Mesh Optimization for Surface Approximation Using an Efficient Genetic Algorithm with a novel 2-D Orthogonal Crossover

Hui-Ling Huang*, Shinn-Ying Ho, Tzu-Chien Wu, Fu-Sin Yau, and Yan-Fan Chen
（黃慧玲） （何信瑩） （吳子謙） （尤富新） （陳彥帆）
Department of Information Engineering, Feng Chia University,
100 Wen-hwa Road, Taichung, Taiwan 407, ROC
E-mail: hlhuang@plum.iecs.fcu.fcu.tw. FAX:886-4-4516101

Abstract

In this paper, the surface approximation using a mesh optimization approach is investigated. The mesh optimization problem is how to locate a limited number n of grid points such that the established mesh of n grid points approximates the digital surface of N points as closely as possible. The resultant combinatorial problem has an NP-hard search space of C(N, n) instances, i.e., the number of ways of choosing n grid points out of N points. A genetic-algorithm-based method has been proposed for establishing optimal mesh surfaces. It was shown that the GA-based method is effective in searching the combinatorial space which is intractable when n and N are in order of thousands. This paper proposes an efficient genetic algorithm with a novel 2-D orthogonal crossover for obtaining the optimal solution to the surface approximation problem using a triangular mesh. It is shown empirically that the proposed efficient genetic algorithm outperforms the existing GA-based method in solving the mesh optimization problem in terms of the approximation quality and the convergence speed, especially in solving large mesh optimization problems.

Keywords: Genetic algorithm; Evolutionary algorithm; Optimization; Surface approximation; 2-D Orthogonal array crossover; Mesh Optimization

1. INTRODUCTION

Surface representation plays an important role in a variety of disciplines, including computer graphics, computer-aided design, computer vision, and geographic data processing. Surface approximation is one of the popular approaches, which can provide good representations of 3-D shapes at different resolutions. Many surface approximation methods can be found in literature such as bicubic [1], subdivision surface [2], a 3-D finite element mesh generator [3], multilevel finite element method [4], wavelets and quadtree data structures [5], and spherical wavelets [6].

There are relatively few studies on mesh optimization for surface approximation using the evolutionary computation approach. A new optimization method for surface approximation using an evolutionary computation approach was proposed by Fujiwara and Sawai [7]. The optimization problem of reconstructing the approximating surface is to select n grid points out of N sample points such that the resultant 2-D Delaunay triangulation can determine a 3-D facial triangular mesh which approximates the 3-D surface as closely as possible. This problem may be viewed as a combinatorial problem having an NP-hard search space of C(N, n) instances. In the context of their problem, the original surface $z = f(x, y)$ is regarded as a manifold of data in R^2 and a volume difference in Euclidean R^3 space of the volumn sandwiched by the facial surface and the approximating 3-D triangular mesh surface is regarded as the approximation error. In the GA-based approach by Fujiwara and Sawai [7], phenotype value is the same as genotype simply representing the spatial position of each point, a set of triangulations is a population and each triangulation is subject to genetic operations. To minimize the approximation error, the used GA has to avoid unnecessarily locating the points on smooth portions and to concentrate the distribution optimally on curved parts. It was shown that the GA-based method is effective in searching the combinatorial space, which is intractable when n and N are in order of thousands.

The main power of GAs arises from crossover [8]. The crossover operator used in [7] *randomly* chooses a horizontal or vertical grid line to separate the mesh of each parent into two parts and exchanges information of one part between parents. After performing this operator on the parents of n grid points, the children may yield a mesh with a number of grid points karger or smaller than n. To cope with the combinatorial problem efficiently using GAs, especially while n and N are large, a powerful crossover operator is desirable.

In this paper, we solve the mesh optimization problem using an efficient genetic algorithm with a novel 2-D orthogonal crossover (OAX). OAX is based on the systematic reasoning ability of orthogonal arrays (OAs). Theoretical analysis and experimental studies for the superiority of the crossover using OAs, called 1-D

orthogonal crossover, can be found in our recent work [9]. The random repair operator may disrupt the good genes collected by the crossover operator. The proposed 2-D OAX can always produce feasible solutions such that no repair operator is needed. Furthermore, the 2-D OAX separates the mesh of each parent into many parts and the meshes of the children are formed from the best combinations of the better parts from the parents rather than the conventional random combinations. The choice of the better parts is derived by way of a systematic reasoning approach for evaluating the contribution of the individual parts based on OA. It will be shown that the proposed efficient genetic algorithm outperforms the existing GA-based method
[7] in solving the mesh optimization problem in terms of the approximation quality and the convergence speed, especially in solving large mesh optimization problem.

2. Problem formulation

2.1 Surface representation

Let us view the 3-D surface image as a function f:(x, y) Rn→z R. Thus, we can yield a rectangular grid on the (x, y) plane, where each grid point's height z represents the depth of the surface. Let Nx and Ny be the numbers of grid points in the x and y axes, respectively. The total number N of grid points is equal to Nx• Ny

This generic surface representation is also available for representing the surface of the cylindrical coordinate system. For example, facial image data are generally represented using the standard cylindrical coordinate system (ρ, θ, ζ) where the ρ = 0 axis runs vertically through the middle of the head, θ is the yaw angle, and ζ axis is the vertical axis. The point on the facial surface can be represented by a triplet (ρ, θ, ζ). It is easy to spread out the facial surface over R^2 as x = θ, y = ζ and z = ρ, as shown in Fig. 1.

2.2 Approximation Error

We follow the definition of the approximation error used in [7] in order to compare the performance. In the mesh optimization, Delauay triangulation for a polygonal mesh is employed. A certain configuration of points on R^2 is denoted by P_n and the corresponding Delanay triangulation is denoted by $D(P_n)$. Each Delaunay triangle $T_i \in D(P_n)$ contains a certain number of grid points (x, y). The Euclidean distance d_i at each such grid point (x, y) is defined as follows: $d_i \cong \left\| z(x, y) - \tilde{z}_i(x, y) \right\|$

where $\tilde{z}_i$ (x,y) is the linearly interpolated value of height at (x, y) determined by the triplet of hights for the three vertices of triangle T_i. The error e_i for T_i is the sum of squares of these Eulidean differences over all the grid pints P inside T_i: $e_i \equiv \sum_{P \in T_i} d_i^2$. The total error is then defined by

$$e = \sum_{T_i \in D(P_n)} e_i \qquad (1)$$

Since n is fixed, it is obvious that the total error becomes larger if too many points are located on relatively smooth parts of the face such as the forehead, or if too few points are placed on highly curved parts such as the ridge of the nose. Therefore, to reduce the error in approximation, we have to balance the number of points in low curvature regions with these in high curvature regions, because we use a linear interplant, namely a plane, for each unit of the triangular mesh.

3. Preliminary of crossover operators

Genetic algorithms (GAs) are commonly used evolutionary algorithms that employ parallel search to explore poorly understood, irregular spaces. Since GAs have good performance in solving optimization problems [7], they can also be used to solve the mesh optimization problem for surface approximation. The crossovers, main power of GAs, without and with maintaining feasibility for the constrained mesh optimization problems are described in Sections 3.1 and 3.2, respectively.

3.1 Crossover without maintaining feasibility

Because crossover produces new solutions by recombining the encoded solutions from a population, maintaining feasibility is difficult for many constrained problems. Generally, a penalty function approach and a repair operator approach are used to cope with the infeasible children problems.

(1) Bit string representation: Let α and β be parent pairs and each have a specific number n of 1's in a bit string chromosome with a bit string length N before performing crossover operations. Assume that the feasible solution has a number n of 1s in a chromosome. The crossover operation may generate infeasible children, described as follows: An integer number *s* between [1, N-1] is randomly generated. The children α' and β' are obtained by swapping all the bits of α and β after position *s* using a traditional one-cut-point crossover. For example, feasible parents α and β with N = 16, n=4 and *s* = 7 can generate infeasible children α' and β' as follows:

α = 0 1 0 0 0 1 0 1 0 0 0 0 0 1 0
α' = 0 1 0 0 0 1 0 0 0 0 1 1 1 0
β = 0 0 1 0 0 0 0 0 0 0 1 1 1 0 0
β' = 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0

(2) Bit matrix representation: A bit matrix m_{ij} with 0 ≤ i < N_x and 0 j< N_y|, for representing the 2-D mesh plane is encoded as a chromosome, where m_{ij} = 1 means that there exists a grid point at position (i, j) and otherwise, m_{ij} = 0 means that no grid point at position (i, j). Let α and β be feasible parent pairs and each have a specific number n of 1's in a bit matrix chromosome with size N = N_x x N_y before performing crossover operations. The crossover operation may generate infeasible children, described as follows: The crossover is done by randomly choosing a

horizontal or vertical line on the mesh plane that the line separates the plane into two parts. The children α' and β' are obtained by swapping all the bits of one part between parents. An illustration for explaining that feasible parents α and β with $N_x = 4$, $N_y = 4$ and $n = 4$ can generate infeasible children α' and β' is shown in Fig. 2.

3.2 Crossover with maintaining feasibility

One solution for preserving feasibility of children is to increase the complexity of the crossover. Therefore, some application-dependent crossovers have been proposed [10]. A generally used repair operator of GAs, used by Fujiwara and Sawai [7] for maintaining the feasibility of crossover is described as follows:

This operation incorporates random deletion and addition of grid points, which can always produce feasible offspring. An example is illustrated in Fig. 3. Feasible parents α and β can generate feasible children α' and β' using crossovers with a repair operator.

4. Proposed Efficient Genetic Algorithm

The proposed efficient genetic algorithm (EGA) uses a novel 2-D orthogonal crossover OAX based on the ability of OAs which are described in Section 4.1. Section 4.2 presents the chromosome representation. Section 4.3 introduces OAX. The illustration of 2-D OAX using a concise example is given in Section 4.4. The complete EGA is provided in Section 4.5.

4.1 Orthogonal arrays and factor analysis

Orthogonal array (OA) and factor analysis, which are representative methods of quality control [11], also work to improve the crossover more efficiently. The superiority of OA to achieve an intelligent crossover on obtaining the optimal solution has been demonstrated in our recent work [9]. The definition of OA used in OAX is as described follows. Let there be F factors and each factor have two levels. The number of total combination is 2^F. The columns of two factors are orthogonal when the four pairs, (1,1), (1,2), (2,1), and (2,2), occur equally over all experiments. When any two factors in an experimental set are orthogonal, the set is called an OA. To establish an OA of F factors of two levels, we obtain an integer $f = 2^{\lceil \log_2 (F+1) \rceil}$, build an orthogonal array $L_f(2^{f-1})$ with f rows and (f-1) columns, and select the first F columns. For instance, Table 1 shows the used nine columns of an orthogonal array $L_{16}(2^{15})$. Generally, level1 and level2 of a factor represent the selected genes coming from parent1 and parent2, respectively. Factor analysis can evaluate the effects of factors on the evaluation function, rank the most effective factors, and determine the best level for each factor such that the evaluation is optimized. Orthogonal experiment design can reduce the number of experiments for the factor analysis. The number of OA experiments for single factor analysis is only f. Let y_t be the positive function evaluation value of experiment no. t. Define the main effect of factor j with level k to be S_{jk},

$$S_{jk} = \sum_{t=1}^{f} Y_t^2 \times \begin{bmatrix} the\ level\ of\ Exp.no.\ t \\ of\ \ factor\ j\ is\ k \end{bmatrix}$$

where $Y_t = \begin{cases} y_t & if\ the\ function\ is\ to\ be\ \max. \\ 1/y_t & if\ the\ function\ is\ to\ be\ \min. \end{cases}$

and $$[condition] = \begin{cases} 1 & if\ the\ condition\ is\ true \\ 0 & otherwise\ , \end{cases} \quad (2)$$

Note that the main effect reveals the individual effect of a factor. The most effective factor j has the largest main effect difference (MED) $| S_{j1} - S_{j2} |$. If $S_{j1} > S_{j2}$, the level 1 of factor j is better than the level 2 on the contribution for the optimization function. Otherwise, level 2 is better.

4.2 Chromosome Representation

A configuration P_n on the 2-D grid plane is encoded as a chromosome. Let the size of the grid plane be $N = N_x \times N_y$. Construct a $M_x \times M_y$ matrix M of P_n where each element m_{ij} of M represents the number of grid points in a $l \times l$ block M_{ij} and $M_x = \left\lceil \frac{N_x}{l} \right\rceil$ and $M_y = \left\lceil \frac{N_y}{l} \right\rceil$. A chromosome is denoted as $(m_{11} \cdot m_{12} \cdots m_{1Mx} \cdot m_{21} \cdots m_{MyMx})$. It is noted that $m_{ij} \in [0, l^2]$. An example for the chromosome representation of a parent pair C_1 and C_2 is illustrated in Fig.4.

4.3 2-D orthogonal crossover OAX

Let the size of the grid plane be $N_x \times N_y$ and the number of grid points be n. The newly produced offspring after OAX are forced to contain the same number of grid points as their parents. The OAX procedure is as follows:
1.)Determine the maximal number a of segments using a -1 cut points such that the numbers of grid points in any two corresponding segments of the parents are identical and every segment has at least one grid points. Furthermore, no any two-segment pair from the parents has the same configuration on the grid plane. If $a = 1$, OAX will not be applied before performing the mutation operation. For example, let parents C_1 and C_2 are as follows. It can be determined that $a = 8$.

C_1 : (1120 10 1 1 10 100 10 1)
C_2 : (2101 01 1 1 10 100 10 1)

2.) Select the first a columns of an OA $L_\beta(2^{\beta-1})$ where $b = 2^{\lceil \log_2 (b+1) \rceil}$. Note that one segment in a chromosome is regarded as a factor in OA. Let level 1 and level 2 of factor *j* represent the j^{th} segments coming from C_1 and C_2, respectively. 3.) Evaluate the function value y_t for experiment no. *t* where $t = 1, 2, .., \beta$. 4.) Compute the main effect $S_{j\,k}$ where $j = 1, 2, .., a$ and $k = 1, 2$. 5.) Determine the best level for each segment. Select level 1 for the j^{th} segment if $S_{j1} > S_{j2}$. Otherwise, select level 2. 6.) The chromosome string of the first child is formed

from the best combinations of the better segment derived from the corresponding parents. 7.) Rank the most effective factors from rank 1 to rank a . The factor with large MED has higher rank. 8.) The chromosome string of the second child is formed similarly as the first child except that the segment with the lowest rank adopts the other level.

4.4 Efficient genetic algorithm

In this section, solving surface approximation problems using an efficient genetic algorithm (EGA) is described in detail.

EGA can be written as follows:

Step 1: Initiation: Randomly generate an initial population consisting of individuals I_i i =*1, 2,..*, N_{pop}.

Step 2: Elitist strategy : Repeat the following steps for i=*2* to N_{pop} :

2a: Select I_1 and I_i as the parents and produce two children I_{c1} and I_{c2} using 2-D OAX.

2b: Replace I_1 and I_i with the best and the second best individuals according to fitness performance among I_1, I_i, I_{c1} and I_{c2}, respectively.

Step 3: Evaluation: Evaluate the fitness function values for all individuals.

Step 4: Selection: Use the rank selection that replaces the worst P_s * N_{pop} individuals with the best P_s * N_{pop} individuals to form the new population, where P_s is the selection probability.

Step 5: Crossover: Select P_c * N_{pop} parents for 2-D OAX, where P_c is the crossover probability. Apply 2-D OAX to the selected pairs of parents. Replace the two children with two individuals having the best fitness function values among the parents and children for the elitist strategy.

Step 6: Mutation: Each located grid point coded in each chromosome, is randomly moved to one of the nearest neighbors on the grid with a probability P_m , called the mutation probability [7]. To prevent the fitness value from decreasing, mutation is not applied to the best individual.

Step 7: Termination test: If a prespecified termination condition is satisfied, end the algorithm. Otherwise, go to Step 3.

5. Experimental results and performance comparisons

In order to demonstrate the superiority of the proposed algorithm, we compare the performance of our method with that of Fujiwara' s genetic algorithm (FGA) [7], which has been shown to be an effective method for surface approximation problems using an evolutionary computation approach.

In Section 5.1, we illustrate the efficiency of the proposed 2-D OAX using a simple mesh optimization problem. Next, in Sections 5.2 and 5.3, we will show that EGA outperforms FGA using a 2-D mesh and a 3-D image, respectively.

The parameters of EGA and FGA are as follows: N_{pop} = 100, P_s = 0.04, P_c = 0.5, P_m = 0.18 and l=2.

5.1 Efficiency of 2-D OAX

We use an artificial 2-D binary array opt with N_x = 16 and N_y = 16 to generate a search space of C(256, 34) instances for the mesh optimization problem. The given binary array is served as an optimal solution of a mesh optimization problem, as shown in Fig. 5. In this case, the mesh optimization problem is how to efficiently select n = 34 right positions of 1' s in the array such that the matching error E_2 can be minimized. The matching-error E_2 of a mesh p is equal to a fitness function value f(p) for a encoded chromosome p . The fitness function value of P is defined as f(p) = the number of mismatched points between p and opt. To illustrate the use of 2-D OAX, a concise instance of OAX is given as follows. Assume that two chromosome C_1 and C_2 of configuration P_1 and P_2 are give in Fig. 6. Let D_1 and D_2 be children of parents C_1 and C_2. The maximal number a of segments for C_1 and C_2 is 9. Therefore, the first 9 columns of OA $L_{16}\left(2^{15}\right)$ are used. The results of performing a 2-D OAX on C_1 and C_2 are shown in Table 1 and Table 2.

The performance comparisons of EGA and FGA are shown in Fig. 7. It can be seen that EGA outperforms FGA in term of convergence speed and solution quality under the same number of function evolution, Fits, and generation G, as shown in Fig. 7(a) and 7(b), respectively.

5.2 Performance evolution using a simple function

In this experiment, we demonstrate the superiority of our algorithm using a simple function, f(x, y) = x*exp(-(x^2+y^2))*-0.5, where $-2.5 \leqq x \leqq 2.5$, y = x, and z = f(x, y), as shown in Fig. 8. In the surface, there is two height curvature on the ridge and in the valley. The data set is given on a grid plane with N_x = 30 and N_y = 30 and served as a 3-D image. The optimization problem for surface approximation is given with N = 900 and n = 160. Four points are located in the corners of the grid plane which are not subject to genetic operations. This maintains the boundary edges of the region during the two evolution processing.

Comparing the performance of EGA and FGA using Equ. (1), the convergence speed and accuracy are reported in Fig. 9. It shows that EGA outperforms FGA. Fig. 10 and Fig. 11 show the results of Delaunay triangulation and the reconstructed 3-D surface using triangular meshes after 50 generations. It is clear from the figures that the unnecessary locations of points in relative flat regions is avoided more points are on the ridge and valley.

Mesh optimization is an effective approach for surface approximation, as illustrated in the following aspects:

(1) While the fitness value e is quickly improved, a more accurate approximation surface can be obtained.

(2) More grid points are located in highly curved parts such as the ridge (top) and the valley

5.3 Performance evolution using a 3-D image

In this experiment, we examine the effectiveness of the proposed algorithm using a 3-D image, as shown in Fig. 12. The mesh optimization problem is the same as Section 5.2. The problem has a search space of C(N, n) instances with N = 40 X40 and n = 200 for FGA and EGA. The evolution results are shown in Figs. 13 – 15. From the simulation results, EGA is more superior to FGA in solving the large mesh optimization problem,

6. Conclusions

In this paper, the surface approximation using a mesh optimization approach is investigated. A GA-based method has been shown to be effective in solving the mesh optimization problems for surface approximation. This paper proposes an efficient genetic algorithm with a novel 2-D orthogonal crossover for obtaining the optimal solution to the surface approximation problem using a triangular mesh. It was shown empirically that the proposed efficient genetic algorithm outperforms the existing GA-based method in solving the mesh optimization problem in terms of the approximation quality and the convergence speed, especially in solving large mesh optimization problems.

References

[1] T. Ueshiba and G Roth, "Generating Smooth Surfaces with Bicubic Splines over Triangular Meshes: Toward Automatic Model Building from Unorganized 3D Points," Second International Conference on 3-D Digital Imaging and Modeling, pp. 302–311, 1999.

[2] H. Suzuki, S. Takeuchi, and T. Kanai, "Subdivision Surface Fitting to a Range of Points," Seventh Pacific Conference on Computer Graphics and Applications, vol. 322, pp. 158–167, 1999.

[3] F. G. Uler and O. A. Mohammed, "A 3-D Finite Element Mesh Generator for Complex Volumes," IEEE Transactions on Magnetics, vol. 3052, pp. 3539–3542, 1994.

[4] W. I. Gross, O. G. Staadt, and R. Gatti, "Efficient Triangular Surface Approximations Using Wavelets and Quadtree Data Structures," IEEE Transactions on Visualization and Computer Graphics, vol. 22, pp. 130–143, 1996.

[5] L. P. Rodriguez, "Surface Approximation of 3D Objects from Irregularly Sampled Clouds of 3D Points Using Spherical Wavelets," International Conference on Image Analysis and Processing, pp. 70–75, 1999.

[6] R. Grosso, C. Lurig, and T. Ertl, "The Multilevel Finite Element Method for Adaptive Mesh Optimization and Visualization of Volume Data," Visualization, vol. 563, pp. 387–394, 1997.

[7] Y. Fujiwara and H. Sawai, "Evolutionary Computation Applied to Mesh Optimization of a 3-D Facial Image," IEEE Transactions on Evolutionary Computation, vol. 32, pp. 113–123, 1999.

[8] M. Hrinivas and L. M. Patnaik, "Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms," IEEE Transactions on System, Man and Cybernetics, vol. 24, no. 4, pp. 656–667, 1994.

[9] Shinn-Ying Ho, L.-S. Shu, and H.-M. Chen, "Intelligent Genetic Algorithm with a New Intelligent Crossover Using Orthogonal Arrays," Genetic and Evolutionary Computation Conference, pp. 289–296, 1999.

[10] N. R. Pal, S. Nandi, and M. K. Kundu, "Self Crossover: a New Genetic Operator and Its Application to Feature Selection," International Journal of Systems Science, vol. 29, pp. 207–212, 1998.

[11] S. Taguchi and S. Konishi, Orthogonal Arrays and Linear Graphs. Dearbon MI: America Supplier Institute, 1987.

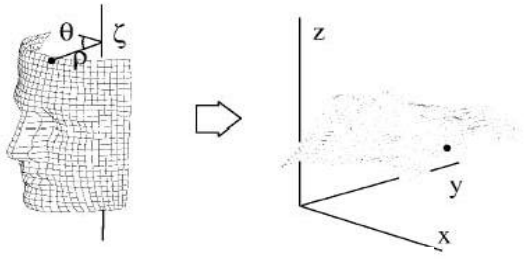
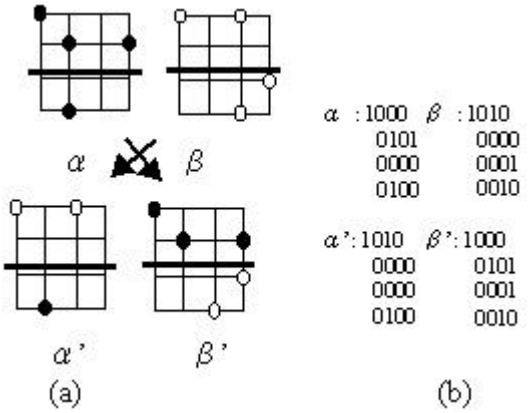


Fig. 1. Spreading he face over R^2 in [7].



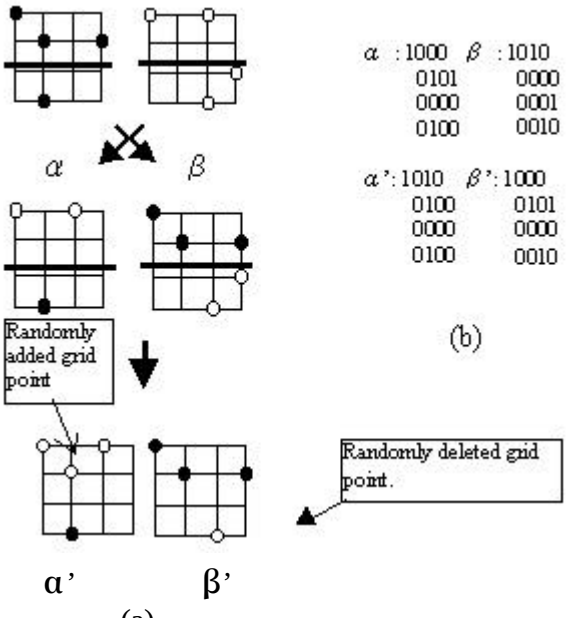
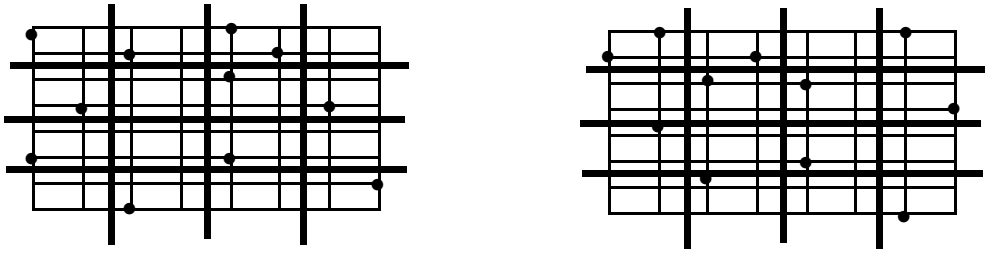


Fig.3. (a) Crossover with maintaining feasibility uses a repair operator. (b) The chromosomes correspond to (a).

Fig. 2. (a) Crossover swaps points of two parents over a chosen gridline.and (b) The chromosomes correspond to (a).



L_1: 1120 10 1 1 10 100 10 1 L_2: 2101 01 1 1 10 100 10 1

(a) (b)

Fig. 4 A configuration of a 8x8 gird plane with l = 2. (a) L_1: 1120101110100101 (b) L_2:2101011110100101.

Table 1. OA $L_{16}(2^9)$ and factor analysis.

	Factors										
Exp. No.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**$y_i(E_2)$**	
1	1	1	1	1	1	1	1	1	1	y_1	56
2	1	1	1	1	1	1	1	2	2	y_2	54
3	1	1	1	2	2	2	2	1	1	y_3	58
4	1	1	1	2	2	2	2	2	2	y_4	56
5	1	2	2	1	1	2	2	1	1	y_5	60
6	1	2	2	1	1	2	2	2	2	y_6	58
7	1	2	2	2	2	1	1	1	1	y_7	58
8	1	2	2	2	2	1	1	2	2	y_8	56
9	2	1	2	1	2	1	2	1	2	y_9	60
10	2	1	2	1	2	1	2	2	1	y_{10}	58
11	2	1	2	2	1	2	1	1	2	y_{11}	58
12	2	1	2	2	1	2	1	2	1	y_{12}	56
13	2	2	1	1	2	2	1	1	2	y_{13}	60
14	2	2	1	1	2	2	1	2	1	y_{14}	58
15	2	2	1	2	1	1	2	1	2	y_{15}	54
16	2	2	1	2	1	1	2	2	1	y_{16}	52
$S_{j1}(10^{-3})$	**2.469**	**2.469**	**2.566**	**2.387**	**2.566**	**2.566**	**2.469**	**2.387**	**2.474**		
$S_{j2}(10^{-3})$	**2.479**	**2.479**	**2.382**	**2.561**	**2.382**	**2.382**	**2.479**	**2.561**	**2.474**		
MED(10^{-6})	9.84497	9.84497	183.385	174.2441	183.3853	183.3853	9.845215	174.2441	0.704102		
Rank	**3**	**2**	**7**	**6**	**9**	**8**	**4**	**5**	**1**		

Table 2. Results of OAX

	1	2	3	4	5	6	7	8	9	E_2
C_1	010	1	001	0011	100001210010	010200100	101121100200010012100 0	010101	1010	56
C_2	001	1	001	0200	000000102102	000011011	000000010112101111022 1	020100	0002	58
D_1	001	1	001	0200	100001210010	010200100	000000010112101111022 1	020100	1010	52
D_2	001	1	001	0200	000000102102	010200100	000000010112101111022 1	020100	1010	54

0000000100000100
0000010000000000
1000000000000000
1000000000000000
0010000000000100
0000001101001000
0000001000000000
0000000001001001
0000000000000010
0000000000000000
0000001100000000
0010000001001001
0000000100000000
1001000111010011
0000000000000010
0001000000000001

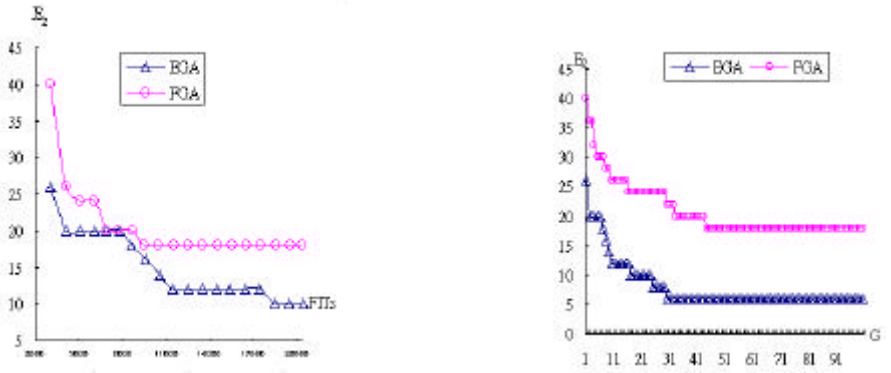
00100001 00001000
00000000 00000000
0000000000000000
0001101000000000
1010000000000000
0010100000100000
0100000000010000
0000110000000000
0000000110100000
1000100001001000
0001000000100000
0010000000000000
0010000000000001
0101010000000000
0000000000001000
0001000101000000

0000000100000100
0000010000000000
1000000000000000
1000000000000000
0010000000000100
0000001101001000
0000001000000000
0000000001001001
0000000000000010
0000000000000000
0000001100000000
0010000001001001
0000000100000000
1001000111010011
0000000000000010
0001000000000001

Fig. 5. A given 16x16 (a) (b)

C_1=(010 1 001 0011 100001210010 010200100 1011211002000100121000 010101 1010)
C_2=(001 1 001 0200 000000102102 000011011 00000001011210111110221 020100 0002)

Fig. 6. Two configurations. (a) p_1 (b) p_2 (c) Chromosomes C_1 and C_2



(a) (b)

Fig. 7. Comparison results of EGA and FGA.

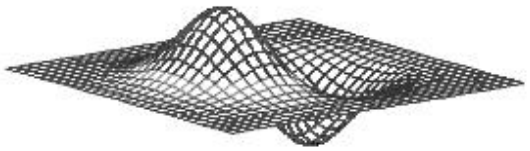


Fig. 8. A simple function mesh with a search space.(900,160)

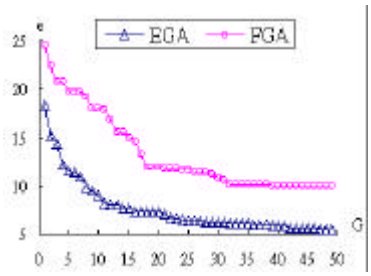
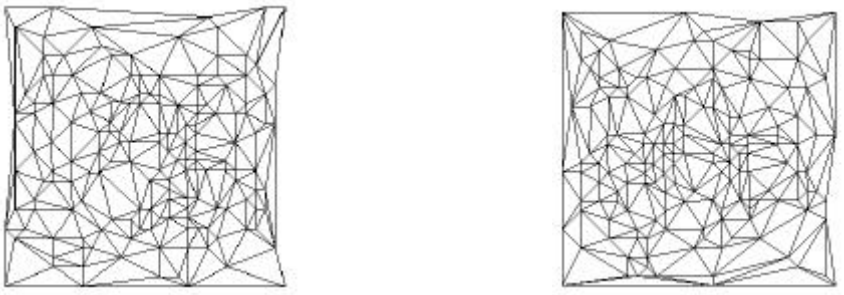
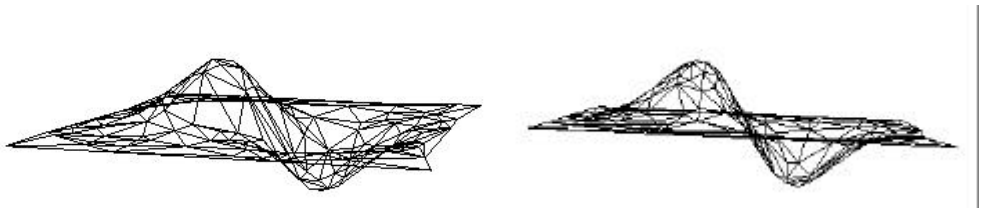


Fig. 9. The convergence speed and accuracy of EGA and FGA for fig.8.



(a) (b)

Fig. 10. The Delaunay triangulation after 50 generations using by (a) FGA and (b) EGA.



(a) (b)

Fig. 11. Reconstructed surfaces using trianglar meshes for approximating the surface of fig. 8, (a) FGA and (b) EGA.

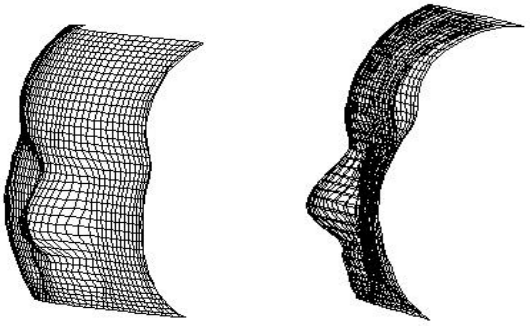


Fig. 12. A 3-D image in two different poses.

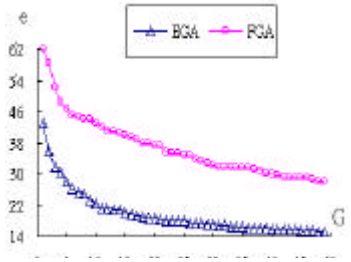
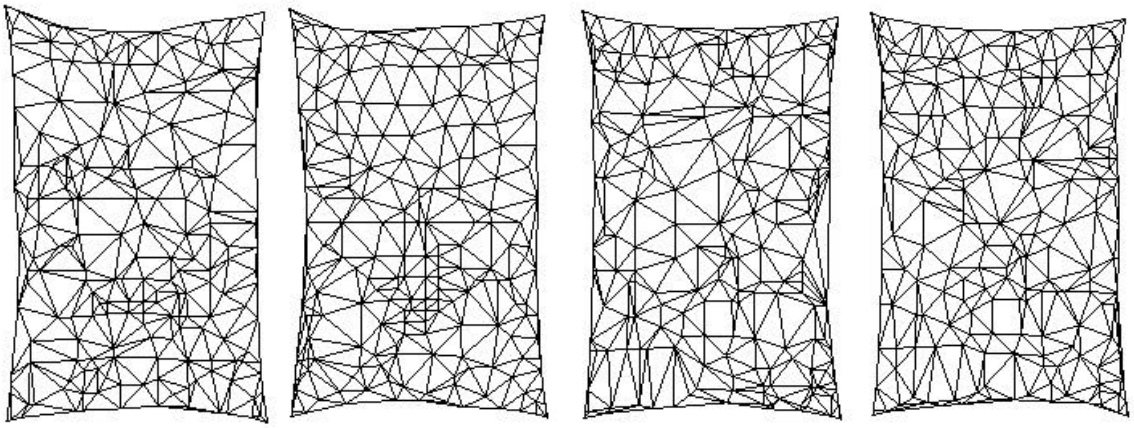
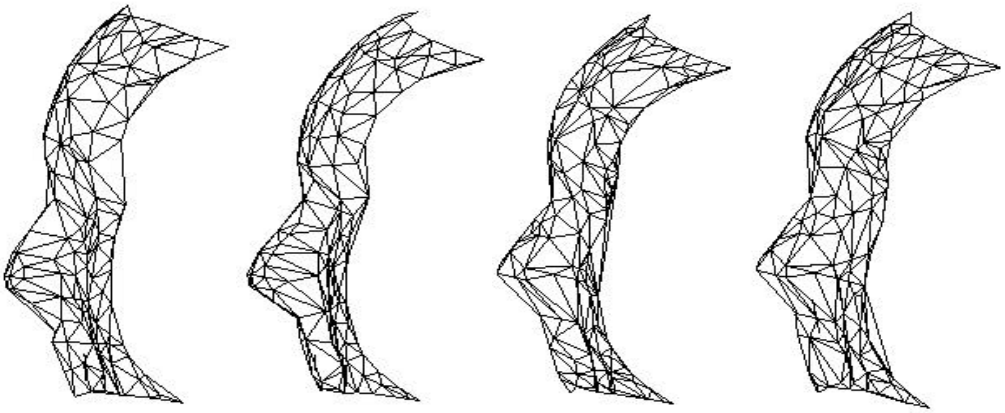


Fig. 13. The convergence speed and the accuracy of EGA and FGA. For



(a) (b) (c) (d)

Fig. 14. Location of points: (a) after 1 generation by EGA, (b) after 50 generation by EGA, (c) after 1 generation by FGA, and (e) after 50 generation by FGA.



(a) (b) (c) (d)

Fig. 15. The reconstructed 3-D images: (a) after 1 generation by EGA, (b) after 50 generation by EGA, (c) after 1 generation by FGA, and (e) after 50 generation by FGA corresponding to Fig. 14.