A New Algorithm for Polygonal Approximation Based on Ant Colony Optimization

Cecilia Di Ruberto and Andrea Morgera

Dipartimento di Matematica e Informatica, Università di Cagliari, Cagliari, Italy

Abstract. In 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. In this paper we present a novel method that combines the dominant point detection and the ant colony optimization search. The excellent results have been compared both to works using an optimal search approach and to works based on exact approximation strategy.

1 Introduction

Computer imaging has developed as an interdisciplinary research field whose focus is on the acquisition and processing of visual information by computer. Among all aspects underlying visual information, the shape of the objects plays a special role. In shape analysis an important step consists in extracting meaningful features from digital curves. 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 [1],[2]. There are many approaches developed for detecting dominant points. They can be classified into two main categories: corner detection approaches and polygonal approximation approaches. Corner detection approaches aim to detect potential significant points, but they cannot represent smooth curve appropriately. The performance of these detectors depends on the accuracy of the curvature evaluation [9],[10]. For polygonal approximation approaches, sequential, iterative and optimal algorithms are commonly used. Sequential approaches are simple and fast, but the results depend on the location of the point where they start the scan-along process and they can miss important features. The iterative approaches split and merge curves iteratively until they meet the preset allowances [7],[8]. Their results may be far from the optimal one if a poor initial segmentation is used. The optimal approaches tend to find the optimal polygonal approximation based on specified criteria and error bound constraints [4]-[6]. The idea 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. 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 results in an exponential complexity. However, there exist some heuristic approaches (genetic algorithms and tabu search) which can find solutions very close to the global optimal one in a relative short time. 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 in [6] but it results in a much more efficient and effective approximation algorithm. The excellent results have been compared to works using an optimal search approach and to works based on exact approximation strategy.

2 Ant Colony Optimization Algorithms

Ant Colony Optimization (ACO) firstly proposed by Dorigo in [3] is based on a computational paradigm inspired by the way real ant colonies function. The ants make use of a substance, called pheromone, to communicate information regarding shortest paths to food. A moving ant lays some pheromone on the ground, thus making a path by a trail of this substance. While an ant moves practically at random, it encounters a previously laid trail and can decide, with high probability, to follow it, thus reinforcing the trail with its own pheromone. The process is characterized by a positive feedback loop, where the probability with which an ant chooses a path increases with the number of ants that previously chose the same path. The ACO paradigm is inspired by this process. Artificial ants build solutions (tours) of a problem by moving on the problem graph from one node to another. The algorithm executes N_{max} iterations. During each iteration m ants build a tour in which a probabilistic decision rule is applied. If an ant in node i chooses to move to node j, the edge (i, j) is added to the tour under construction. This step is repeated until the ant has completed its tour. After ants have built their tours, each ant deposits pheromone on the visited edges to make them more desirable for future ants. The amount of pheromone trail τ_{ij} associated to edge (i, j) is intended to represent the learned desirability of choosing node j when in node i. The pheromone trail information is changed during problem solution to reflect the experience acquired by ants during problem solving. 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 it deposits on the edges which it used to generate the tour. This choice helps to direct search towards good solutions. The ACO algorithms have been applied to many different discrete optimization problems, like the travelling salesman problem and the quadratic assignment problem. Recent applications cover problems like vehicle routing, sequential ordering, graph coloring, routing in communications networks, and so on. In this work we use the ACO paradigm to solve the polygonal approximation problem.

3 The ACO-Based Proposed Method

We first define the problem of polygonal approximation in terms of optimization problem and then we describe how we use the ACO algorithm to solve it.

3.1 Problem Formulation and Graph Representation

Given a digital curve represented by a set of N clockwise-ordederd points, $C = \{c_1, c_2, ..., c_N\}$ where $c_{(i+1)modN}$ is the succeeding point of c_i, we define arc $\widehat{c_ic_j}$ as the set of 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 $e(\widehat{c_ic_j}, \overline{c_ic_j})$ that can be measured by a distance norm. We use the sum of squared perpendicular distance from every data point on $\widehat{c_ic_j}$ to $\overline{c_ic_j}$:

$$e(\widehat{c_ic_j}, \overline{c_ic_j}) = \sum_{c_k \in \widehat{c_ic_j}} d^2(c_k, \overline{c_ic_j}) \tag{1}$$

where $d(c_k, \overline{c_ic_j})$ is the perpendicular distance from point c_k to the corresponding chord $\overline{c_ic_j}$. 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 ordered set $T = \{c_{p_1}, c_{p_2}, ..., c_{p_M}\}$ where $T \subset C$ and $M \leq N$ such that M is minimal and the set of M line segments, $P = \{\overline{c_{p_1}c_{p_2}}, \overline{c_{p_2}c_{p_3}}, ..., \overline{c_{p_{M-1}}c_{p_M}}, \overline{c_{p_M}c_{p_1}}\}$, composes an approximating polygon to the point set C with an error norm between C and P no more than a pre-specified tolerance, ϵ. The error between C and P, denoted by $E_2(C, P)$, is defined as the sum of the approximation errors between the M arcs $\{\widehat{c_{p_1}c_{p_2}}, \widehat{c_{p_2}c_{p_3}}, ..., \widehat{c_{p_{M-1}}c_{p_M}}, \widehat{c_{p_M}c_{p_1}}\}$ and the corresponding M line segments $\{\overline{c_{p_1}c_{p_2}}, \overline{c_{p_2}c_{p_3}}, ..., \overline{c_{p_{M-1}}c_{p_M}}, \overline{c_{p_M}c_{p_1}}\}$, with $c_{p_{M+1}} \equiv c_{p_1}$:

$$E_2(C, P) = \sum_{i=1}^{M} e(\widehat{c_{p_i}c_{p_{i+1}}}, \overline{c_{p_i}c_{p_{i+1}}}). \tag{2}$$

To apply the ACO paradigm, 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. $V = C$. The edge set E is generated as follows. Initially, the set E is created as an empty set. Then, 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 $\overrightarrow{c_ic_j}$ is added to the edge set E. Thus, we have:

$$E = \{\overrightarrow{c_ic_j} | e(\widehat{c_ic_j}, \overline{c_ic_j}) \leq \epsilon\}. \tag{3}$$

This means that E contains edges the error of which is smaller or equal to ϵ. The edge is directed to avoid the ants walking backward. Then, the problem of polygonal approximation is equivalent to finding the shortest closed circuit on the directed graph $G =< V, E >$ such that $E_2 \leq \epsilon$. In the following, the closed circuit completed by the ant k will be denoted by $tour_k$, its length will be $|tour_k|$ and the approximation error between the original curve C and the approximating polygon corresponding to $tour_k$ will be $E_2(C, tour_k)$.

3.2 Starting Node Initialization and Selection

During each cycle an ant chooses a starting node and sequentially constructs a closed path to finish its tour. To find the shortest closed tour, it is convenient to place the ants at the nodes with a high 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*. They differ with respect to the way the initial set of starting nodes is defined. Let's describe the two algorithms in detail.

Selection_1. Shape signature is one of the most common shape descriptors and it 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 relative boundary points guide us 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 distributed ants is equal to the number of the signature extrema and the m relative boundary points represent the m starting nodes at the beginning of 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 $T_i, i = 1, 2, ..., N$. Initially,

$$T_i = \begin{cases} 1 & \text{if } i \text{ is a starting node} \\ \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

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,

$$Choose_i = \frac{T_i}{\sum_{j=1}^{N} T_j} \tag{5}$$

At the end of each cycle, the value of $T_i, i = 1, 2, ..., N$ is updated. Let's denote by $Start\_Node_i$ the set of ants which start with the node i at the current cycle and by $|Start\_Node_i|$ 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

$$T_i = \begin{cases} \frac{1}{|Start\_Node_i|} \sum_{j \in Start\_Node_i} \frac{1}{|tour_j|} + Choose_i & \text{if } i \text{ is a start node} \\ & \text{at current cycle} \\ \\ T_i & \text{otherwise} \end{cases} \tag{6}$$

During the process, in the early iterations the ants will prefer the nodes which have not been chosen yet as starting nodes, by exploring new solutions. After all the nodes have been tested, they will be 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. For each node $i, i = 1, ..., N$, of the graph we evaluate the greatest approximation error among all the directed edges departing from i. We generate a list of the N greatest 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 detect the nodes where these edges depart from. The list of such nodes, after eliminating duplications, represents the set of starting nodes at the beginning of the first cycle and its size is the number of the distributed ants. The selection phase continues as described previously.

3.3 Node Transition and Pheromone Updating Rules

According to ACO paradigm, 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 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 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 $\eta_{ij} = |\widehat{c_i c_j}|$, 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

$$p_{ij} = \frac{\tau_{ij}\eta_{ij}}{\sum_{\text{for all } \overrightarrow{c_i c_h} \text{ from} c_i} \tau_{ih}\eta_{ih}}. \tag{7}$$

At the end of each cycle we update the intensity of pheromone trails of an edge by the average quality of the solutions that use this edge. At the end of each cycle, the pheromone intensity at directed edge $\overrightarrow{c_i c_j}$ is updated by combining the pheromone evaporation and deposition as follows:

$$\tau_{ij} = \rho\tau_{ij} + \max(\sum_{k=1}^{m} \Delta_{ij}^k, 0) \tag{8}$$

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

$$\Delta_{ij}^k = \begin{cases} \frac{1}{|tourk|} & \text{if the } \overrightarrow{c_i c_j} \in tour_k \text{ and } E_2(C, tour_k) \leq \epsilon \\ -\frac{E_2(C,tour_k)}{\epsilon N} & \text{if the } \overrightarrow{c_i c_j} \in tour_k \text{ and } E_2(C, tour_k) > \epsilon \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

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

3.4 Refinement of Approximation Polygon

On the detected dominant points we apply an enhancement process to refine their localization according to a minimal distance criterion. Let $T = \{c_{p_1}, c_{p_2}, ..., c_{p_M}\}$ the ordered set of detected dominant points. Considering the couple c_{p_i} and $c_{p_{i+2}}$, $i = 1, ..., M$, 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 distance minimization. For all the points c_k on the arc $\widehat{c_{p_i}, c_{p_{i+2}}}$ we evaluate the sum of the approximation errors, $e(\widehat{c_{p_i}k}, \overline{c_{p_i}k}) + e(\widehat{kc_{p_{i+2}}}, \overline{kc_{p_{i+2}}})$. The new position for the point $c_{p_{i+1}}$ is chosen as follows:

$$c_{p_{i+1}} = \min_{c_k} e(\widehat{c_{p_i}c_k}, \overline{c_{p_i}c_k}) + e(\widehat{c_k c_{p_{i+2}}}, \overline{c_k c_{p_{i+2}}}) \tag{10}$$

The new 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 $(c_{p_i}, c_{p_{i+2}})$, $i = 1, ..., M$.

4 Experimental Results and Conclusions

We present now the experimental results of the proposed approximation algorithm. In the following, we call by *Method_1* and by *Method_2* the methods based on *Selection_1* and *Selection_2* algorithms, respectively. The methods have been first tested on some real contours, showed in Fig.1. In Fig.2 we show the initial starting nodes, the approximating polygon before and after the refinement step by applying *Method_1* and *Method_2* for the aircraft curve. The performances have been compared to those of the method proposed by Yin [6] and presented in table 1. We have evaluated the initial number of points, N, the number of dominant points, Np, and the approximation error, E_2. For all the curves, our

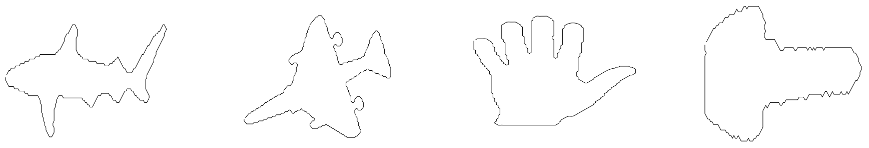

Fig. 1. Some real world curves: fish, aircraft, hand, key

Table 1. The initial number of points, N, the number of dominant points, Np, and the approximation error E_2 for the real curves for our methods and Yin's algorithm. In bold the best approximation errors.

	N	Np	*Method_1* before refine	*Method_1* after refine	*Method_2* before refine	*Method_2* after refine	Yin [6]
Fish	325	25	172.45	102.03	171.10	**101.83**	176.19
F10	399	32	193.03	135.52	192.48	**134.28**	208.02
Hand	513	31	210.63	**149.56**	210.63	**149.56**	218.28
Key	257	16	116.14	**80.96**	117.85	82.79	121.25

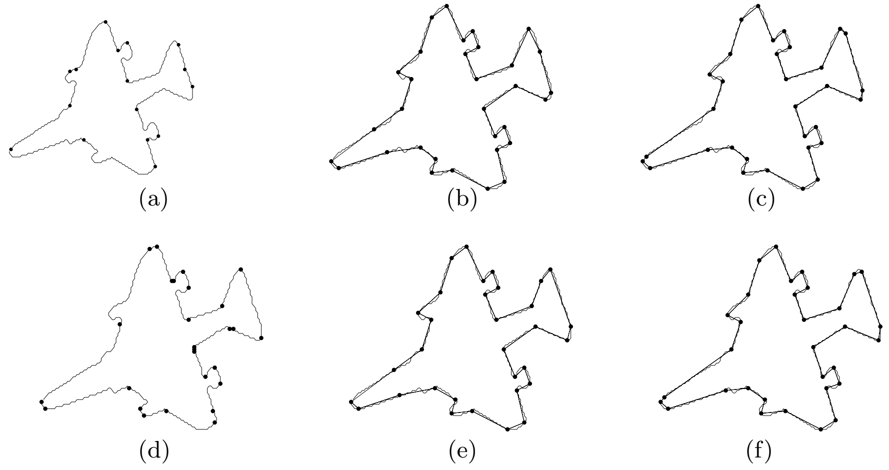

Fig. 2. F10 contour: the initial starting nodes, the polygonal approximation before and after refining the detected dominant points by applying *Method_1* in (a), (b), (c) and by applying *Method_2* in (d), (e), (f), respectively

Table 2. Comparison in terms of number of dominant points, Np, and E_2 on the benchmark curves

Curves	E_2	GA-based	TS-based	ACO_Poly	Method_1	Method_2
Leaf	150	17	11	13	9	9
$N = 120$	100	16	14	13	11	11
	90	17	15	14	12	11
	30	21	20	19	17	16
	15	23	23	23	19	19
Choromosome	30	7	6	6	6	6
$N = 60$	20	8	8	8	7	7
	10	10	11	11	11	9
	8	12	12	11	11	11
	6	15	14	14	13	12
Semicircle	60	13	11	10	10	10
$N = 102$	30	14	14	13	12	11
	25	17	15	13	12	12
	20	19	16	16	15	15
	15	23	18	18	17	16
Infinite	30	N/A	N/A	6	5	5
$N = 45$	20	N/A	N/A	7	6	6
	15	N/A	N/A	7	6	6
	10	N/A	N/A	7	7	7
	6	N/A	N/A	9	8	8
	3	N/A	N/A	17	10	10

Table 3. Comparison with other methods on the benchmark curves

Curves	Method	Np	E_2	$CR = N/Np$	E_2/CR	E_2/CR^2	E_2/CR^3
Leaf	Teh_Chin	28	15.43	4.286	3.600	0.840	0.196
$N = 120$	Cornic	N/A	N/A	N/A	N/A	N/A	N/A
	Ray-Ray	26	16.43	4.615	3.560	0.771	0.167
	Wu	24	15.93	5.000	3.186	0.637	0.127
	Marji-Siy	17	28.67	7.059	4.062	0.575	0.082
	Carmona	23	15.63	5.217	2.996	0.574	0.110
	Yin	24	13.73	5.000	2.746	0.549	0.110
	Our Method_1	24	10.62	5.000	2.124	0.425	0.085
	Our Method_2	24	10.62	5.000	2.124	0.425	0.085
Chromosome	Teh_Chin	16	6.40	3.750	1.707	0.455	0.121
$N = 60$	Cornic	17	5.54	3.529	1.570	0.445	0.126
	Ray-Ray	14	7.67	4.286	1.790	0.418	0.097
	Wu	16	4.70	3.750	1.253	0.334	0.089
	Marji-Siy	10	10.01	6.000	1.668	0.278	0.046
	Carmona	14	4.93	4.286	1.150	0.268	0.063
	Yin	14	6.41	4.286	1.496	0.349	0.081
	Our Method_1	14	5.39	4.286	1.258	0.293	0.068
	Our Method_2	14	5.39	4.286	1.258	0.293	0.068
Semicircle	Teh_Chin	22	20.61	4.636	4.445	0.959	0.207
$N = 102$	Cornic	30	9.19	3.400	2.703	0.795	0.234
	Ray-Ray	19	16.33	5.368	3.042	0.567	0.106
	Wu	26	9.04	3.923	2.304	0.587	0.150
	Marji-Siy	15	22.70	6.800	3.338	0.491	0.072
	Carmona	24	9.88	4.250	2.325	0.547	0.129
	Yin	23	10.50	4.435	2.368	0.534	0.120
	Our Method_1	23	6.28	4.435	1.416	0.319	0.072
	Our Method_2	23	6.28	4.435	1.416	0.319	0.072
Infinite	Teh_Chin	13	3.46	3.462	1.000	0.289	0.083
$N = 45$	Cornic	10	4.30	4.500	0.956	0.212	0.047
	Ray-Ray	12	3.75	3.750	1.000	0.267	0.071
	Wu	13	5.78	3.462	1.670	0.482	0.139
	Marji-Siy	N/A	N/A	N/A	N/A	N/A	N/A
	Carmona	10	5.56	4.500	1.236	0.275	0.061
	Yin	11	4.44	4.091	1.085	0.265	0.065
	Our Method_1	11	3.44	4.091	0.841	0.206	0.050
	Our Method_2	11	3.44	4.091	0.841	0.206	0.050

methods show a much better efficacy, 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 method uses other five parameters. The detected 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 error levels off after only few iterations of the iterative process (in case of F10 curve after 3 iterations we achieve the best value $E_2 = 193.03$ and in case of hand curve just one iteration is needed to get $E_2 = 210.63$). This means that our methods are computationally more efficient, too. Our algorithms have been, also, tested on four commonly used curves (leaf, chromosome, semicircle and infinite). We have, first, compared our approach to other methods based on global search heuristic, like ACS_Poly method [6], the GA-based method [4] and the TS-based method [5]. The results confirm the superiority of our method both in effectiveness and in efficiency. The experimental results have been, finally, compared to many existing methods for dominant point detection. In table 3 for each algorithm we show the number of detected point, Np, the approximation error, E_2, the compression ratio, $CR = N/Np$, as suggested in [1] and [7] to measure the efficiency of the dominant point detectors and to compare them. Analyzing the table, we can affirm that the proposed approach is superior to many existing algorithms based on exact search methodology, too. Our future research will include the task of analyzing the stability of the proposed method with respect to affine transformations (rotation, scale, translation).

References

1. Carmona-Poyato, A., Fernández-García, N.L., Medina-Carnicer, R., Madrid-Cuevas, F.J.: Dominant point detection: A new proposal. Image and Vision Computing 23, 1226–1236 (2005)
2. Cornic, P.: Another look at dominant point detection of digital curves. Pattern Recognition Letters 18, 13–25 (1997)
3. Dorigo, M.: Optimization, learning, and natural algorithms. Ph.D. Thesis, Dip. Elettronica e Informazione, Politecnico di Milano, Italy (1992)
4. Yin, P.Y.: Genetic algorithms for polygonal approximation of digital curves. Int. J. Pattern Recognition Artif. Intell. 13, 1–22 (1999)
5. Yin, P.Y.: A tabu search approach to the polygonal approximation of digital curves. Int. J. Pattern Recognition Artif. Intell. 14, 243–255 (2000)
6. Yin, P.Y.: Ant colony search algorithms for optimal polygonal approximation of plane curves. Pattern Recognition 36(8), 1783–1797 (2003)
7. Marji, M., Siy, P.: A new algorithm for dominant points detection and polygonization of digital curves. Pattern Recognition 36, 2239–2251 (2003)
8. Ray, B.K., Ray, K.S.: Detection of significant points and polygonal approximation of digitized curves. Pattern Recognition Letters 22, 443–452 (1992)
9. Teh, C.H., Chin, R.T.: On the detection of dominant points on digital curves. IEEE Trans. on Pattern Analysis and Machine Intelligence 11(8), 859–871 (1989)
10. Wu, W.Y.: Dominant point detection using adaptive bending value. Image and Vision Computing 21, 517–525 (2003)