Applied Mathematics, **2012, 3, 1750-1762**
http://dx.doi.org/10.4236/am.2012.331241 Published Online November 2012 (http://www.SciRP.org/journal/am)

On the Detection of Visual Features from Digital Curves Using a Metaheuristic Approach

Cecilia Di Ruberto
Department of Mathematics and Computer Science, University of Cagliari, Cagliari, Italy
Email: dirubert@unica.it

Received May 28, 2012; revised June 27, 2012; accepted July 5, 2012

ABSTRACT

In computational shape analysis a crucial step consists in extracting meaningful features from digital curves. Dominant points are those points with curvature extreme on the curve that can suitably describe the curve both for visual perception and for recognition. Many approaches have been developed for detecting dominant points. In this paper we present a novel method that combines the dominant point detection and the ant colony optimization search. The method is inspired by the ant colony search (ACS) suggested by Yin in [1] but it results in a much more efficient and effective approximation algorithm. The excellent results have been compared both to works using an optimal search approach and to works based on exact approximation strategy.

Keywords: Digital Curve; Dominant Point; Polygonal Approximation; Ant Colony Optimization

1. Introduction

Computer imaging has developed as an interdisciplinary research field whose focus is on the acquisition and processing of visual information by computer and has been widely used in object recognition, image matching, target tracking, industrial dimensional inspection, monitoring tasks, etc. Among all aspects underlying visual information, the shape of the objects plays a special role. In computational shape analysis an important step is representation of the shape in a form suitable for storage and/or analysis. Due to their semantically rich nature, contours are one of the most commonly used shape descriptors, and various methods for representing the contours of 2D objects have been proposed, each achieves, more or less successfully, the most desirable features of a proper representation, such as data compression, simplicity of coding and decoding, scaling and invariance under rigid motions, etc. After shape representation, another crucial step in shape analysis consists in extracting meaningful features from digital curves. Attneave [2] pointed out that information on a curve is concentrated at the dominant points. Dominant points are those points that have curvature extreme on the curve and they can suitably describe the curve for both visual perception and recognition. Following Attneave's observation, there are many approaches developed for detecting dominant points. They can be classified into two main categories: corner detection approaches and polygonal approximation approaches. The phrases dominant point detection and polygonal approximation are used alternatively by some researchers in the area of pattern recognition. The polygonal approximation methods in fact construct polygon by connecting the detected dominant points. Although dominant points can constitute an approximating polygon, the polygonal approximation is different from the dominant point detection in concept. The polygonal approximation seeks to find a polygon that best fits the given digital curve. It can be applied to produce a simplified representation for storage purposes or further processing. Corner detection approaches aim to detect potential significant points, but they cannot represent smooth curve appropriately. For dominant point-detection approaches, Teh and Chin [3] determined 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 [4] proposed the curvature-based polygonal approximation for dominant point detection. It combines the corner detection and polygonal approximation methods, and it can detect the dominant points effectively. Wang *et al.* [5] proposed a simple method that uses the directions of the forward and backward vectors to find the bending value as the curvature. Held *et al.* [6] 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. Carmona *et al.* [7] proposed a new method for corner detection. A normalized measurement is used to compute the estimated

curvature and to detect dominant points, and an optimization procedure is proposed to eliminate collinear points. The performance of most dominant point-detection methods depends on the accuracy of the curvature evaluation. For polygonal approximation approaches, sequential, iterative, and optimal algorithms are commonly used. For sequential approaches, Sklansky and Gonzales [8] proposed a scan-along procedure which starts from a point and tries to find the longest line segments sequentially. Ray and Ray [9] 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 depends on the location of the point where they start the scan-along process and, as a consequence, they can miss important features. The family of iterative approaches splits and merges curves iteratively until they meet the preset allowances. For split-and-merge approaches, Ramer [10] estimated the distances from the points to the line segments of two ending points. The segment is partitioned at the point with the maximum distance until the maximum distance is not greater than an allowable value. Ansari and Delp [11] proposed another technique which first uses Gaussian 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 [12] proposed an orientation-invariant and scale-invariant method by introducing the use of ranks of points and the normalized distances. If a poor initial segmentation is used, the approximation results of the split-and-merge approaches maybe far from the optimal one. The iterative approaches, in fact, suffer the sensitivity to the selection of the starting points for partitioning curves. The optimal approaches tend to find the optimal polygonal approximation based on specified criteria and error bound constraints. 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 [1]. All of the local optimal methods are very fast but their results maybe very far from the optimal one. Unfortunately, an exhaustive search for the vertices of the optimal polygon from the given set of data points results in an exponential complexity. Dunham [13] and Sato [14] used dynamic programming to find the optimal approximating polygon. But, when the starting point is not specified, the method requires 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 find solutions very close to the global optimal one in a relative short time. Approaches based on genetic algorithms [15,16] and tabu search [17] have been proposed to solve the polygonal approximation problem and they obtain better results than most of the local optimal methods. Sun and Huang [15] presented a genetic algorithm for polygonal approximation. An optimal solution of dominant points can be found. However, it seems to be time consuming. Yin [17] focused in the computation efforts and proposed the tabu search technique to reduce the computational cost and memory in the polygonal approximation. Horng [18] proposed a dynamic programming approach to improve the fitting quality of polygonal approximation by combining the dominant point detection and the dynamic programming. Generally, the quality of the approximation result depends upon the initial condition where the heuristics take place and the metric used to measure the curvature. To solve complex combinatorial optimization problems metaheuristic techniques have been introduced. Fred Glover [19] first coined the term metaheuristic as a strategy that guides another heuristic to search beyond the local optimality such that the search will not get trapped in local optima. Metaheuristic techniques combine two components, an exploration strategy and an exploitation heuristic, in a framework. The exploration strategy searches for new regions, and once it finds a good region the exploitation heuristic further intensifies the search for this area. In this context, metaheuristics encompass several well-known approaches such as genetic algorithm (GA), simulated annealing, tabu search (TS), scatter search, ant colony optimization and particle swarm optimization. Most of the central metaheuristic methods have been applied to the polygonal approximation problems and attained promising results.

In this paper we present a method that combines the dominant point detection and the ant colony optimization search. The method is inspired by the ant colony search suggested by Yin in [1] but it results in a much more efficient and effective approximation algorithm. The excellent results have been compared both to works using an optimal search approach and to works based on exact approximation strategy. The ant colony optimization framework is presented in Section 2. The proposed method for detecting dominant points is illustrated in Section 3. In Section 4 the proposed algorithm is applied both on real world curves and on four well-known benchmark curves. The performance is compared visually and numerically with many existing methods. Conclusions are presented in Section 5.

2. Ant Colony System Algorithms

In [20] Dorigo first proposed the Ant Colony System (ACS) algorithm as a computational scheme inspired by the way in which real ant colonies operate. Ants make use of a substance called pheromone to communicate information about the shortest paths to reach the food.

An ant on the move leaves a certain amount of pheromone on the ground, creating a path formed by a trail of this substance. While an isolated ant moves practically at random, an ant encounters a trace left above. The ant is able to detect it and decide, most likely, to follow it, thus reinforcing the trail with its own pheromone. As a result, the more ants follow a trail, the more attractive the path becomes to follow and the probability with which an ant chooses a path increases with the number of ants that previously chose the same path (**Figure 1** [21]).

The ACS scheme is inspired by this process. We apply it to the problem of the polygonal approximation. The method we propose is similar to that described by Yin in [1] but it solves the problem in a much more efficient and effective way. In the ACS algorithm artificial ants build solutions (tours) of a problem moving from one node of a graph to another. The algorithm performs $N_{\max}$ iterations. During each iteration *m* ants construct a tour by performing *n* steps in which it is applied a probabilistic decision rule (transition state). In practice, if an ant in node *i* chooses to move to node *j*, the edge (i, j) is added to the tour in progress. This step is repeated until the ant has completed its tour. After all the ants have built their tours, each ant deposits some pheromone on the pheromone trail associated to the visited edges to make them most desirable for future ants. The amount of pheromone trail τ_{ij} associated with edge (i, j) represents the desirability of choosing node *j* from the node *i* and also represents the desirability that the edge (i, j) belongs to the tour built by an ant. The pheromone trail information is changed during solution of the problem to reflect the experience acquired by ants during solving the problem. Ants deposit an amount of pheromone proportional to the quality of the solutions they produced: the shorter the tour generated by an ant, the greater the amount of pheromone deposited on the edge used to generate the tour. This choice supports research directed towards good solutions. A pheromone evaporation is introduced to avoid stagnation, that is the situation where all ants choose the same tour [22]. Each ant has memory of the nodes already visited. The memory (or internal state) of each ant, called tabu list, is used to define, for each ant *k*, the set of nodes that an ant lying on the node *i*, has yet to visit. Exploiting the memory then, an ant *k* can build feasible solutions by generating a graph of the state space. The probability that an ant *k* chooses to move from node *i* to node *j* is defined as:

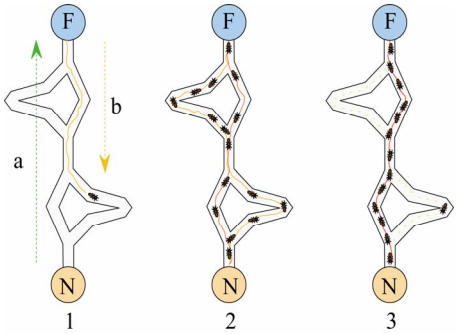

Figure 1. The first ant locates the food source (F), following one of the possible ways (a), then comes back to the nest (N), leaving behind a pheromone trail (b). Ants follow any way at random, but the reinforcement of the beaten path makes it more attractive as the shortest route. Ants take preferably this way while other paths lose their pheromone trails.

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

where τ_{ij} is the intensity of pheromone associated with edge (i, j), η_{ij} is the value of visibility of edge (i, j), α and β are control parameters, and tabu_k is the set of currently inaccessible nodes for the ant *k* according to the problem-domain constraints. The value of visibility is determined by a greedy heuristic for the initial problem, which considers only the local information on the edge (i, j), as its length. The role of the parameters α and β is the following. If $\alpha = 0$, closer nodes are more likely to be chosen: this corresponds to a classical stochastic greedy algorithm (with multiple starting points since ants are initially randomly distributed on the nodes). If, however, $\beta = 0$, the solution depends on the pheromone only: this case will lead to a rapid emergence of a stagnation situation with the corresponding generation of tours which are strongly sub-optimal. A trade-off between the heuristic value of track and intensity is therefore necessary.

After all ants have completed their tour, each ant k deposits a quantity of pheromone Δ_{ij}^{k} on each edge which has used:

$$\Delta_{ij}^{k} = \begin{cases} \dfrac{1}{|\text{tour}|_k} & \text{if } (i, j) \in \text{tour}_k \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where tour_k is the tour done by ant k at current cycle and $|\text{tour}|_k$ is its length. The value Δ_{ij}^{k} depends on how well the ant has worked: the shorter the tour done, the greater the amount of pheromone deposited. At the end of every cycle, the intensity of traces of pheromone on each edge is updated by the pheromone updating rule:

$$\tau_{ij} = \rho\tau_{ij} + \sum_{k=1}^{m} \Delta_{ij}^{k} \quad (3)$$

where $\rho \in (0,1)$ is the persistence rate of previous trails, Δ_{ij}^{k} is the amount of pheromone laid on edge (i, j) by the ant k at the current cycle, and m is the number of ants.

ACS algorithms have been applied to several discrete optimization problems, such as the travelling salesman problem and the quadratic assignment problem. Recent applications cover problems like vehicle routing, sequential ordering, graph coloring, routing in communication networks, and so on. In this work we propose the scheme of ACS to solve the problem of the polygonal approximation.

3. The ACS-Based Proposed Method

In this section we describe how we use the ACS algorithm to solve the problem of polygonal approximation. We first define our problem in terms of discrete optimization problem and how we represent it in terms of a graph. Then, we illustrate the proposed method from the initialization phase to the searching and refining the optimal path.

3.1. Problem Formulation

Given a digital curve represented by a set of *N* clockwise-ordered points, $C=\{c_1,c_2,\cdots,c_N\}$ where $c_{(i+1)\bmod N}$ is considered as the succeeding point of c_i, we define arc $\widehat{c_ic_j}$ as the collection of those points between c_i and c_j, and chord $\overline{c_ic_j}$ as the line segment connecting c_i and c_j. In approximating the arc $\widehat{c_ic_j}$ by the chord $\overline{c_ic_j}$, we make an error, denoted by $e\left(\widehat{c_ic_j},\overline{c_ic_j}\right)$ that can be measured by any distance norm; for here, the L_2 norm, *i.e.*, the sum of squared perpendicular distance from every data point on $\widehat{c_ic_j}$ to $\overline{c_ic_j}$, is adopted. Thus, we have

$$e\left(\widehat{c_ic_j},\overline{c_ic_j}\right)=\sum_{c_k\in\widehat{c_ic_j}} d^2\left(c_k,\overline{c_ic_j}\right) \tag{4}$$

where $d\left(c_k,\overline{c_ic_j}\right)$ is the perpendicular distance from point c_k to the corresponding chord $\overline{c_ic_j}$. The distance $d\left(c_k,\overline{c_ic_j}\right)$ is measured as follows:

$$d\left(c_k,\overline{c_ic_j}\right)=\frac{\left|\left(x_j-x_i\right)\times\left(y_i-y_k\right)-\left(x_i-x_k\right)\times\left(y_j-y_i\right)\right|}{\sqrt{\left(x_j-x_i\right)^2+\left(y_j-y_i\right)^2}} \tag{5}$$

where $(x_k,y_k),(x_i,y_i),(x_j,y_j)$ are the spatial coordinates of the points c_k,c_i,c_j, respectively.

The objective 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 less than a pre-specified tolerance. Formally, the aim is to find the minimal ordered set $T=\left\{c_{p_1},c_{p_2},\cdots,c_{p_M}\right\}$ where $T\subset C$ and $M\le N$, and the set of *M* line segments $P=\left\{\overline{c_{p_1}c_{p_2}},\overline{c_{p_2}c_{p_3}},\cdots,\overline{c_{p_{M-1}}c_{p_M}},\overline{c_{p_M}c_{p_1}}\right\}$ composes an approximating polygon to the point set *C* such that the error norm between *C* and *P* is no more than a pre-specified tolerance, ε. The error norm between *C* and *P*, denoted by $E_2(C,P)$, is then defined as the sum of the approximation errors between the *M* arcs $\left\{\widehat{c_{p_1}c_{p_2}},\widehat{c_{p_2}c_{p_3}},\cdots,\widehat{c_{p_{M-1}}c_{p_M}},\widehat{c_{p_M}c_{p_1}}\right\}$ and the corresponding *M* line segments $\left\{\overline{c_{p_1}c_{p_2}},\overline{c_{p_2}c_{p_3}},\cdots,\overline{c_{p_{M-1}}c_{p_M}},\overline{c_{p_M}c_{p_1}}\right\}$:

$$E_2(C,P)=\sum_{i=1}^{M} e\left(\widehat{c_{p_i}c_{p_{i+1}}},\overline{c_{p_i}c_{p_{i+1}}}\right) \tag{6}$$

where $c_{p_{M+1}}\equiv c_{p_1}$ and $e\left(\widehat{c_{p_i}c_{p_{i+1}}},\overline{c_{p_i}c_{p_{i+1}}}\right)$ is the approximation error between the arc $\widehat{c_{p_i}c_{p_{i+1}}}$ and the chord $\overline{c_{p_i}c_{p_{i+1}}}$, as defined in (4).

3.2. Graph Representation

In order to apply the ACS algorithm, we need to represent our problem in terms of a graph, $G=(V,E)$. For the polygonal approximation problem, each point on the curve should be represented as a node of the graph, *i.e.*, $V=C$, where *C* is the set of points on the given curve. *E* is an ideal edge set that 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, $E_2\le\varepsilon$. We construct the edge set by penalizing the intensity of pheromone trails on the paths to make them less attractive if they don't satisfy the ε-bound constraint. The edge set *E* is thus generated as follows. Initially, the set *E* is created as an empty set. Then, new edges are added to it. For every node $c_i\in C$, we examine each of the remaining nodes, $c_j\in C$, in clockwise order. If the approximation error between the arc $\widehat{c_ic_j}$ and the line segment $\overline{c_ic_j}$ is no more than ε, then the directed edge $\overleftarrow{c_ic_j}$ is added to the edge set *E*. Thus, we have:

$$E=\left\{\overleftarrow{c_ic_j}\,\middle|\,e\left(\widehat{c_ic_j},\overline{c_ic_j}\right)\le\varepsilon\right\}. \tag{7}$$

The edge is directed to avoid the ants walking backward. Then, the problem of polygonal approximation is equivalent to finding the shortest closed path on the directed graph $G=\langle V,E\rangle$ such that $E_2\le\varepsilon$. In the following, the closed path completed by the ant *k* will be

denoted by tour_k, the number of nodes visited by the ant k in the tour_k will be $|\text{tour}|_k$ and the approximation error between the original curve C and the approximating polygon corresponding to tour_k will be $E_2(C, \text{tour}_k)$.

3.3. Starting Node Initialization and Selection

During each iteration each ant chooses a starting node in the graph and sequentially constructs a closed path to finish its tour. In order to find the shortest closed tour, it is convenient to place the ants at the nodes with a higher probability of finding such a tour, instead of a randomly distribution. For the selection of the starting nodes we propose two alternative strategies, we call *Selection*_1 and *Selection*_2. Let's describe the two algorithms in detail.

***Selection*_1 algorithm**

One of the most common shape descriptors is the shape signature (or centroidal profile) [23]. A signature is one-dimensional functional representation of boundary, obtained by computing the distance of the boundary from the centroid as a function of angle (a centroidal profile). This simple descriptor is useful to understand the complexity of a shape. The more the extrema (maxima and minima) of the signature, the more articulated the object is. The number of signature extrema and the boundary points corresponding to them help us both to determine automatically the number of distributed ants and to select the nodes on the graph they choose to start their tour. In this algorithm the number m of ants to distribute on the graph is equal to the number of the signature extrema. Also, the m boundary points correspondent to the extrema represent the m starting nodes at the beginning of the first cycle. If $(s_1, s_2, \cdots, s_m)$ is the list of the boundary points relative to the extrema of the signature, then the ant k starts its first tour from the node s_k, $k = 1, \cdots, m$. The signature extrema are localized in the boundary points near to the most concave or convex portions of the curve. In such portions the most significant points (*i.e.* dominant points) of the curve are located. This is the reason for choosing the signature extrema as starting nodes at the first cycle. In the next cycles, the ants are placed at the nodes with higher probability to find the shortest closed tour. We thus create a selection list for the starting nodes, denoted by $T_i, i = 1, 2, \cdots, N$. Initially,

$$T_i = \begin{cases} 1 & \text{if } i \text{ is a starting node} \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

The probability with which the node i is chosen as a starting node in the next cycles, denoted Choose_i, is estimated as the value T_i normalized by the sum of all values,

$$\text{Choose}_i = \frac{T_i}{\sum_{j=1}^{N} T_j} \quad (9)$$

At the end of each cycle, the value of $T_i, i = 1, 2, \cdots, N$ is updated. Let's denote by $|\text{Start\_Node}_i|$ the set of ants which start with the node i at the current cycle and by $|\text{Start\_Node}_i|$ its size. We update the value T_i making a trade-off between the average quality of current solutions constructed by the ants in Start_Node_i and the value of Choose_i derived from older cycles. Thus, we let

$$T_i = \begin{cases} \dfrac{1}{|\text{Start\_Node}_i|} \displaystyle\sum_{j \in \text{Start\_Node}_i} \frac{1}{|\text{tour}_j|} + \text{Choose}_i \\ \qquad \text{if } i \text{ is a starting node at current cycle} \\ T_i \qquad \text{otherwise.} \end{cases} \quad (10)$$

During the process, the ants will tend to choose the starting nodes which have more experiences of constructing shorter tours and enforce an exploitation search to the neighborhood of better solutions.

***Selection*_2 algorithm**

For each node $i, i = 1, \cdots, N$ of the graph we evaluate the greatest approximation error among all the directed edges departing from *i*. We thus generate a list of the N greatest approximation errors sorted in ascending order. We select the first D edges, where D is a percentage on N (in all the experiments $D = 15$) and derive the nodes where these edges depart from. The list of such nodes, after eliminating duplications, represents the set of the starting nodes at the beginning of the first cycle and its size is the number m of the ants to distribute on the graph. As in the previous algorithm, we then create a selection list for the starting node $T_i, i = 1, 2, \cdots, N$, initialized as described in Equation (8). In the next iterations, the probability with which the node i is chosen as a starting node and the updating of the value T_i are accomplished as in the previous strategy, according to the Equations (9) and (10).

3.4. Node Transition Rule

As described in Section 2, the probability with which an ant k chooses to move from node i to node j is 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 (actually, any small constant positive value may be fine), and is updated at the end of each cycle according to the average quality of the solutions that involve this edge. The value of η_{ij} is determined by a greedy heuristic which guides 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 $\eta_{ij} = |\widehat{c_i c_j}|$, where $|\widehat{c_i c_j}|$ is the number of points on

$\widehat{c_i c_j}$. The value of η_{ij} is fixed during all the cycles since it considers local information only. The transition probability from node *i* to node *j* through directed edge $\overrightarrow{c_i c_j}$ is defined as

$$p_{ij} = \frac{\tau_{ij}\eta_{ij}}{\sum_{\text{for all } \overrightarrow{c_i c_h} \text{ from } c_i} \tau_{ih}\eta_{ih}}. \quad (11)$$

3.5. Pheromone Updating Rule

At the end of each cycle we update the intensity of pheromone trails of an edge by the average quality of the solutions involving this edge by simply applying Equations (2) and (3). At the end of each cycle, the pheromone intensity at directed edge $\overrightarrow{c_i c_j}$ is updated by

$$\tau_{ij} = \tau_{ij} + \max\left(\sum_{k=1}^{m} \Delta_{ij}^{k}, 0\right) \quad (12)$$

where Δ_{ij}^{k} is the quantity of new trails left by the ant *k* and it is computed by

$$\Delta_{ij}^{k} = \begin{cases} \dfrac{1}{\left|\text{tour}_k\right|} & \text{if the } \overrightarrow{c_i c_j} \in \text{tour}_k \text{ and } E_2\left(C, \text{tour}_k\right) \le \varepsilon \\ -\dfrac{E_2\left(C, \text{tour}_k\right)}{\varepsilon N} & \text{if the } \overrightarrow{c_i c_j} \in \text{tour}_k \text{ and } E_2\left(C, \text{tour}_k\right) > \varepsilon \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

According to the ACS paradigm, more quantities of pheromone trails will be laid at the edges along which most ants have constructed shorter feasible tours. As a result, the proposed rule will guide the ants to explore better tours corresponding to high quality solutions.

3.6. Refinement of Approximation Polygon

Once dominant points have been detected we apply an enhancement process to refine the point localization according to a minimal distance criterion. Let $T = \left\{c_{p_1}, c_{p_2}, \cdots, c_{p_M}\right\}$ the ordered set of detected dominant points. Considering the couples c_{p_i} and $c_{p_{i+2}}$, $i = 1, \cdots, M$, the refinement process consists in moving the intermediate point $c_{p_{i+1}}$ between c_{p_i} and $c_{p_{i+2}}$ in a new point by a local minimization distance. For all the points c_k in the arc $\widehat{c_{p_i} c_{p_{i+2}}}$ we evaluate the sum of the approximation errors, $e\left(\widehat{c_{p_i} c_k}, \overline{c_{p_i} c_k}\right) + e\left(\widehat{c_k c_{p_{i+2}}}, \overline{c_k c_{p_{i+2}}}\right)$. The new position for the point $c_{p_{i+1}}$ is chosen as follows:

$$c_{p_{i+1}} = \min_{c_k} e\left(\widehat{c_{p_i} c_k}, \overline{c_{p_i} c_k}\right) + e\left(\widehat{c_k c_{p_{i+2}}}, \overline{c_k c_{p_{i+2}}}\right) \quad (14)$$

The new dominant point $c_{p_{i+1}}$ between c_{p_i} and $c_{p_{i+2}}$ is then used in the rest of the process. The refinement step is repeated for each pair $\left(c_{p_i}, c_{p_{i+2}}\right)$, $i = 1, \cdots, M$. The refining phase terminates by deleting very close dominant points.

3.7. Summary of the Proposed Method

We summarize the proposed algorithm, denoted Poly_by_ACS, as follows:

The Poly_by_ACS Algorithm

Input:

$C = \left\{c_1, c_2, \cdots, c_N\right\}$: a set of clockwise ordered points.

N_max: the maximal number of running cycles.

1) Inizialization:

Construct the directed graph $G = \langle V, E \rangle$ as described in Subsection 3.2.

Determine the number m of ants as described in Subsection 3.3.

Set $\tau_{ij} = 1/N$ for every directed edge $\overrightarrow{c_i c_j}$.

Initialize T_i for $i = 1, \cdots, N$ as described in Subsection 3.3.

Set NC = 1, where NC is the cycle counter.

Set best_tour $= \left\{\overrightarrow{c_1 c_2}, \overrightarrow{c_2 c_3}, \cdots, \overrightarrow{c_{N-1} c_N}, \overrightarrow{c_N c_1}\right\}$.

2) For every ant do

Select a starting node as described in Subsection 3.3.

Repeat

Move to next node according to the node transition rule using Equation (11).

until a closed tour is completed.

3) Find out the shortest feasible tour, say current_tour, among the *m* tours obtained in Step 2.

4) If $\left(\left|\text{current\_tour}\right| < \left|\text{tour\_best}\right|\right)$ or $\left|\text{current\_tour}\right| = \left|\text{best\_tour}\right|$ and $E_2\left(C, \text{current\_tour}\right) < E_2\left(C, \text{best\_tour}\right)$ then best_tour = current_tour.

5) For every directed edge $\overrightarrow{c_i c_j}$ do

Update the pheromone intensity using Equations (12) and (13).

6) For every selection entry T_i do

Update the entry value using Equation (8).

7) If (NC < N_max) NC = NC + 1 and go to Step 2;

8) Refine the approximation polygon as described in Subsection 3.6.

4. Experimental Results and Comparisons

In this section we present the experimental results of the proposed ACS-based approximation algorithm. For all the processed curves we have tested both the two node selection strategies. From now on, we call *Method*_1 and *Method*_2 the approximation methods based on *Selection*_1 and *Selection*_2 algorithms, respectively. The two methods have been firstly tested on some real world images (fish, aircraft, hand, noisy key), showed in **Figure 2**. For each curve, the initial starting nodes, the polygonal approximation before and after refining the detected dominant points by applying *Method*_1 and *Method*_2 are showed in **Figures 3-6**. The performances of our algorithms have been compared to those of the method proposed by Yin [1]. The parameters used for Yin's method have been chosen according to the suggestion of the author's algorithm. In **Table 1** we present the comparative performance evaluation between our methods and Yin's one. For each image we have evaluated the initial number of points, *N*, the number of detected dominant points, *Np*, and the approximation error between the original curve and the corresponding optimal polygon, E_2. For all the images, our methods show a much better efficacy in approximating real noisy world shapes, both before and after the refining step. In all cases we obtain a lower approximation error associated to a significant reduction of *Np*. Also, our algorithms are less parameters dependent than Yin's one. Apart the ε-bound constraint and the number of the maximum cycles, Yin's ACS method uses other five parameters. The goodness of the approximating polygon is strongly dependent on the chosen parameter values. Our methods automatically choose the number of distributed ants and do not need any other parameter. We only define the maximum number of iterations and the ε-bound. Moreover, since the approach is non deterministic, different runs can lead to different solutions. We have approached this problem by choosing the initial set of starting nodes automatically and by introducing a refinement step to reduce the approximation error. Such two aspects are not present in Yin's approach. Finally, the approximation

Figure 2. Some real world curves: fish, aircraft, hand and key.

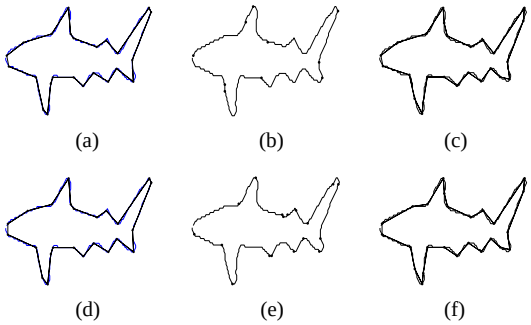

Figure 3. Fish contour: the initial starting nodes, the polygonal approximation before and after refining the detected dominant points by applying *Method*_1 in (a), (b), (c) and by applying *Method*_2 in (d), (e), (f), respectively.

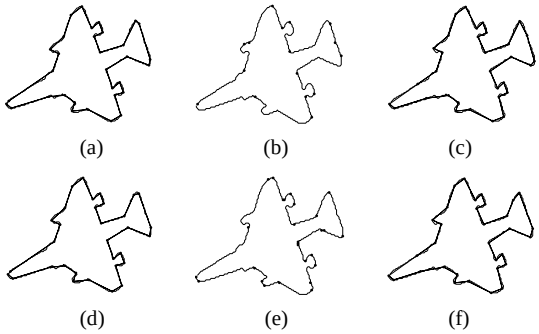

Figure 4. F10 contour: the initial starting nodes, the polygonal approximation before and after refining the detected dominant points by applying *Method*_1 in (a), (b), (c) and by applying *Method*_2 in (d), (e), (f), respectively.

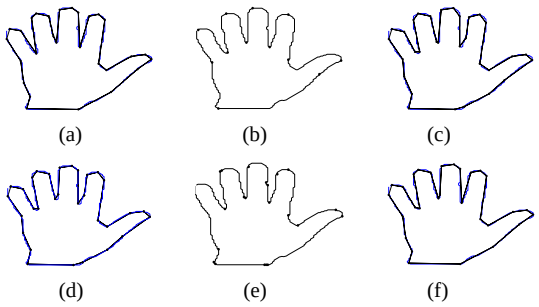

Figure 5. Hand contour: the initial starting nodes, the polygonal approximation before and after refining the detected dominant points by applying *Method*_1 in (a), (b), (c) and by applying *Method*_2 in (d), (e), (f), respectively.

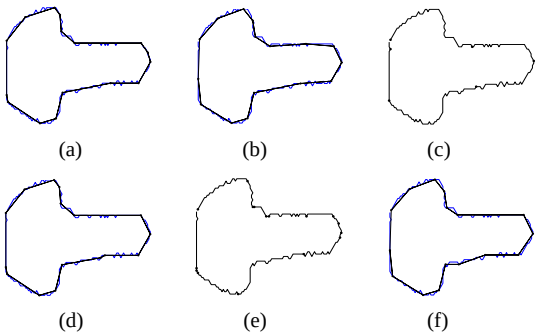

Figure 6. Key contour: the initial starting nodes, the polygonal approximation before and after refining the detected dominant points by applying *Method*_1 in (a), (b), (c) and by applying *Method*_2 in (d), (e), (f), respectively.

Table 1. The approximation error E_2 for the real curves for our methods and Yin's algorithm (in parentheses the number of dominant points).

Curve	*N*	*Method*_1 before refine	*Method*_1 after refine	*Method*_2 before refine	*Method*_2 after refine	Yin [1]
Fish	325	172.45 (25)	112.42 (24)	171.10 (25)	112.23 (24)	176.19 (25)
F10	399	198.05 (32)	135.52 (32)	192.48 (32)	134.28 (32)	208.02 (32)
Hand	513	210.63 (31)	189.99 (28)	208.63 (31)	182.10 (29)	218.28 (31)
Key	257	117.85 (16)	84.35 (16)	121.62 (16)	86.80 (16)	121.25 (16)

error levels off after only few iterations of the iterative process (in case of F10 curve after 3 iterations we achieve the best value $E_2 = 192.48$ and in case of hand curve just one iteration is needed to get $E_2 = 208.63$). This means that our methods are computationally more efficient, too. The proposed algorithms have been, also, tested on four benchmark curves (leaf, chromosome, semicircle and infinite), commonly used in many previous approaches. The experimental results have been compared to many existing methods for dominant point detection. For each proposed approximation algorithm we show the number of detected point, *Np*, the approximation error, E_2, the compression ratio, $CR = N/Np$, the combinations of the compression ratio and the approximation error, E_2/CR, E_2/CR^2 and E_2/CR^3, as suggested in [7] and [24]. The compression ratio and the approximation error are combined to measure the efficiency of the dominant point detectors and to compare them. The dominant points (circled) detected by our methods on benchmark curves (**Figure 7**) are showed in **Figure 8** and the comparative results with other methods are presented in **Table 2**. The results can be summarized as follows:

- The number of dominant points detected by our approach is an average value of the number detected by other algorithms.
- The values of E_2 and E_2/CR are smaller than all the other algorithms, except Carmona [7] in processing chromosome curve and Horng [18] in processing semicircle curve.
- The values of E_2/CR^2 and E_2/CR^3 are smaller than the values supplied by other algorithms. In some cases they are a little bit greater than Marji-Siy [27] and Carmona [7] methods.

By analyzing the comparative results, we can affirm that the proposed approach is superior to many existing algorithms based on exact search methodology. To demonstrate the feasibility of our approach we have, also, compared it to other methods based on global search heuristic, like ACS_Poly method proposed in [1], the GA-based method proposed in [16] and the TS-based method proposed in [17]. The results presented in **Table 3** confirm the superiority of our method both in effectiveness (in terms of *Np* and E_2) and in efficiency (in terms of computational cost) since only few iterations are needed to localize the vertices of the approximating polygon. Finally, we have compared our approach to other methods in order to check how the refining step is able to improve the quality of the polygonal approximation. We have analyzed the four real world curves, fish, F10, hand and key. Each of them has been first processed by our methods, *Method*_1 and *Method*_2, by Yin's one [1], by Teh-Chin's one [3] and by Wu's ones [26,28]. On each polygonal approximation we have then applied the refining procedure in order to get a better localization of dominant points and consequently a lower error, E_2. As

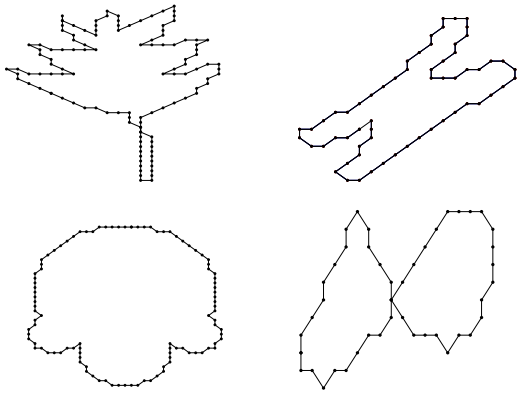

Figure 7. The benchmark curves: leaf, chromosome, semicircle, infinite.

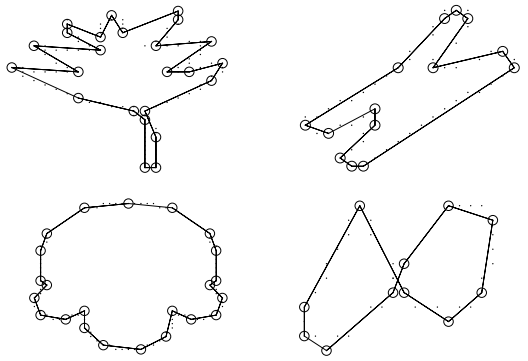

Figure 8. Dominant points (circled) for the benchmark curves.

Table 2. Comparison with other methods on the benchmark curves.

Curve	Method	Np	E_2	$CR = N/Np$	E_2/CR	E_2/CR^2	E_2/CR^3
Leaf $N = 120$	Teh & Chin [3]	28	15.43	4.286	3.600	0.840	0.196
	Cornic [25]	N/A	N/A	N/A	N/A	N/A	N/A
	Ray & Ray [12]	26	16.43	4.615	3.560	0.771	0.167
	Wu [26]	24	15.93	5.000	3.186	0.637	0.127
	Marji & Siy [27]	17	28.67	7.059	4.062	0.575	0.082
	Carmona [7]	23	15.63	5.217	2.996	0.574	0.110
	Yin [1]	24	13.73	5.000	2.746	0.549	0.110
	Horng [18]	21	14.17	5.710	2.482	0.434	0.076
	Our Methods	24	10.62	5.000	2.124	0.425	0.085
Chromosome $N = 60$	Teh & Chin [3]	16	6.40	3.750	1.707	0.41	0.121
	Cornic [25]	17	5.54	3.529	1.570	0.445	0.126
	Ray & Ray [12]	14	7.67	4.286	1.790	0.418	0.097
	Wu [26]	16	4.70	3.750	1.253	0.334	0.089
	Marji & Siy [27]	10	10.01	6.000	1.668	0.278	0.046
	Carmona [7]	14	4.93	4.286	1.150	0.268	0.063
	Yin [1]	14	6.41	4.286	1.496	0.349	0.081
	Horng [18]	10	16.99	6.000	2.832	0.472	0.079
	Our Methods	14	5.39	4.286	1.258	0.293	0.068
Semicircle $N = 102$	Teh & Chin [3]	22	20.61	4.636	4.445	0.959	0.207
	Cornic [25]	30	9.19	3.400	2.703	0.795	0.234
	Ray & Ray [12]	19	16.33	5.368	3.042	0.567	0.106
	Wu [26]	26	9.04	3.923	2.304	0.587	0.150
	Marji & Siy [27]	15	22.70	6.800	3.338	0.491	0.072
	Carmona [7]	24	9.88	4.250	2.325	0.547	0.129
	Yin [1]	23	10.50	4.435	2.368	0.534	0.120
	Horng [18]	26	4.59	3.920	1.170	0.299	0.076
	Our Methods	23	6.28	4.435	1.416	0.319	0.072
Infinite $N = 45$	Teh & Chin [3]	13	3.46	3.462	1.000	0.289	0.083
	Cornic [25]	10	4.30	4.500	0.956	0.212	0.047
	Ray & Ray [12]	12	3.75	3.750	1.000	0.267	0.071
	Wu [26]	13	5.78	3.462	1.670	0.482	0.139
	Marji & Siy [27]	N/A	N/A	N/A	N/A	N/A	N/A
	Carmona [7]	10	5.56	4.500	1.236	0.275	0.061
	Yin [1]	11	4.44	4.091	1.085	0.265	0.065
	Horng [18]	13	3.58	3.460	1.035	0.299	0.086
	Our Methods	11	3.44	4.091	0.841	0.206	0.050

Table 3. Comparison in terms of approximation error, E_2, number of dominant points, N_p, and CPU time, t in seconds, on the benchmark curves.

Curve	E_2	GA-based (t)	TS-based (t)	ACS_Poly (t)	*Method*_1 (t)	*Method*_2 (t)
Leaf N = 120	150	17 (5.7)	11 (0.9)	13 (0.7)	9 (0.8)	9 (0.8)
	100	16 (4.5)	14 (0.9)	13 (0.7)	11 (0.8)	11 (0.8)
	90	17 (5.3)	15 (0.9)	14 (0.7)	12 (0.7)	11 (0.7)
	30	21 (4.6)	20 (0.9)	19 (0.7)	17 (0.7)	16 (0.7)
	15	23 (5.4)	23 (0.9)	23 (0.7)	19 (0.7)	19 (0.7)
Chromosome N = 60	30	7 (3.0)	6 (0.5)	6 (0.4)	6 (0.4)	6 (0.4)
	20	8 (3.0)	8 (0.5)	8 (0.4)	7 (0.4)	7 (0.4)
	10	10 (3.1)	11 (0.5)	11 (0.4)	11 (0.3)	9 (0.4)
	8	12 (3.1)	12 (0.5)	11 (0.4)	11 (0.3)	11 (0.3)
	6	15 (3.3)	14 (0.5)	14 (0.4)	13 (0.3)	12 (0.3)
Semicircle N = 102	60	13 (4.6)	11 (0.9)	10 (0.6)	10 (0.6)	10 (0.6)
	30	14 (4.8)	14 (0.8)	13 (0.6)	12 (0.6)	11 (0.6)
	25	17 (4.3)	15 (0.8)	13 (0.6)	12 (0.6)	12 (0.6)
	20	19 (4.7)	16 (0.8)	16 (0.6)	15 (0.5)	15 (0.5)
	15	23 (4.4)	18 (0.8)	18 (0.6)	17 (0.5)	16 (0.5)
Infinite N = 45	30	N/A	N/A	6 (0.6)	5 (0.6)	5 (0.6)
	20	N/A	N/A	7 (0.6)	6 (0.6)	6 (0.6)
	15	N/A	N/A	7 (0.6)	6 (0.6)	6 (0.6)
	10	N/A	N/A	7 (0.6)	7 (0.6)	7 (0.6)
	6	N/A	N/A	9 (0.6)	8 (0.5)	8 (0.5)
	3	N/A	N/A	17 (0.6)	10 (0.4)	10 (0.4)

we can see from the numerical results showed in **Table 4**, the refining procedure is able to improve all the polygonal approximations. In all the cases we get a lower error, E_2. However, the best approximations are still achieved by applying our method, confirming its superiority as a general approach for dominant point detection and polygonal approximation. Some visual comparisons are showed in **Figures 9** and **10**.

5. Conclusion

In this work we have presented a novel method for approximating a digital curve. The algorithm is inspired by the ant colony search suggested by Yin in [1] but it has resulted in a much more efficient and effective approximation algorithm. The performance of our approach has been first compared to that of the method proposed by Yin on some real world curves. The experimental results have showed a much better efficacy in approximating real noisy world shapes, both before and after the refining step. Also, our algorithms are less parameters dependent than Yin's one. Apart the ε-bound constraint and the number of the maximum cycles, Yin's ACS method uses other five parameters and the goodness of the approximating polygon is strongly dependent on the chosen parameter values. On the contrary, our approach automatically choose the number of distributed ants. No other parameters are used in the whole process. Also, the localization of the best dominant points can be obtained by the refinement step in a very fast way. Finally, the performances of our methods level off after only few iterations of the approximating process. This means that our methods are computationally more efficient, too. We can summarize the differences and the contribution of our approach against the one in [1] in four main aspects: less parameter dependance, automatic choosing of starting points and number of distributed ants, updating of the best tour during search (Step 4 of the Poly_by_ACS Algorithm) and refining. First preliminary results of our

Table 4. Comparison with other methods, without or with the refinement step, on real world curves.

	Fish N_p	E_2	F10 N_p	E_2	Hand N_p	E_2	Key N_p	E_2
*Method*_1 before refine	25	172.45	32	198.05	31	210.63	16	117.85
*Method*_1 after refine	24	112.42	32	135.52	28	189.99	16	84.35
*Method*_2 before refine	25	171.10	32	192.48	31	208.63	16	121.62
*Method*_2 after refine	24	112.23	32	134.28	31	149.56	16	86.80
Yin [1]	25	176.19	32	208.02	31	218.28	16	121.25
Yin [1] with refine	25	107.95	32	146.23	31	159.56	16	89.39
Teh & Chin [3]	41	971.16	64	1488.60	48	10080.20	35	435.12
Teh & Chin [3] with refine	41	147.75	64	183.61	48	519.61	35	145.79
Wu [26]	33	834.18	35	1977.36	36	8154.60	21	6160.35
	24	1536.60	29	5899.61	30	8165.34	13	16856.29
Wu [26] with refine	33	124.88	35	750.03	36	413.70	21	190.49
	24	326.90	29	1064.23	30	482.61	13	2021.33
Wu [28]	18	20160.08	32	19629.77	31	25912.99	14	6141.48
Wu [28] with refine	18	2113.43	32	1687.33	31	537.19	14	1189.87

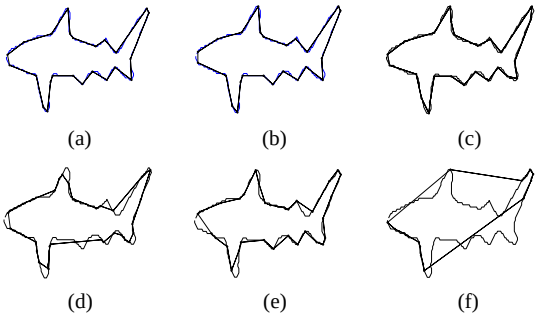

Figure 9. Visual comparisons with other methods on fish contour: (a) *Method*_1; (b) *Method*_2; (c) Yin's [1], (d) Teh-Chin's [3], (e) Wu's [28] and (f) Wu's [26].

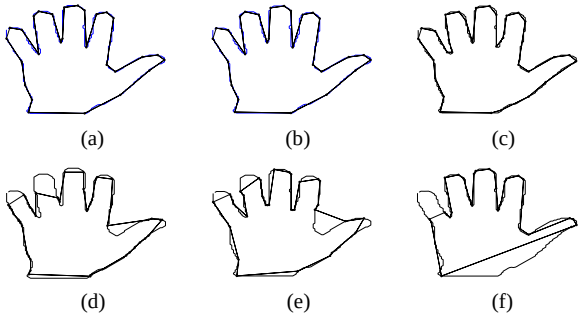

Figure 11. Visual comparisons with other methods on hand contour: (a) *Method*_1; (b) *Method*_2; (c) Yin's [1]; (d) Teh-Chin's [3]; (e) Wu's [28]; and (f) Wu's [26].

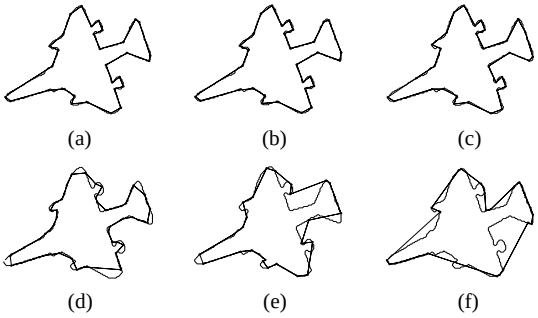

Figure 10. Visual comparisons with other methods on F10 contour: (a) *Method*_1; (b) *Method*_2; (c) Yin's [1]; (d) Teh-Chin's [3]; (e) Wu's [28]; and (f) Wu's [26].

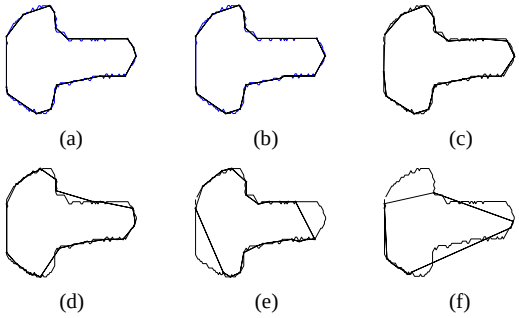

Figure 12. Visual comparisons with other methods on key contour: (a) *Method*_1; (b) *Method*_2; (c) Yin's [1]; (d) Teh-Chin's [3]; (e) Wu's [28]; and (f) Wu's [26].

approach have been presented in [29]. In the new version we have proposed in this paper we have introduced the possibility to delete very close dominant points, by obtaining a significant reducing of N_P with a negligible increasing of approximation error. We have expanded the experimental results by visual comparisons with other methods (**Figures 9-12**). Also, we have tested the importance and the effectiveness of refinement step, as showed

in **Table 4**. The performance of our algorithms have been tested on four contour commonly used curves (leaf, chromosome, semicircle and infinite) in many previous approaches. This testing has confirmed that the proposed approach is superior to many existing algorithms based on exact search methodology. Finally, our approach has been compared to other methods based on global search heuristic, like ACS_Poly method proposed in [1], the GA-based method proposed in [16] and the TS-based method proposed in [17]. Again, the excellent results have confirmed the superiority of our methods both in effectiveness (in terms of *Np* and E_2) and in efficiency (in terms of computational cost). The two proposed algorithms, *Method*_1 and *Method*_2, have comparable performances, if applied on very simple curves, as we can see from the results shown in **Table 2**. However, if applied on real world curves, *Method*_2 presents a lower approximation error, both before and after the refining step, as we can see in **Table 1**. Finally, we can notice another little difference on behalf of *Method*_2 in **Table 3**, where we present the number of detected dominant points with a same approximation error.

REFERENCES

[1] P. Y. Yin, "Ant Colony Search Algorithms for Optimal Polygonal Approximation of Plane Curves," *Pattern Recognition*, Vol. 36, No. 8, 2003, pp. 1783-1797. doi:10.1016/S0031-3203(02)00321-7

[2] E. Attneave, "Some Informational Aspects of Visual Perception," *Psychological Review*, Vol. 61, No. 3, 1954, pp. 183-193. doi:10.1037/h0054663

[3] C.-H. Teh and R. T. Chin, "On the Detection of Dominant Points on Digital Curves," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 8, 1989, pp. 859-871. doi:10.1109/34.31447

[4] W. Y. Wu and M. J. Wang, "Detecting the Dominant Points by the Curvature-Based Polygonal Approximation," *CVGIP*: *Graphical Model Image Processing*, Vol. 55, No. 2, 1993, pp. 79-88. doi:10.1006/cgip.1993.1006

[5] M. J. Wang, W. Y. Wu, L. K. Huang and D. M. Wang, "Corner Detection Using Bending Value," *Pattern Recognition Letters*, Vol. 16, No. 8, 1995, pp. 575-583. doi:10.1016/0167-8655(95)80003-C

[6] A. Held, K. Abe and C. Arcelli, "Towards a Hierarchical Contour Description via Dominant Point Detection," *IEEE Transactions on System Man Cybernetics*, Vol. 24, No. 6, 1994, pp. 942-949. doi:10.1109/21.293514

[7] A. Carmona-Poyato, N. L. Fernández-Garca, R. Medina-Carnicer and F. J. Madrid-Cuevas, "Dominant Point Detection: A New Proposal," *Image and Vision Computing*, Vol. 23, No. 13, 2005, pp. 1226-1236. doi:10.1016/j.imavis.2005.07.025

[8] J. Sklansky and V. Gonzalez, "Fast Polygonal Approximation of Digitized Curves," *Pattern Recognition*, Vol. 12, No. 5, 1980, pp. 327-331. doi:10.1016/0031-3203(80)90031-X

[9] B. K. Ray and K. S. Ray, "Determination of Optimal Polygon from Digital Curve Using L1 Norm," *Pattern Recognition*, Vol. 26, No. 4, 1993, pp. 505-509. doi:10.1016/0031-3203(93)90106-7

[10] U. Ramer, "An Iterative Procedure for the Polygonal Approximation of Plane Curves," *Computer Graphics Image Processing*, Vol. 1, No. 3, 1972, pp. 244-256. doi:10.1016/S0146-664X(72)80017-0

[11] N. Ansari and E. J. Delp, "On Detecting Dominant Points," *Pattern Recognition*, Vol. 24, No. 5, 1991, pp. 441-450. doi:10.1016/0031-3203(91)90057-C

[12] B. K. Ray and K. S. Ray, "A New Split-and-Merge Technique for Polygonal Approximation of Chain Coded Curves," *Pattern Recognition Letters*, Vol. 16, No. 2, 1995, pp. 161-169. doi:10.1016/0167-8655(94)00081-D

[13] J. G. Dunham, "Optimum Uniform Piecewise Linear Approximation of Planar Curves," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 1, 1986, pp. 67-75. doi:10.1109/TPAMI.1986.4767753

[14] Y. Sato, "Piecewise Linear Approximation of Plane Curves by Perimeter Optimization," *Pattern Recognition*, Vol. 25, No. 12, 1992, pp. 1535-1543. doi:10.1016/0031-3203(92)90126-4

[15] S. C. Huang and Y. N. Sun, "Polygonal Approximation Using Genetic Algorithms," *Pattern Recognition*, Vol. 32, No. 8, 1999, pp. 1409-1420. doi:10.1016/S0031-3203(98)00173-3

[16] P. Y. Yin, "Genetic Algorithms for Polygonal Approximation of Digital Curves," *International Journal of Pattern Recognition and Artificial Intelligence*, Vol. 13, No. 7, 1999, pp. 1-22. doi:10.1142/S0218001499000598

[17] P. Y. Yin, "A Tabu Search Approach to the Polygonal Approximation of Digital Curves," *International Journal of Pattern Recognition and Artificial Intelligence*, Vol. 14, No. 2, 2000, pp. 243-255. doi:10.1142/S0218001400000167

[18] J. H. Horng, "Improving Fitting Quality of Polygonal Approximation by Using the Dynamic Programming Technique," *Pattern Recognition Letters*, Vol. 23, No. 14, 2002, pp. 1657-1673. doi:10.1016/S0167-8655(02)00129-0

[19] F. Glover, "Future Paths for Integer Programming and Links to Artificial Intelligence," *Computers & Operations Research*, Vol. 13, No. 5, 1986, pp. 533-549. doi:10.1016/0305-0548(86)90048-1

[20] M. Dorigo, "Optimization, Learning, and Natural Algorithms," Ph.D. Thesis, Politecnico di Milano, Milano, 1992.

[21] Wikipedia, 2012. http://en.wikipedia.org/wiki/Ant_colony_optimization

[22] M. Dorigo, G. Di Caro and L. M. Gambardella, "Ant Algorithms for Discrete Optimization," *Artificial Life*, Vol. 5, No. 2, 1999, pp.137-172. doi:10.1162/106454699568728

[23] L. Costa and R. Cesar, "Shape Analysis and Classification Theory and Practice," CRC Press, Boca Raton, 2001.

[24] P. Y. Yin, "Polygonal Approximation of Digital Curves

Using the State-of-the-Art Metaheuristics," In: G. Obinata and A. Dutta, Eds., *Vision Systems*: *Segmentation and Pattern Recognition*, I-Tech, Vienna, 2007, pp. 451-464. doi:10.5772/4974

[25] P. Cornic, "Another Look at Dominant Point Detection of Digital Curves," *Pattern Recognition Letters*, Vol. 18, No. 1, 1997, pp. 13-25. doi:10.1016/S0167-8655(96)00116-X

[26] W. Y. Wu, "Dominant Point Detection Using Adaptive Bending Value," *Image and Vision Computing*, Vol. 21, No. 6, 2003, pp. 517-525. doi:10.1016/S0262-8856(03)00031-3

[27] M. Marji and P. Siy, "A New Algorithm for Dominant Points Detection and Polygonization of Digital Curves," *Pattern Recognition*, Vol. 36, No. 10, 2003, pp. 2239-2251. doi:10.1016/S0031-3203(03)00119-5

[28] W. Y. Wu, "A Dynamic Method for Dominant Point Detection," *Graphical Models*, Vol. 64, No. 5, 2003, pp. 304-315. doi:10.1016/S1077-3169(02)00008-4

[29] C. Di Ruberto and A. Morgera, "A New Algorithm for Polygonal Approximation Based on Ant Colony System," *Lecture Notes in Computer Science*, Vol. 5716, 2009, pp. 633-641. doi:10.1007/978-3-642-04146-4_68

