $\hat{E}$ if the approximation error between the arc $\widehat{x_i x_j}$ and the line segment $\overline{x_i x_j}$ is no more than ε, i.e.,

$$\hat{E} = \{\vec{x_i x_j} \,|e(\widehat{x_i x_j}, \overline{x_i x_j}) \leqslant \varepsilon\}. \tag{6}$$

The reason for using a directed edge is to avoid the ants walking backward.

Now, the problem of polygonal approximation is equivalent to finding the shortest closed circuit on the directed graph $G = \langle S, \hat{E} \rangle$ subject to the ε-bound constraint. For the convenience of presentation, we define some notations as follows. Let the closed circuit completed by the kth ant be denoted $tour_k$, the number of nodes visited in $tour_k$ be $|tour_k|$, and the approximation error between the original curve S and the approximating polygon corresponding to $tour_k$, be $E(S, tour_k)$.

T	1	1	1	1	1	1	1	1	1	1
(a)	1	2	3	4	5	6	7	8	9	10
T	1	1	0.123	1	0.15	1	1	1	1	1
(b)	1	2	3	4	5	6	7	8	9	10

Fig. 2. An example showing the selection table of the starting node where $n_1 = 10$, $m = 3$, and $r = 0.5$. (a) At the beginning of the first cycle, and (b) at the end of the first cycle.

3.3. Initial ant distribution

In Dorigo's work [16], where the traveling salesman problem is addressed, it does not matter which nodes the ants are initially placed at because each ant should visit all the nodes to complete a tour. However, in the polygonal approximation problem where the goal is to find the shortest closed tour, we prefer to place the ants at the nodes with a higher probability of finding such a tour. We thus establish a selection table for the starting node which is a linear array of n_1 entries denoted by T_i, $i = 1, 2, \ldots, n_1$. Initially, we let each $T_i = 1$. The probability with which the ith node is chosen as a starting node, denoted $Select_i$, is estimated as the entry value T_i normalized by the sum of all entry values,

$$Select_i = \frac{T_i}{\sum_{j=1}^{n_1} T_j}. \tag{7}$$

The ties with respect to $Select_i$ are broken randomly. Apparently, at the beginning of the first cycle, every node has equal probability of being chosen as a starting node since $Select_i = 1/n_1$ for $1 \leqslant i \leqslant n_1$. We then update the entry value of the selection table at the end of each cycle. Let the set of ants which start with the ith node at the current cycle be Ant_Start_i, and the size of Ant_Start_i be $|Ant\_Start_i|$. We update entry T_i based on a trade-off between the average quality of current solutions constructed by those ants in Ant_Start_i and the value of $Select_i$ derived from older cycles. Thus, we let

$$T_i \leftarrow \begin{cases} \dfrac{(1-r)}{|Ant\_Start_i|} \displaystyle\sum_{j \in Ant\_Start_i} \frac{1}{|tour_j|} + r Select_i & \text{if the } i\text{th node was chosen as a starting node at current cycle} \\ T_i & \text{otherwise,} \end{cases} \tag{8}$$

where $r \in (0,1)$ is the parameter which controls the relative contribution of each component. In particular, if we let $r=0$ then T_i is updated using the experience of current cycle, and if we let $r=1$ then T_i is set to $Select_i$ that corresponds to experiences of older cycles. Fig. 2 shows an example of the selection table where $n_1 = 10$. At the beginning of the first cycle, each entry is assigned value 1 (see Fig. 2(a)), so every node has the same selection probability of 0.1 (see Eq. (7)). Assume three ants (i.e., $m=3$) are used in the system and are denoted ant_1, ant_2, and ant_3, respectively. Let ant_1 and ant_2 start their tours with the third node, and ant_3 with the fifth node. At the end of this cycle, let ant_1 and ant_2 obtain tours with $|tour_1| = 6$ and $|tour_2| = 8$, and let ant_3 obtain a tour with $|tour_3| = 5$. If we set $r = 0.5$ in Eq. (8), then the corresponding entries are updated by

$$T_3 = \frac{0.5}{2}\left(\frac{1}{6}+\frac{1}{8}\right) + 0.5(0.1) = 0.123$$

and

$$T_5 = \frac{0.5}{1}\left(\frac{1}{5}\right) + 0.5(0.1) = 0.15$$

as shown in Fig. 2(b). Hence, the fifth node will have a higher selection probability than that of the third node in subsequent cycles. As such, in the early cycles, the ants will prefer to choose the nodes which have not been chosen yet as starting nodes and thus enforce an exploration search to new regions. After all the nodes have been given a try, the ants will be prone to choose the starting nodes which have more experiences of constructing shorter tours and enforce an exploitation search to the neighborhood of better solutions.

3.4. Node transition rule

The node transition rule is a probabilistic one determined by the pheromone intensity τ_{ij} and the visibility value η_{ij} of the corresponding edge. In the proposed method, τ_{ij} is equally initialized to $1/n_1$ (actually, any small constant positive value will suffice), and is gradually updated at the end of each cycle according to the average quality of the solutions that involve this edge. On the other hand, the value of η_{ij} is determined by a greedy heuristic which encourages the ants to walk to the farthest accessible node in order to construct the longest possible line segment in a hope that an approximating polygon with fewer vertices is obtained eventually. This can be accomplished by setting $\eta_{ij} = |\widehat{x_i x_j}|$, where $|\widehat{x_i x_j}|$ is the number of points on $\widehat{x_i x_j}$. The value of η_{ij} is fixed during all the cycles since it considers local information only.

We now define the transition probability from node i to node j through directed edge $\vec{x_i x_j}$ as

$$p_{ij} = \frac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{\substack{\forall \vec{x_i x_h} \\ \text{from } x_i}} (\tau_{ih})^{\alpha}(\eta_{ih})^{\beta}}. \tag{9}$$

Also, the ties with respect to p_{ij} are broken randomly. There is no need to keep a tabu list as compared to Eq. (1) since the inaccessible nodes have been avoided by the use of directed edges.

3.5. Pheromone updating rule

The intensity of pheromone trails of an edge is updated at the end of each cycle by the average quality of the solutions that traverse along this edge. If it is guaranteed that any completed tour on the graph corresponds to a feasible solution to the original problem, then one can simply apply Eqs. (2) and (3) to update pheromone intensity. However, it is very difficult to generate an ideal graph that satisfies this strong requirement; instead, it is more practical to generate a pseudo-ideal graph and then decrease the pheromone intensity if an infeasible solution is yielded. Consequently, we propose the following pheromone updating rule. At the end of each cycle, the pheromone intensity at directed edge $\vec{x_i x_j}$ is updated by

$$\tau_{ij} \leftarrow \rho\tau_{ij} + \max\left(\sum_{k=1}^{m} \Delta\tau_{ij}^{k}, 0\right), \tag{10}$$

where $\rho \in (0,1)$ is the persistence rate of previous pheromone trails, and $\Delta\tau_{ij}^{k}$ is the quantity of new trails left by the kth ant and it is computed by

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{1}{|tour_k|} & \text{if } \vec{x_i x_j} \in tour_k \\ & \text{and } E(S, tour_k) \leqslant \varepsilon; \\ -\dfrac{E(S, tour_k)}{\varepsilon n_1}, & \text{if } \vec{x_i x_j} \in tour_k \\ & \text{and } E(S, tour_k) > \varepsilon; \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

Therefore, more quantities of pheromone trails will be laid at the edges along which most ants have constructed shorter feasible tours. On the other hand, in the worst case, the edges will receive no positive rewards because either no ants walked through them or most passing ants constructed infeasible tours. As such, the proposed rule can guide the ants to explore better tours corresponding to high quality solutions.

3.6. The proposed algorithm and the best parameter setting

We summarize the proposed algorithm (denoted ACS/Poly) in Fig. 3. To obtain the best performance of the ACS/Poly, various parameter values have been experimented with. The test values are $\alpha \in \{0,1,2,3,4,5,6\}$, $\beta \in \{0,1,2,3,4,5,6\}$, $r \in \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$, $\rho \in \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$, and $m \in \{1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50\}$. Three synthesized benchmark curves (Figs. 9–11) which are broadly used in the literature [3, 6, 7, 9, 12–14] and two real image curves (Figs. 12 and 13) are used for the experiments. Fig. 9(a) is a leaf curve with 120 points,

Input.

$S = \{x_1, x_2, \ldots, x_{n_1}\}$: a set of clockwise-ordered points along the given curve.

m: the number of ants.

MAX_CYCLE: the maximal number of running cycles.

1. Initialize.
   Construct the directed graph $G =< S, \hat{E} >$ as described in Subsection 3.2.
   Set $\tau_{ij} = 1/n_1$ for every directed edge $\overrightarrow{x_i x_j}$.
   Set $T_i = 1$ for $1 \le i \le n_1$.
   Set $NC = 1$.
   Set $tour_{global\_best} = \{\overrightarrow{x_1 x_2}, \overrightarrow{x_2 x_3}, \ldots, \overrightarrow{x_{n_1-1} x_{n_1}}, \overrightarrow{x_{n_1} x_1}\}$.
2. For every ant do
   Select a starting node according to the selection probability computed by Eq. (7).
   Repeat
      Move to next node according to the node transition rule using Eq. (9).
   until a closed tour is completed.
   // a closed tour is completed when the ant arrives at the starting node again //
3. Find out the shortest feasible tour, say $tour_{current\_best}$, among the m tours obtained at Step 2.
4. If $|tour_{current\_best}| < |tour_{global\_best}|$ then $tour_{global\_best} = tour_{current\_best}$
5. For every directed edge $\overrightarrow{x_i x_j}$ do
   Update the pheromone intensity using Eqs. (10) and (11).
6. For every selection table entry T_i do
      Update the entry value using Eq. (8).
7. If $(NC = MAX\_CYCLE)$, then output $tour_{global\_best}$ and stop;
   otherwise, $NC = NC + 1$ and goto Step 2.

Fig. 3. The ACS/Poly algorithm.

Fig. 10(a) is a chromosome curve with 60 points, Fig. 11(a) is a semi-circle curve with 102 points, Fig. 12(a) is a fish contour image with 700 edge points, and Fig. 13(a) is a plane contour image with 682 edge points. The best performance of the ACS/Poly was found when we set $\alpha = 1$, $\beta = 5$, $r = 0.4$, $\rho = 0.1$, and $m = 20$. The observations on the test of extreme parameter values can tell us about some properties of the ACS/Poly. *First*, when either $\alpha = 0$ or $\beta = 0$, the performance is deteriorated rapidly. In the case of $\alpha = 0$, the ACS/Poly is reduced to multiple greedy heuristics (see Eq. (9)). The ants just ignore the global information represented as pheromone trails and do not know about the quality of the previously established tours, so there is no communication happening between the ants. In the case of $\beta = 0$, the ACS/Poly becomes an exploitation search method which focuses on a small neighborhood of the best-so-far solution and has a very low probability of exploring new regions of the solution space. *Second*, the performance of the ACS/Poly is devastated when either $r = 0$ or $\rho = 0$. Both of which mean that we should memorize a proper proportion of the experiences of order cycles (long-term memory), instead of only using the information obtained at current cycle (short-term memory). *Third*, the performance is improved as the number of ants increases, and the improvement becomes insignificant when the number of ants is greater than 20. The optimal number of ants seems to be close to the number of vertices of the finally-obtained approximating polygon, this is possibly because that the most important information is exchanged between those ants that chose the vertices of the optimal polygon as their starting nodes. Since the computational time of the ACS/Poly grows linearly with the number of ants used, the number of ants is set to 20 to reach the maximal synergism.

4. Experimental results and discussions

In this section, we discuss more important properties of the ACS/Poly through empirical studies. Two advanced strategies, the elitist strategy and the hybrid strategy, are given. The performance of various strategies of the ACS/Poly is compared to those of some existing methods. For all experiments, the best parameter settings for the ACS/Poly algorithm are employed.

4.1. Properties of the proposed algorithm

The important properties of the ACS/Poly are empirically examined in this subsection. The experiments are conducted on the leaf curve given in Fig. 9(a).

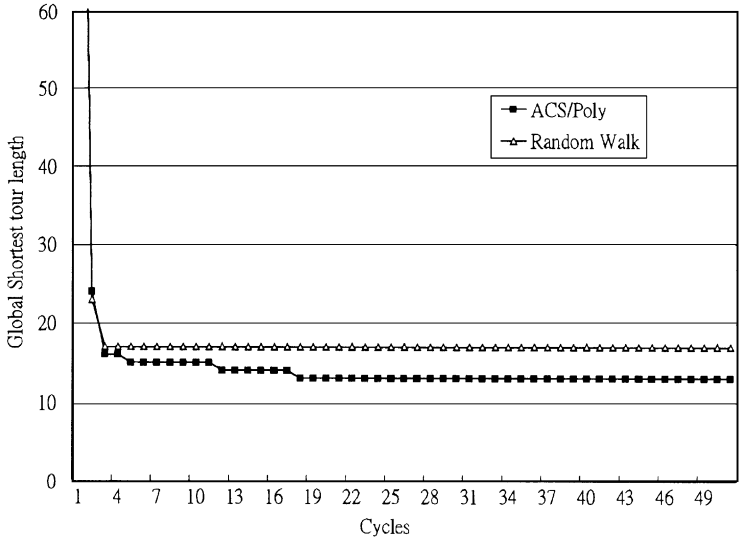

Fig. 4. The global shortest tour length versus the number of running cycles for ACS/Poly and random walk, for a typical run on the leaf curve with an ε-bound of 90.

4.1.1. ACS/Poly versus random walk

Since the stochastic nature of the ACS/Poly, we would like to see what its advantage is over a naive random walk method. We have implemented a random walk search method (referred to as random walk for simplicity) from the ACS/Poly by setting the probability distributions of $Select_i$ and p_{ij} to uniform distributions in all running cycles. Thus, in the random walk, the distributed ants have no communications in-between and no memory of previous experiences. The starting node and the next node are simply chosen randomly. Fig. 4 shows the length of the global shortest tour (inverse proportional to the approximation performance) versus the number of running cycles for both of the ACS/Poly and the random walk, for a specific run on the leaf curve with an ε-bound of 90. It is seen that at the beginning stage of the running cycles, the ACS/Poly which does not yet cumulate enough experiences has a similar performance to that of the random walk. After the fourth cycle, the global shortest tour length found by the ACS/Poly keeps decreasing, while the one found by the random walk is almost unchanged. Obviously, the mechanisms facilitating the inter-ant communication (global information) and the persistence rate of previous experiences (long-term memory) play significant roles in the ACS/Poly search paradigm.

4.1.2. Node branch entropy

To examine the convergence behavior of the ACS/Poly, we can measure the average node branch entropy. For the ith node on the graph, the node branch entropy is computed by measuring the information purity of the transition probabilities associated with all the directed edges which originate from the ith node. That is,

$$E_i = -\sum_{\substack{\forall \overrightarrow{x_i x_j} \\ \text{from } x_i}} p_{ij} \log p_{ij}. \tag{12}$$

The node transition rule becomes more deterministic when the node branch entropy approaches 0 (By L'hospital's rule, if $p_{ij} \to 0$ then $\lim_{p_{ij}\to 0} p_{ij} \log p_{ij} = 0$). We then compute the average entropy $\bar{E}$ over all node branch entropy values, i.e., $\bar{E} = \sum_{i=1}^{n_1} E_i / n_1$. Fig. 5 shows the value of $\bar{E}$ at each cycle for a typical run on the leaf curve with an ε-bound of 20. Initially, the value of $\bar{E}$ decreases gradually since there are few experiences cumulated and the ACS/Poly tries to explore new branches to gain more experiences. In the middle cycles (between the fortieth and the hundredth cycles), the value of $\bar{E}$ drops drastically because some edges become more preferable and thus have higher transition probabilities. In the later cycles, the value of $\bar{E}$ decreases gradually again since the dominant edges stand out and the transition probabilities become stable. Hence, the value of the maximal number of running cycles which decides the stopping criterion of the ACS/Poly could be set to one falling in the stable period. Note that if we let $\alpha > \beta$, the value of $\bar{E}$ will decrease dramatically and approach 0 in the early cycles, since the pheromone trails will dominate the transition probability, which results in premature convergence, such that, no better solutions can be obtained thereafter.

4.1.3. Starting node selection probability

Unlike the ACS which places the ants randomly in the initialization step, the ACS/Poly places the ants on the nodes

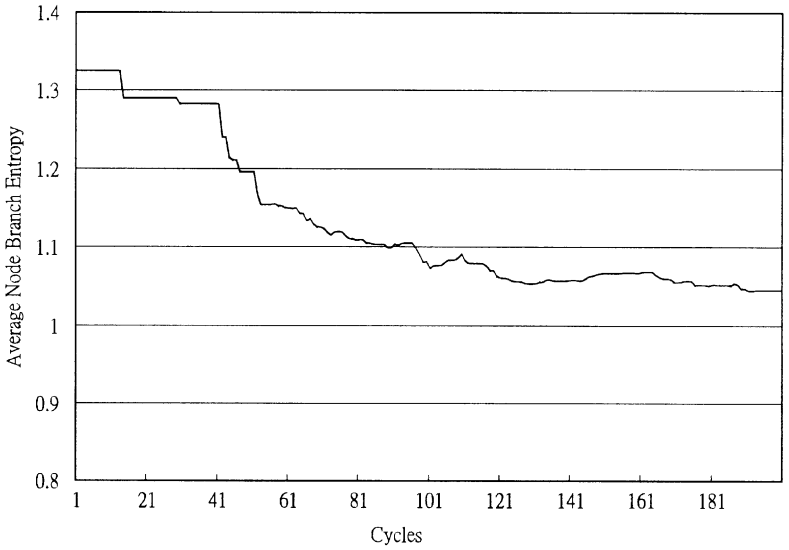

Fig. 5. The average node branch entropy versus the number of running cycles, for a typical run on the leaf curve with an ε-bound of 20.

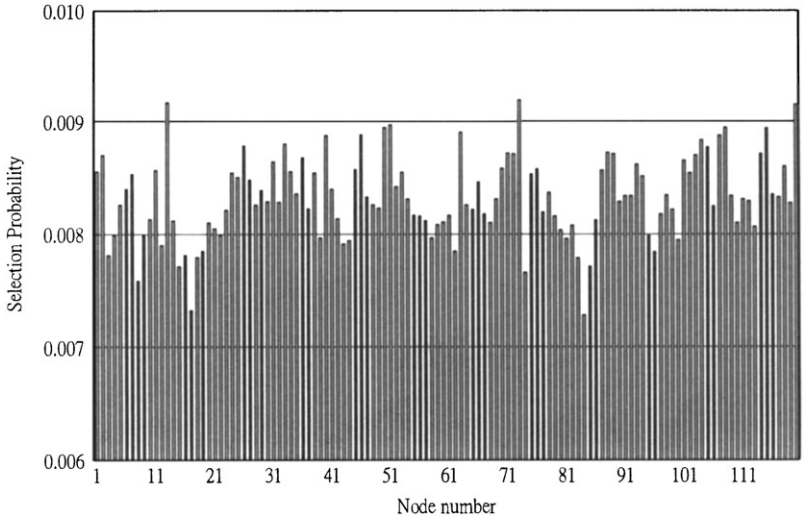

Fig. 6. The selection probability of each node at the final cycle, for a typical run on the leaf curve with an ε-bound of 20.

according to the selection probability. Initially, each node has an equal probability of being chosen as a starting node. As the positive experience is cumulated, the starting nodes which have more experiences of constructing shorter feasible tours will attain higher selection probabilities. Fig. 6 shows the selection probabilities of each node at the final cycle, for a typical run on the leaf curve with an ε-bound of 20. It is observed that some nodes have higher selection probability than the others. Also, since the edges along the shorter tours originating from these starting nodes have gained more quantities of pheromone, the ants starting with these nodes will intensively traverse the edges along the shorter tours which means that in the later cycles the ants exploit the neighborhood of good solutions by exchanging information between the shorter tours emanating from the starting nodes with high selection probabilities.

4.2. Searching strategy

In this subsection, we present two advanced searching strategies for the ACS/Poly.

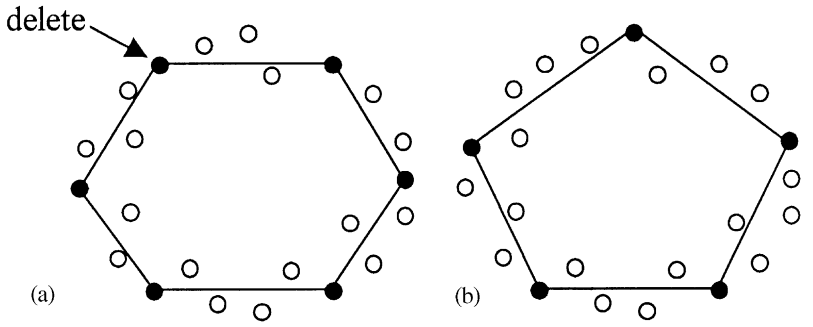

Fig. 7. Illustration of the local search method. (a) Randomly delete one node, and (b) approximation error-reduction by replacing remaining nodes with other data points.

4.2.1. Elitist strategy

The elitist strategy which comes from GA [21] is also suggested to be used in the ACS by Dorigo [16]. The basic idea is that the pheromone intensity on the best tour obtained so far (denoted $tour_{global\_best}$) is further reinforced such that $tour_{global\_best}$ has higher probability of exchanging edges with other tours. The elitist strategy of the ACS/Poly which shall be referred to as ELITIST is implemented as follows. In addition to updating the pheromone intensity of every edge according to the pheromone updating rule, the edges which compose $tour_{global\_best}$ are given extra quantities of pheromone by

$$\tau_{ij} \leftarrow \tau_{ij} + \frac{1}{|tour_{global\_best}|} \quad \text{if } \vec{x_i x_j} \in tour_{global\_best}. \tag{13}$$

The performance of the ELITIST will be compared with that of the next strategy.

4.2.2. Hybrid strategy

Every global search method combines two elements: exploration and exploitation. The exploration aspect searches for new regions, and once it finds a good region the exploitation part kicks in. However, since the two elements are usually inter-wound, the search is conducted to other regions before it finds the local optima. Hence, some researchers suggest to use a hybrid strategy which embeds a local search method in between the iterations of the global search method. For example, a complex of a GA and a simple hill-climbing method can improve the result obtained by applying a GA alone [21]. Inspired by this, we propose a hybrid strategy for the ACS/Poly which shall be referred to as HYBRID. The local search method used here is an iterative one, and for saving the computational time, it is applied to $tour_{global\_best}$ only. First, randomly delete one node from $tour_{global\_best}$. Then each of the remaining nodes of $tour_{global\_best}$ is examined in clockwise order and replaced by the data point which gives the minimal approximation error. Fig. 7(a) shows an example where $tour_{global\_best}$ involves six nodes (indicated by solid points), and one of them is randomly deleted (indicated by arrow). Each of the remaining five nodes is examined sequentially in clockwise order and is replaced by the data point which gives the minimal approximation error between the newly yielded line segments and corresponding arcs (see Fig. 7(b)). An iteration is completed when all of the remaining nodes of $tour_{global\_best}$ have been checked. The next iteration is then activated if any node has been replaced at the current iteration, i.e., there is an error reduction. The procedure is iterated at most five times to save computations. Once the procedure stopped, the final approximation error is checked as to whether it satisfies the ε-bound constraint. If the answer is yes, we can use this new $tour_{global\_best}$; otherwise, the old $tour_{global\_best}$ should be recovered. Since the local search method is applied only to $tour_{global\_best}$ (and not to every completed tour), the ELITIST algorithm should be applied thereafter to reinforce the pheromone intensity of $tour_{global\_best}$. The HYBRID algorithm is implemented by inserting the following steps after Step 6 of the ACS/Poly algorithm given in Fig. 3.

1. Randomly delete one node from $tour_{global\_best}$.
2. Repeat
   For each remaining node on
   $tour_{global\_best}$ do
   Replace the node with the data point which gives the minimal approximation error.
   until no node was replaced or
   the iteration has been repeated five times
3. If the new $tour_{global\_best}$ violates
   ε-bound constraint, then recover the old $tour_{global\_best}$.
4. Apply the ELITIST algorithm.

Fig. 8 shows the global shortest tour length versus the number of running cycles for the ACS/Poly, ELITIST, and HYBRID on the leaf curve with an ε-bound of 20. It can be seen that both the ELITIST and the HYBRID overcome the ACS/Poly, and that the HYBRID has the best performance. As will be seen in the next subsection, the two advanced algorithms cost negligible extra CPU time relatively to the ACS/Poly, and they provide more significant approximation improvement for smaller ε-bounds.

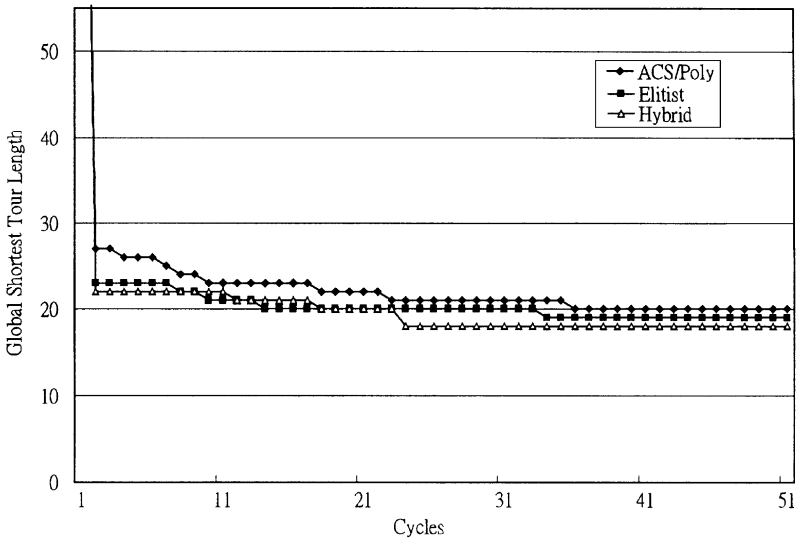

Fig. 8. The global shortest tour length versus the number of running cycles for various searching strategies, for a typical run on the leaf curve with an ε-bound of 20.

4.3. Comparison with GA and TS

In [12–14], global search approaches based on GA and TS have been shown to be superior to many existing methods due to the local search. Table 1 quotes the experimental results of some local search methods from Refs. [3,7,9] on the three synthesized benchmark curves where the approximation error (ε) and the number of vertices (n_2) of the approximating polygon are reported. The experimental results on the same curves using the GA-based [12] and the TS-based [14] approaches are also listed in Table 2. Since GA and TS are stochastic, the average results over 10 independent runs and the standard deviation (σ_{n_2}) of n_2 over these runs are reported. It is seen that the GA-based and the TS-based approaches can produce approximating polygon with fewer number of vertices than those obtained using the local search methods. Also, the standard deviation of n_2 yielded by the GA-based and the TS-based approaches are all less than one unit. However, GA-based and TS-based approaches need a little more computation time (from 0.5 to 5.7 s) to output those results.

Like GA and TS, the proposed algorithms are also global search heuristics, and we shall present extensive comparisons between these algorithms. The last three columns of Table 2 reports the average results over 10 independent runs for the ACS/Poly, ELITIST, and HYBRID. Since the starting node selection probability and the node transition probability used in these algorithms are proportional to $Select_i$ and p_{ij}, respectively, and each independent run may produce different result, we calculate the standard deviation of n_2 to realize the difference extent of these runs. Various values of ε have been specified for each of the benchmark curves. It is seen from Table 2 that the ACS/Poly, ELITIST, and HYBRID have better performance than those of the GA-based and the TS-based approaches. It is also observed that the ELITIST and the HYBRID have a more significant improvement in reducing the number of vertices than the ACS/Poly when the value of ε decreases, since the two advanced algorithms further intensify the search in the neighborhood of $tour_{global\_best}$. The average CPU cost time of the ELITIST is similar to that of the ACS/Poly because only a few computations are needed to update the pheromone trails of $tour_{global\_best}$. The extra CPU cost time of the HYBRID is also negligible if the user prefers to obtain a better approximation with a higher compression ratio. Fig. 9 shows the finally-obtained approximating polygon with its approximation error (ε) and number of vertices (n_2), for one specific run on the leaf curve for each of the competing methods. Fig. 10 corresponds to the chromosome curve, and Fig. 11 corresponds to the semi-circle curve. It can be seen that the proposed ACS/Poly, ELITIST, and HYBRID consistently outperform the other approaches in all experiments.

To demonstrate the feasibility of the proposed algorithms for real-world problems, two real image (see Figs. 12(a) and 13(a)) are further experimented with. As shown in Table 3, the proposed ACS/Poly, ELITIST, and HYBRID are still superior to the GA-based and the TS-based methods in both aspects of effectiveness (in terms of n_2) and efficiency (in terms of CPU time). The ELITIST and the HYBRID yield the most satisfactory results on the cost of negligible extra computational time. Fig. 12 shows the

Table 1
The approximation results on three benchmark curves using the local search methods

Curves	Teh–Chin		Ray–Ray		Held–Abe	
	ε	n_2	ε	n_2	ε	n_2
Leaf ($n_1 = 120$)	15.43	28	16.43	26	NA	NA
Chromosome ($n_1 = 60$)	6.4	16	7.67	14	NA	NA
Semicircle ($n_1 = 102$)	20.61	22	16.33	19	28.52	17

NA: not available.

Table 2
A comparative performance evaluation in terms of average number of vertices on the polygon (n_2) with its standard deviation (σ_{n_2}) and average CPU cost time (t, in seconds) of the GA-based approach, TS-based approach, and ACS/Poly approach (for both ELITIST and HYBRID) on the three benchmark curves

Curves	ε	GA-based		TS-based		ACS/Poly		ELITIST		HYBRID	
		$n_2(\sigma_{n_2})$	t	$n_2(\sigma_{n_2})$	t	$n_2(\sigma_{n_2})$	t	$n_2(\sigma_{n_2})$	t	$n_2(\sigma_{n_2})$	t
	150	15.6 (0.6)	5.7	10.6 (0.5)	0.9	11.2 (0.5)	0.7	11.6 (0.2)	0.7	11.0 (0.0)	0.9
	100	16.3 (0.5)	4.5	13.7 (0.6)	0.9	13.0 (0.3)	0.7	13.0 (0.3)	0.7	12.6 (0.2)	0.8
Leaf	90	17.3 (0.5)	5.3	14.6 (0.5)	0.9	13.2 (0.4)	0.7	13.0 (0.3)	0.7	12.8 (0.3)	0.9
($n_1 = 120$)	30	20.5 (0.6)	4.6	20.1 (0.5)	0.9	17.2 (0.4)	0.7	17.0 (0.3)	0.7	16.6 (0.4)	0.9
	15	23.8 (0.6)	5.4	23.1 (0.5)	0.9	22.2 (0.5)	0.7	20.9 (0.4)	0.7	19.7 (0.3)	0.9
	30	7.3 (0.4)	3.0	6.7 (0.4)	0.5	6.0 (0.0)	0.4	6.0 (0.0)	0.4	6.0 (0.0)	0.4
	20	9.0 (0.6)	3.0	8.0 (0.3)	0.5	8.0 (0.3)	0.4	8.0 (0.3)	0.4	7.6 (0.3)	0.5
Chromosome	10	10.2 (0.4)	3.1	11.0 (0.4)	0.5	10.0 (0.3)	0.4	10.0 (0.3)	0.4	10.0 (0.3)	0.5
($n_1 = 60$)	8	12.2 (0.5)	3.1	12.2 (0.5)	0.5	11.0 (0.4)	0.4	11.0 (0.4)	0.4	11.0 (0.4)	0.5
	6	15.2 (0.6)	3.3	14.4 (0.5)	0.5	12.8 (0.3)	0.4	12.4 (0.3)	0.4	12.2 (0.3)	0.5
	60	13.2 (0.4)	4.6	11.0 (0.4)	0.9	10.0 (0.0)	0.6	10.0 (0.0)	0.6	10.0 (0.0)	0.8
	30	13.9 (0.7)	4.8	13.6 (0.5)	0.8	12.6 (0.4)	0.6	12.4 (0.3)	0.6	12.0 (0.0)	0.8
Semicircle	25	16.8 (0.7)	4.3	14.9 (0.6)	0.8	13.4 (0.5)	0.6	13.0 (0.0)	0.6	13.0 (0.0)	0.7
($n_1 = 102$)	20	19.2 (0.6)	4.7	16.2 (0.6)	0.8	16.4 (0.5)	0.6	16.0 (0.2)	0.6	15.8 (0.4)	0.7
	15	23.0 (0.9)	4.4	18.3 (0.7)	0.8	18.0 (0.7)	0.6	17.4 (0.5)	0.6	16.8 (0.4)	0.7

yielded approximation polygon with its approximation error and number of vertices for each of the competing methods on the fish image, and Fig. 13 corresponds to those on the plane image. It is seen that the proposed algorithms also give better approximation results than existing methods on real images.

5. Summary

Polygonal approximation of curves is very important since it not only facilitates the reduction of memory storage and computational time but also provides a feature analysis of these curves. Most existing approaches are based on local search methods and can be classified to three classes: sequential approaches, split-and-merge approaches, and dominant point-detection approaches. Although they are computationally fast, the approximation results may be far from the global optimal one. Dynamic programming has been applied to produce the exact optimal answer. However, its computational complexity is not acceptable.

In this paper, we have proposed a new polygonal approximation method using a global search heuristic called the ant colony search (ACS) algorithm. Its major components, namely, graph representation, initial ant distribution, node transition rule, and pheromone updating rule have been investigated and adapted to the underlying problem. The important properties of the proposed method were thoroughly examined through empirical studies. Inspired by research on genetic algorithms, we have proposed the elitist strategy and the hybrid strategy for our method. The performances of the proposed methods were compared to those of genetic-based and tabu search-based methods. The numerical results found are very encouraging.

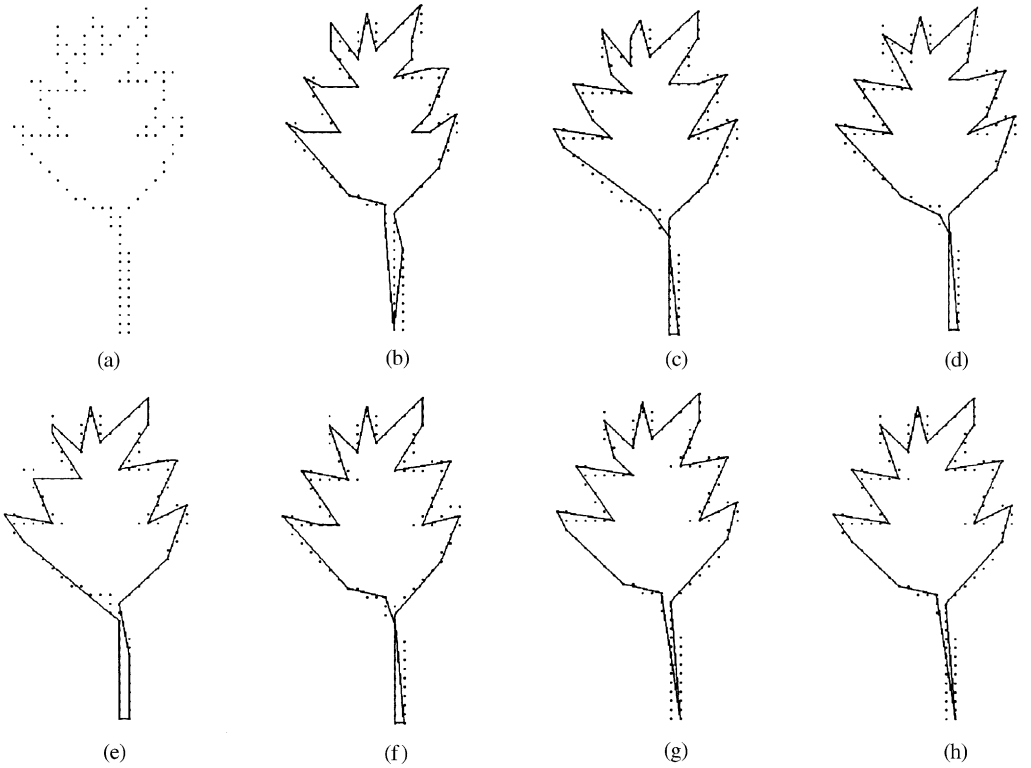

Fig. 9. The approximating polygon with its approximation error (ε) and number of vertices (n_2) obtained by the competing approaches on the leaf curve with $n_1 = 120$: (a) Leaf curve; (b) Teh–Chin ($\varepsilon = 15.43, n_2 = 28$); (c) Ray–Ray ($\varepsilon = 16.43, n_2 = 26$); (d) GA-based ($\varepsilon = 15, n_2 = 24$); (e) TS-based ($\varepsilon = 15, n_2 = 23$); (f) ACS/Poly ($\varepsilon = 15, n_2 = 22$); (g) ELITIST ($\varepsilon = 15, n_2 = 21$); (h) HYBRID ($\varepsilon = 15, n_2 = 20$).

Table 3
A comparative performance evaluation in terms of average number of vertices on the polygon (n_2) with its standard deviation (σ_{n_2}) and average CPU cost time (t, in seconds) of the GA-based approach, TS-based approach, and ACS/Poly approach (for both ELITIST and HYBRID) on the two real images

Curves	ε	GA-based		TS-based		ACS/Poly		ELITIST		HYBRID	
		n_2 (σ_{n_2})	t	n_2 (σ_{n_2})	t	n_2 (σ_{n_2})	t	n_2 (σ_{n_2})	t	n_2 (σ_{n_2})	t
Fish ($n_1 = 700$)	4000	16.5(0.5)	34.4	14.0 (0.3)	6.0	13.3 (0.5)	5.2	12.5 (0.5)	5.3	12.2 (0.4)	5.7
	3000	17.4 (0.6)	32.3	16.0 (0.3)	5.7	15.9 (0.7)	5.2	15.6 (0.3)	5.3	14.6 (0.3)	5.9
	2000	22.1 (1.0)	33.4	21.2 (0.4)	5.5	19.5 (0.5)	5.1	18.4 (0.6)	5.1	17.1 (0.5)	5.5
	1000	32.4 (0.9)	34.3	29.1 (1.0)	5.6	28.6 (0.6)	5.0	27.1 (0.5)	5.0	26.8 (0.7)	5.6
	500	37.0 (1.1)	36.3	35.9 (1.2)	5.4	36.1 (0.7)	4.9	34.9 (0.9)	5.1	34.8 (0.7)	5.6
Plane ($n_1 = 682$)	3000	14.2 (0.8)	37.2	13.0 (0.3)	5.3	12.6 (0.6)	4.8	12.3 (0.5)	4.8	12.1 (0.4)	5.0
	2000	15.1 (0.9)	36.3	14.4 (0.6)	5.3	13.4 (0.6)	4.4	13.3 (0.8)	4.6	13.0 (0.2)	4.7
	1000	17.4 (0.6)	34.4	16.7 (0.5)	5.0	16.3 (0.8)	4.5	15.8 (0.7)	4.6	14.0 (0.6)	4.8
	500	21.3 (0.8)	33.8	19.6 (0.6)	5.0	18.7 (0.5)	4.5	18.3 (0.2)	4.5	17.8 (0.5)	4.5
	100	33.8 (0.9)	35.9	31.3 (0.5)	5.0	29.4 (0.6)	4.1	28.6 (0.6)	4.2	28.1 (0.7)	4.6

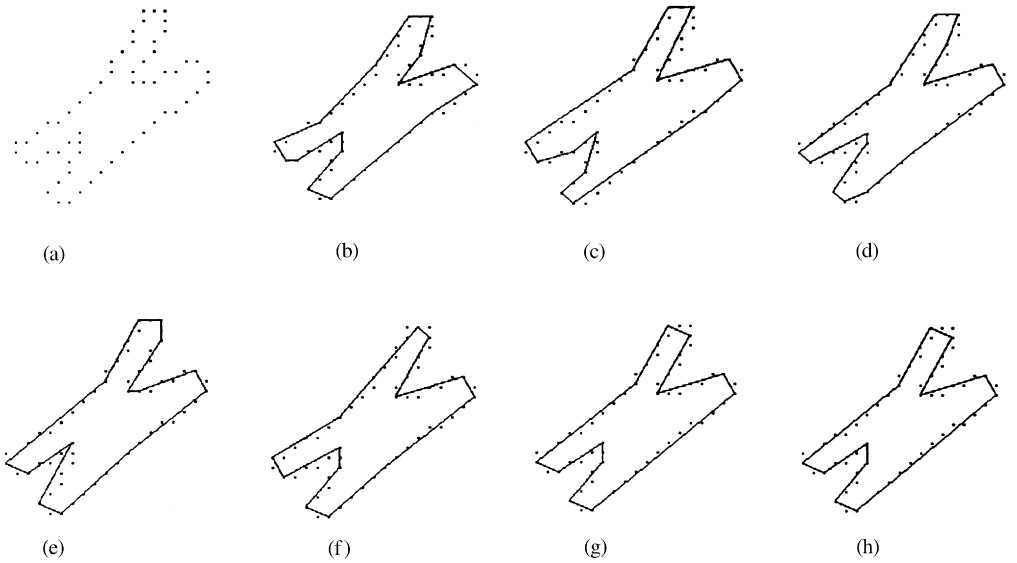

Fig. 10. The approximating polygon with its approximation error (ε) and number of vertices (n_2) obtained by the competing approaches on the chromosome curve with $n_1 = 60$: (a) Chromosome curve; (b) Teh–Chin ($\varepsilon = 6.4, n_2 = 16$); (c) Ray–Ray ($\varepsilon = 7.67, n_2 = 14$); (d) GA-based ($\varepsilon = 6, n_2 = 15$); (e) TS-based ($\varepsilon = 6, n_2 = 14$); (f) ACS/Poly ($\varepsilon = 6, n_2 = 13$); (g) ELITIST ($\varepsilon = 6, n_2 = 12$); (h) HYBRID ($\varepsilon = 6, n_2 = 12$).

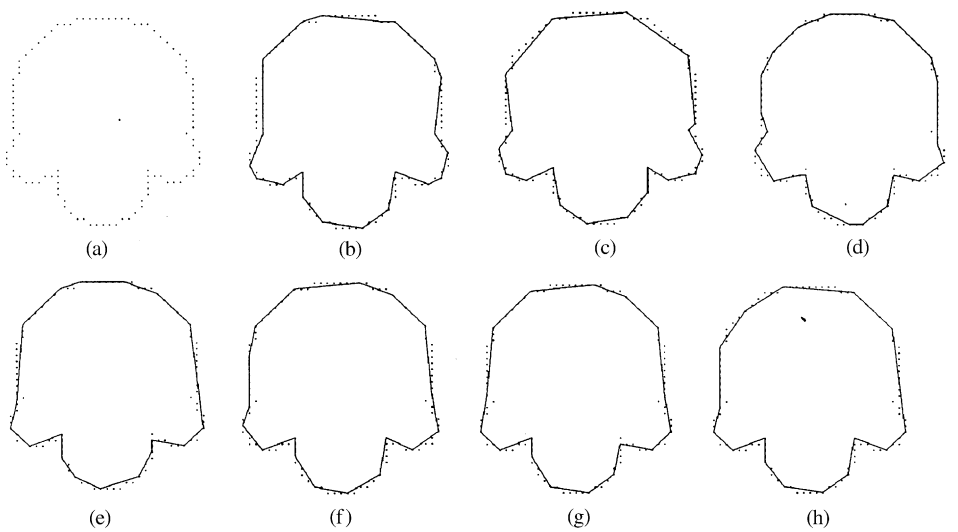

Fig. 11. The approximating polygon with its approximation error (ε) and number of vertices (n_2) obtained by the competing approaches on the semi-circle curve with $n_1 = 102$: (a) semi-circle curve; (b) Teh–Chin ($\varepsilon = 20.61, n_2 = 22$); (c) Ray–Ray ($\varepsilon = 16.33, n_2 = 19$); (d) GA-based ($\varepsilon = 15, n_2 = 23$); (e) TS-based ($\varepsilon = 15, n_2 = 19$); (f) ACS/Poly ($\varepsilon = 15, n_2 = 18$); (g) ELITIST ($\varepsilon = 15, n_2 = 17$); (h) HYBRID ($\varepsilon = 15, n_2 = 16$).

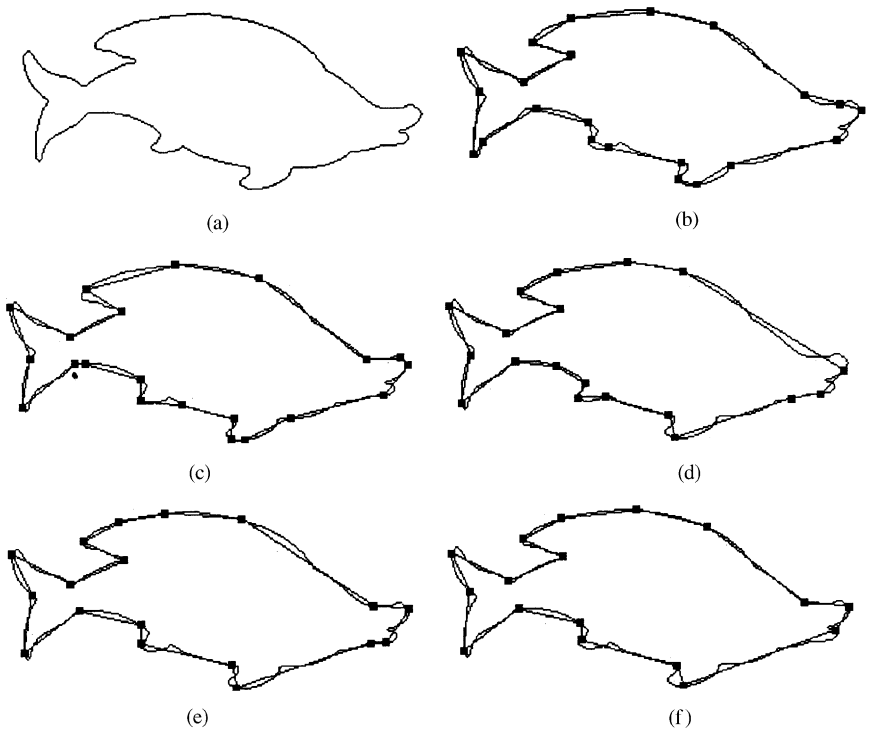

Fig. 12. The approximating polygon with its approximation error (ε) and number of vertices (n_2) obtained by the competing approaches on the fish image with $n_1 = 700$: (a) Fish contour image; (b) GA-based ($\varepsilon = 2000, n_2 = 22$); (c) TS-based ($\varepsilon = 2000, n_2 = 21$); (d) ACS/Poly ($\varepsilon = 2000, n_2 = 19$); (e) ELITIST ($\varepsilon = 2000, n_2 = 18$); (f) HYBRID ($\varepsilon = 2000, n_2 = 17$).

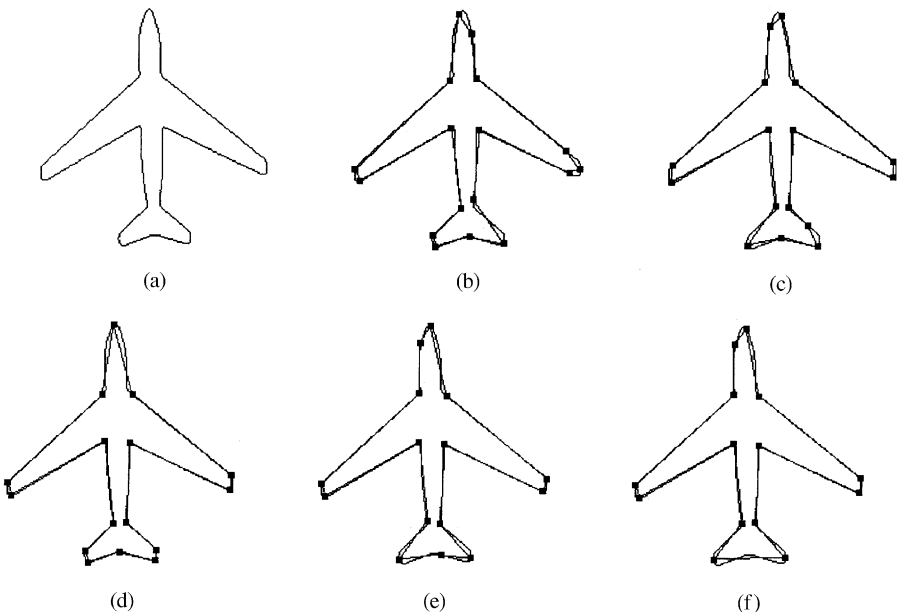

Fig. 13. The approximating polygon with its approximation error (ε) and number of vertices (n_2) obtained by the competing approaches on the plane image with $n_1 = 682$: (a) A plane contour image; (b) GA-based ($\varepsilon = 1000, n_2 = 17$); (c) TS-based ($\varepsilon = 1000, n_2 = 16$); (d) ACS/Poly ($\varepsilon = 1000, n_2 = 16$); (e) ELITIST ($\varepsilon = 1000, n_2 = 15$); (f) HYBRID ($\varepsilon = 1000, n_2 = 14$).

Acknowledgements

The author would like to thank the reviewers whose suggestions have improved the quality of the paper. This research is partially supported by National Science Council of ROC, under Grant NSC-90-2213-E-130-006.

References

[1] J. Sklansky, V. Gonzalez, Fast polygonal approximation of digitized curves, Pattern Recognition 12 (1980) 327–331.
[2] Y. Kurozumi, W.A. Davis, Polygonal approximation by the minimax method, Comput. Graphics Image Process. 19 (1982) 248–264.
[3] B.K. Ray, K.S. Ray, Determination of optimal polygon from digital curve using L_1 norm, Pattern Recognition 26 (1993) 505–509.
[4] U. Ramer, An iterative procedure for the polygonal approximation of plane curves, Comput. Graphics Image Process. 1 (1972) 244–256.
[5] N. Ansari, E.J. Delp, On detection dominant points, Pattern Recognition 24 (1991) 441–450.
[6] B.K. Ray, K.S. Ray, A new split-and-merge technique for polygonal approximation of chain coded curves, Pattern Recognition Lett. 16 (1995) 161–169.
[7] C.H. Teh, R.T. Chin, On the detection of dominant points on digital curves, IEEE Trans. Pattern Anal. Mach. Intell. 11 (1989) 859–872.
[8] W.Y. Wu, M.J. Wang, Detecting the dominant points by the curvature-based polygonal approximation, CVGIP: Graphical Models Image Process. 55 (1993) 79–88.
[9] A. Held, K. Abe, C. Arcelli, Towards a hierarchical contour description via dominant point detection, IEEE Trans. System Man Cybernet. 24 (1994) 942–949.
[10] J.G. Dunham, Optimum uniform piecewise linear approximation of planar curves, IEEE Trans. Pattern Anal. Mach. Intell. 8 (1986) 67–75.
[11] Y. Sato, Piecewise linear approximation of plane curves by perimeter optimization, Pattern Recognition 25 (1992) 1535–1543.
[12] P.Y. Yin, Genetic algorithms for polygonal approximation of digital curves, Int. J. Pattern Recognition Artif. Intell. 13 (1999) 1–22.
[13] S.C. Huang, Y.N. Sun, Polygonal approximation using genetic algorithms, Pattern Recognition 32 (1999) 1409–1420.
[14] P.Y. Yin, A tabu search approach to the polygonal approximation of digital curves, Int. J. Pattern Recognition Artif. Intell. 14 (2000) 243–255.
[15] M. Dorigo, Optimization, learning, and natural algorithms, Ph.D. Thesis, Dip. Elettronica e Informazione, Politecnico di Milano, Italy, 1992.
[16] M. Dorigo, L. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, IEEE Trans. Evolut. Comput. 1 (1997) 53–66.
[17] V. Maniezzo, A. Colorni, M. Dorigo, The ant system applied to the quadratic assignment problem, Universite Libre de Bruxelles, Belgium, Technical Report IRIDIA/94-28, 1994.
[18] Y.H. Song, C.S. Chou, T.J. Stonham, Combined heat and power economic dispatch by improved ant colony search algorithm, Electric Power Systems Res. 52 (1999) 115–121.
[19] I.A. Wagner, M. Lindenbaum, A.M. Bruckstein, Distributed covering by Ant-Robotics using evaporating traces, IEEE Trans. Robotics Automat. 15 (1999) 918–933.
[20] I.A. Wagner, M. Lindenbaum, A.M. Bruckstein, On-line graph searching by a smell-oriented vertex process, in: Proceedings of AAAI'97 Workshop On-Line Search, Providence, RI, 1997, pp. 122–125.
[21] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.

About the Author—PENG-YENG YIN received his B.S., M.S. and Ph.D. degrees in Computer Science from National Chiao Tung University, Hsinchu, Taiwan, in 1989, 1991 and 1994, respectively. From 1993 to 1994, he was a visiting scholar at the Department of Electrical Engineering, University of Maryland, and the Department of Radiology, Georgetown University. In 2000, he was a visiting Associate Professor in the Visualization and Intelligent System Lab (VISLab) at the Department of Electrical Engineering, University of California, Riverside (UCR). He is currently a Professor at the Department of Computer Science and Information Engineering, Ming Chuan University, Taoyuan, Taiwan. Dr. Yin received the Overseas Research Fellowship from Ministry of Education in 1993, Overseas Research Fellowship from National Science Council in 2000. He is a member of the Phi Tau Phi Scholastic Honor Society and listed in *Who's Who in the World*. His current research interests include image processing, pattern recognition, machine learning, computational biology, and evolutionary computation.