OVERLAPPING RANGE IMAGES USING GENETIC ALGORITHMS

Fernando Ortega, Javier San Juan, Francisco Serradilla, Dionisio Cortes

Abstract: *This work introduces a solution based on genetic algorithms to find the overlapping area between two point cloud captures obtained from a three-dimensional scanner. Considering three translation coordinates and three rotation angles, the genetic algorithm evaluates the matching points in the overlapping area between the two captures given that transformation. Genetic simulated annealing is used to improve the accuracy of the results obtained by the genetic algorithm.*

Keywords: *Range images, genetic algorithms, overlaping point cloud captures, genetic simulated annealing.*

Introduction

There are many different tools available on the market to convert range images taken from a three-dimensional grabber on representable models in real-time on a computer. However, a common characteristic in existing tools is the need for a large amount of manual work for the overlapping of the different captures that are required to register the whole figure.

In this paper, we present a novel solution which aims to automate the range image overlapping process using genetic algorithms (GA’s). The GA proposed will receive as input two partially overlapped range images and it must return the set of rotations and translations that must be performed on the second range image to ensure it is perfectly aligned with the first.

This paper’s main challenge is to design a GA that manages to resolve the problem. The size of the search space in which the GA moves is very large, and therefore, great care must be taken with the design of the algorithm in order to resolve the problem in a reasonable amount of time. The GA will mainly focus on the use of Genetic Simulated Annealing (GSA) to increase the GA convergence speed, and also on the design of an effective fitness capable of providing the overlapping level of two captures.

The rest of the paper is structured as follows:

- In Section 2 we will analyze the work related to the study presented here.
- In Section 3 we will formalize the proposed solution, explaining the GA developed in detail.
- In Section 4 we will present the experiments that we use to verify the performance of the GA.
- In Section 5 we will show the results of experiments with the algorithm developed.
- In Section 6 we will present the conclusions obtained and we will develop the possible future papers that have arisen as a result of this study.

Related work

The process of computer representation of real figures starts with the input of data from a series of range images taken by a grabber. From this series, we have to obtain one single range image in a unique reference system. In addition, we assume that each of the images captured may have been taken in random position and rotation conditions. For that reason, the objective of this phase is to find a series of transformations which, based on a random reference system for each of the captures, merges the information from all of these in one single reference system.

The main research into this field belongs to [1]. In this work, they tackle manual calibration techniques such as the overlapping of multiple captures with random transformations. They propose the use of an inverse calibration system, in which the viewing frame is associated with certain coordinates of the 3D scene (which can be performed via the pre-identification of certain elements captured on the view), allowing each of the capture points to be associated with certain coordinates of the scene.

[11] propose an interesting approach for registration in various phases. Firstly, they perform edge-based segmentation for broad overlapping which guides the next phase in fine-grain overlapping based on the ICP (iterative closest point) algorithm.

[10] covers techniques for the alignment of multiple capture points. He proposes the association of different capture points via a point-plane approach.

[3] design a new technique, called DARCES (based on the RANSAC technique) for the overlap of overlapped range images, using their own algorithms for the cataloguing of dispersed objects in range images.

[7] seeks an algorithm that allows the transformation matrix to be found between two systems of coordinates with different rotation. He calculates the rotation between the two systems in search of the appropriate quaternion.

The case we are dealing with has some peculiarities which make it difficult to apply the solutions proposed in the articles. On the one hand, it must be possible to find the overlapping areas without the need to resort to the use of markers. And, although using GPS and scanner compass information maybe of help when performing the process, the system could not depend on the availability of this data, but rather, on the comparison of the distribution of the points in each of the captures to be compared. Nor is it possible to resolve the problem by searching for a change of base matrix which overlaps one capture with the other, as it will be discrete areas of each capture which must be overlapped, and here lies the main problem to be resolved.

Genetic algorithm

Genetic algorithms (GA's) are a type of evolutive algorithms used to resolve search and optimization problems [5, 6]. They are based on simulating the evolutive process produced in nature to resolve problems of adaptation to the environment. GA's simulate, via populations of individuals, the evolution suffered through different operators called genetic operators, such as reproduction, crossover and mutation. Each operator plays a different role in the evolution of the population. This way, the reproduction can be understood as a competition, whilst the rest, like crossover and mutation are capable of creating new individuals from the existing ones in the population.

The individuals in our GA will represent the rotations and translations that must be carried out on the second capture to ensure it overlaps with the first. We have opted to use a GA with binary coding and we assign 6 chromosomes to each individual of the population, 3 for the translations and 3 for the rotations. The chromosomes of the translations have 8 bits, whilst those of the rotations have 12 bits. Figure 1 contains a graphical representation of the coding of the individuals and their chromosomes.

The genetic operators used in the GA are as follows:

- Selection. The chosen method is the Roulette Wheel Selection operator. That is to say, the selection probability of an individual depends on its fitness level.
- Crossover. We use the one-point crossover technique. Two parents are selected, a cut-off point is chosen and the two parents are combined to generate two children.

- Mutation. We use a single point mutation technique in order to introduce diversity. Some bits are modified randomly according to a mutation probability.
- Replacement operator: replacement with elitism. The best individuals of the current population move on to the next population without being modified (elitism) and the rest of the individuals are obtain using crossovers.

Since we have obtained quick satisfactory results using these four classical operators, we have not used other possible operators like migration, regrouping or colonization-extinction.

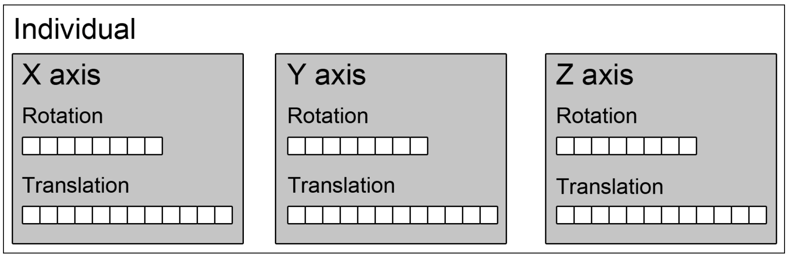

Figure 1. Representation of the individuals and their chromosomes of the GA developed.

Fitness function

In a GA the fitness function is fundamental as it allows the convergence of the population towards the optimum solution of the problem we want to resolve. In our case, we have developed a fitness function that allows us to check the level of overlapping of two captures simply and effectively. The basic idea is to divide the overlapped area into a series of cells and to count the number of points of each capture which belong to each cell. The cells that contain an equal number of points from both captures will have a high level of overlapping, whilst those that have a different number of points will have a very low level of overlapping.

Formally, in order to calculate the fitness we need to define the following elements:

- Let $C_1 = \{p_{1,1}, p_{1,2}, \ldots, p_{1,n-1}, p_{1,n}\}$ be the set of *n* points of the capture 1. (1)
- Let $C_2 = \{p_{2,1}, p_{2,2}, \ldots, p_{2,n-1}, p_{2,n}\}$ be the set of *m* points of the capture 2. (2)
- We define $p_{c,i}^{x}$ as the value of the coordinate of point $p_{c,i}$ on the X axis. (3)
- We define $p_{c,i}^{y}$ as the value of the coordinate of point $p_{c,i}$ on the Y axis. (4)
- We define $p_{c,i}^{z}$ as the value of the coordinate of point $p_{c,i}$ on the Z axis. (5)

We also need to define the zone in which both captures intersect:

- Let $I_{i,j}$ be the set of points belonging to the capture *i* which overlap with the capture *j*. (6)
- We define $\left[\min_{i,j}^{x}, \max_{i,j}^{x}\right]$ as the interval of the X axis in which the points of the intersection of capture *i* with capture *j* move. (7)
- We define $\left[\min_{i,j}^{y}, \max_{i,j}^{y}\right]$ as the interval of the Y axis in which the points of the intersection of capture *i* with capture *j* move. (8)
- We define $\left[\min_{i,j}^{z}, \max_{i,j}^{z}\right]$ as the interval of the Z axis in which the points of the intersection of capture *i* with capture *j* move. (9)
- Likewise, we define the operator $\#Set$ as the number of points of a set. E.g. $\#C_1$ represents the number of points of capture 1. (10)

We divide the volume overlapped in N^3 cells which will determine the level of overlapping of both captures in different places of the overlapped space. We define $cell_{1,2}^{r,s,t}$ as the set of points of capture 1 which are overlapped with capture 2 and which belong to the r, s, t cell.

$$\begin{aligned} cell_{1,2}^{r,s,t} = \{ p_{1,i} \in C_1 \mid & \min_{1,2}^x + r \cdot \left(\max_{1,2}^x - \min_{1,2}^x\right)/N \le p_{1,i}^x < \min_{1,2}^x + (r+1) \cdot \left(\max_{1,2}^x - \min_{1,2}^x\right)/N \\ & \wedge \min_{1,2}^y + s \cdot \left(\max_{1,2}^y - \min_{1,2}^y\right)/N \le p_{1,i}^y < \min_{1,2}^y + (s+1) \cdot \left(\max_{1,2}^y - \min_{1,2}^y\right)/N \\ & \wedge \min_{1,2}^z + t \cdot \left(\max_{1,2}^z - \min_{1,2}^z\right)/N \le p_{1,i}^z < \min_{1,2}^z + (t+1) \cdot \left(\max_{1,2}^z - \min_{1,2}^z\right)/N \} \end{aligned} \quad (11)$$

In the same way, we define $cell_{2,1}^{r,s,t}$ as the set of points of capture 2 which are overlapped with capture 1 and which belong to the r, s, t cell.

$$\begin{aligned} cell_{2,1}^{r,s,t} = \{ p_{2,i} \in C_2 \mid & \min_{2,1}^x + r \cdot \left(\max_{2,1}^x - \min_{2,1}^x\right)/N \le p_{2,i}^x < \min_{2,1}^x + (r+1) \cdot \left(\max_{2,1}^x - \min_{2,1}^x\right)/N \\ & \wedge \min_{2,1}^y + s \cdot \left(\max_{2,1}^y - \min_{2,1}^y\right)/N \le p_{2,i}^y < \min_{2,1}^y + (s+1) \cdot \left(\max_{2,1}^y - \min_{2,1}^y\right)/N \\ & \wedge \min_{2,1}^z + t \cdot \left(\max_{2,1}^z - \min_{2,1}^z\right)/N \le p_{2,i}^z < \min_{2,1}^z + (t+1) \cdot \left(\max_{2,1}^z - \min_{2,1}^z\right)/N \} \end{aligned} \quad (12)$$

With $o^{r,s,t}$ we denote the level of overlapping of the r, s, t cell. The idea is that the cells that contain many points from both captures have a high level of overlapping, whilst those which have a different level of points from both captures have a low level of overlapping. The calculation of the level of overlapping is reflected in Table 1.

Table 1. Level of overlapping according to the number of points of each range image in the r, s, t cell.

$o^{r,s,t}$	$\# cell_{1,2}^{r,s,t} = 0$	$0 < \# cell_{1,2}^{r,s,t} \le Few$	$Few < \# cell_{1,2}^{r,s,t}$
$\# cell_{2,1}^{r,s,t} = 0$	*LO*	*-LO*	*-HO*
$0 < \# cell_{2,1}^{r,s,t} \le Few$	*-LO*	*MO*	*-MO*
$Few < \# cell_{2,1}^{r,s,t}$	*-HO*	*-MO*	*HO*

Where *HO* represents a constant assigned to High Overlapping, *MO* a constant assigned to Medium Overlapping and *LO* a constant assigned to Low Overlapping. Likewise, *-HO*, *-MO* and *-LO* represent the same negated values to indicate Low Non Overlapping, Medium Non Overlapping and High Non overlapping, respectively. The value *Few* quantify the number of points that should have a range image to have a high grade of overlapping / non overlapping.

The fitness will be calculated as the summation of the overlapping of all the cells.

$$fitness = \sum_{r=0}^{N} \sum_{s=0}^{N} \sum_{t=0}^{N} o^{r,s,t} \quad (13)$$

Genetic simulated annealing

One of the problems of GA's is that, although they progress very well in the first stages of learning, they have difficulties with the fine adjustment of the final solution. In our problem, the fitting of scenes, this is translated into the rapid establishment of an approximated fit and a much longer delay to find the definitive value of rotation and translation of the fragments of the scene.

In order to improve this weakness of GA's, [2] theoretically proposed a new family of algorithms which combine the GA with a well-known technique in the field of optimization, Simulated Annealing (SA), in what they called Genetic Simulated Annealing (GSA).

Subsequent research has provided algorithms which mainly modify the operator selection of the GA to incorporate a temperature factor [4, 15], based on a neighborhood criteria among individuals which is gradually restricted as the temperature drops. This method reminds us of the operation of the neighborhood environment in self-organizing maps [8]. This approach has been used in various applications, among which we can highlight telecommunications network planning [14], economics [15], or forecast prediction [9].

In our paper, in order to reduce the computational cost of the search for the optimum fit of the fragments of the scene we have introduced the SA in the mutation and crossover operator. Given that we use a binary coding alphabet, we propose to use the temperature factor (*T*) of the annealing to rule out the *T* less significant bits in early stages of the process, and, as the temperature drops, to gradually incorporate more bits.

The effect of ignoring the less significant bits is to eliminate degrees of freedom from the search process in the solutions space, by sacrificing the precision in the adjustment obtained, which allows the algorithm to quickly converge towards the optimum solution. As more bits are incorporated to the mutation operator and the crossover operator the precision of the solution contributed is increased, ultimately reaching the precision required in the design in accordance with the total number of bits chosen to represent the values of position and orientation of the scenes.

Specifically, we propose that the mutation and crossover operator is applied to the most significant (*L -T*) bits of each rotation or translation, taking for *L* the number of bits of the translation / rotation and for T the value of the temperature represented by the number of bits to be ruled out. E.g. if the rotation has 12 bits, and the value of *T* is 4, the crossover and mutation operations will only be performed on the 8 most significant bits. The temperature is updated every certain number of generations (to be defined in each problem) increasing by 1 the number of significant bits.

In the same way, we will include the temperature concept in the fitness function. Our objective is that, as the GA advances, we increase the number of cells (*N*) into which the overlapped area is divided. This way, at the start, we will be able to carry out a poorly detailed adjustment of the captures and, as the generations elapse, the level of overlapping will be much more detailed. We must define an initial value for *N* and, every certain number of generations, we increase the value of *N* by one unit.

Experiments

In this section we are going to explain the experiments that we have used to test the algorithm. We have chosen two range images: a rosette with 528977 points and a house with 515255 points. The two captures were cut in half with a 38,5% overlap, which means that there is a 38,5% of the fragment in the other.

The fitness of the GA is highly dependent on the parameters. After several tests, we found optimal values for the proper functioning of the GA, this varies depending on the capture. The Table 2 shows the parameters used in the experiments.

Table 2. Parameters used in the experiments.

Parameter	House	Rosette
Number of generations	1000	1000
Population size	100	120
Crossover probability	0,7	0,7
Mutation probability	0,01	0,01
Elitism percent	5%	5%

Initial number of cells (*N*)	16	20
Number of generations to update *N*	50	50
Initial discarted bits (*T*)	6	6
Number of generations to update *T*	25	25
Few (points)	500	500
High Overlapping (*HP*)	0.001	0.001
Medium Overlapping (*MO*)	0.01	0.01
Low Overlapping (*LO*)	0.1	0.1

In order to see the evolution of fitness, we are going to calculate the maximum fitness and we observe how good the fitness is as the generations evolve. The maximum fitness is calculated when we cut the captures, at the end of the cut, the two captures are perfectly overlapped and in this moment the maximum fitness is calculated. The maximum fitness is used to normalize the graph, with different captures fitness results may vary, so it is a good idea normalize to compare and have a way to measure how good it is in different cases.

Results

The figures 2 and 3 show the evolution of the genetic algorithm, in a graphical way. The figure 2a and 3a shows the first state of the algorithm, with the two fragments in completely different positions. The figure 2b and 3b shows the firsts steps of the algorithm, the fragment is starting to aproach to a good solution, the algorithm is in the 200 generation. The figure 2c and 3c shows the lastest steps of the algorithm, the fragment is almost in its position, the algorithm is in the 600 generation. The figure 2d and 3d shows the result of the algorithm, the fragment is in a position near to the perfect position. The adjustment is better in the rosette than in the house because is a complex capture, although the results are still good and we can see a graph of a typical GA, evolving quickly at first, and later doing small adjustments.

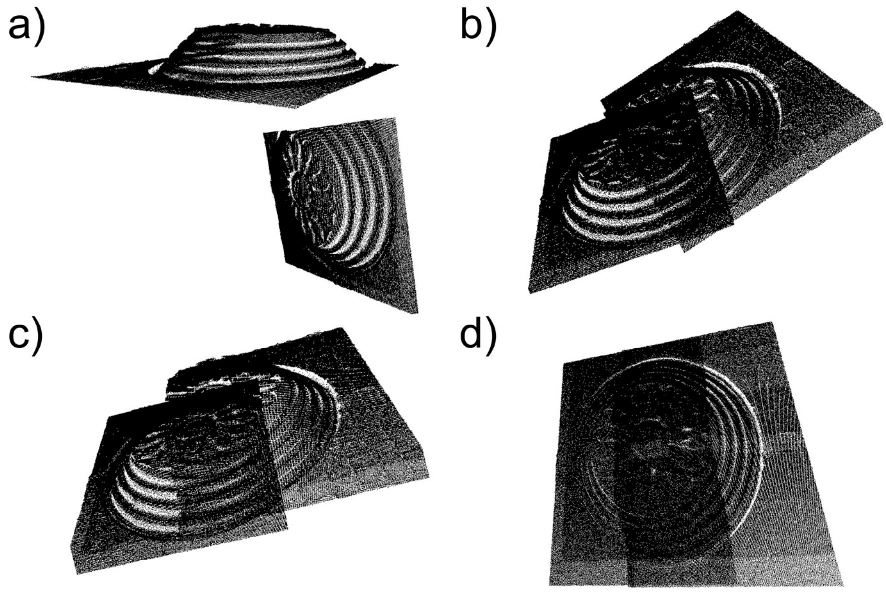

Figure 2. Results of overlapping the range image of the rosette with the proposed GA.

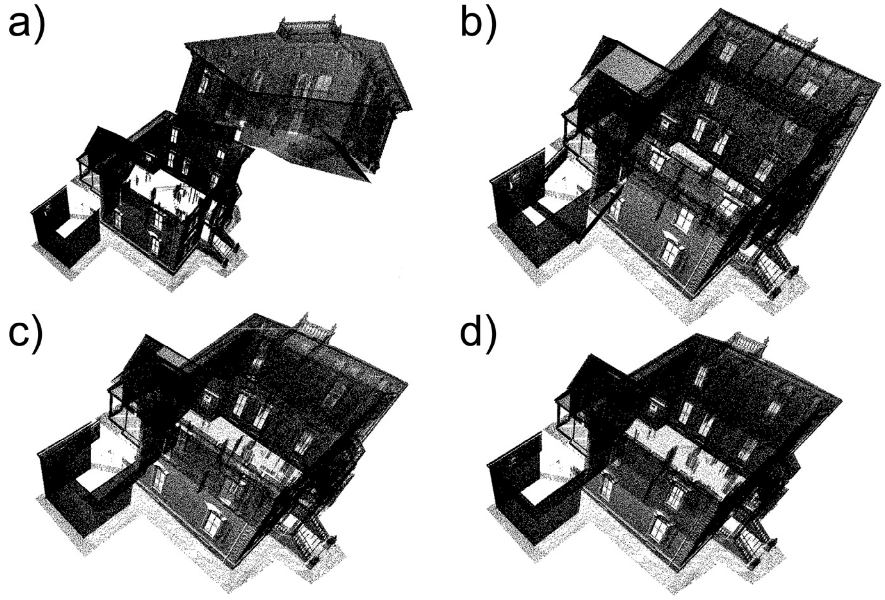

Figure 3. Results of overlapping the range image of the house with the proposed GA.

The figure 4 shows the evolution of the individuals in the genetic algorithm. The graph contains the average of the population normalized fitness in each generation. At first, the population evolves quickly and then, GA makes a slow and more precise adjustment. This is due to the nature of the GA, which at first have a quickly convergence and later have a slowly adjustment, and to the inclusion of the GSA which allows a better adjustment as the GA evolutions.

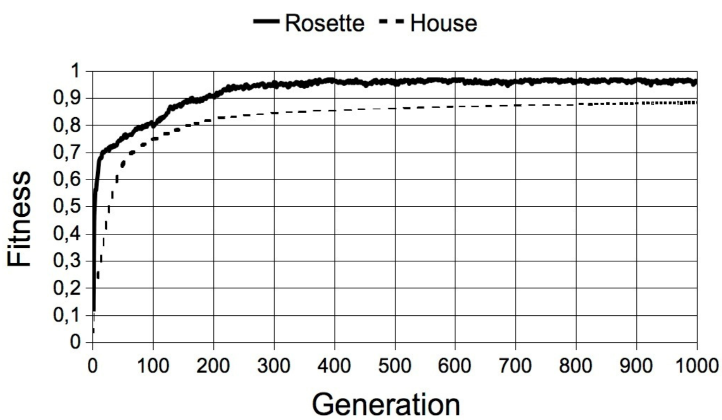

Figure 4. Normalized fitness evolution during de GA with de rosette and the house.

Conclusions and future work

This article presents a strategy to tackle the problem of overlapping two captures using GA's automatically instead of manually as has been the case until the now.

GA's are a great help for the resolution of many optimization problems. As we have seen, it is also possible to use them to resolve the problem of overlapping clouds of points, which is a highly complex problem as, a priori, we have no information about the placement of the captures and they can have random translations and rotations.

In the article we have presented a specific GA and we have placed special importance on including the GSA and on the development of the fitness function. After various experiments we have seen that the GA's behavior greatly depends on the parameters that we use for its execution. This fact highlights that, in order to achieve the overlapping between a pair of range images, we must be perfectly aware of the nature of these range images in order to correctly establish the parameters to be used. If the parameters are not adjusted correctly, the algorithm will converge towards suboptimal solutions and, therefore, towards erroneous overlapping.

Despite the impediment of the parameters, the proposed algorithm continues to present huge benefits compared to the manual overlapping of captures, as, with a little experience, the parameters are quickly adjusted and the algorithm is capable of returning the captures perfectly overlapped.

Normally, the scanners present geographical coordinates, as well as a compass for their orientation. In this paper, we have dispensed with any help as we cannot guarantee that all the captures presented contain this information, and, therefore, the proposed GA has an added value as it works with the minimum information possible on the range images.

In a future paper we will continue to study different fitness functions to see which offers better results. It is possible to establish other fitness functions which offer advantages as regards the precision of the algorithm or the time of its calculation.

It is more and more common to have equipment with multiple process cores, and, therefore, parallelizing these algorithms to reduce their execution time is essential. GA's are highly parallelizable. With little effort we could parallelize the algorithm to obtain faster response times on moving to the exploitation phase.

In this article, we use range images with the same density of points, and, therefore, another problem to resolve is the adaptation of the proposed GA to captures that present different point density. The current fitness is not capable of detecting the overlapping of two range images with different point density because it uses absolute constants to measure whether a capture has many or few points in a cell. It would be necessary to include variables related to the size of the capture to resolve this problem.

Finally, it is possible to replace the genetic algorithms with binary coding used with genetic algorithms with real coding, which can provide a greater convergence speed due to the variety of genetic operators available for these.

Acknowledgement

This project has been partially financed by the Ministerio de Industria, Turismo y Comercio Español through the program AVANZA (TSI-020110-2009-438).

Bibliography

1. Blais, G.; Martin D. L.: Registering Multiview Range Data to Create 3D Computer Objects, Centre for Intelligent Machines, TR-CIM-93-16, Center for Intelligent Machines, McGill University, 1993.
2. Chen H.; Flann, N. S.: Paralled Simulated Annealing and Genetic Algorithm: A Space of Hybrid Methods. *Paralled Problem Solving from Nature*, Vol. 3, pp. 428-438 (1994).
3. Chen, C.; Hung, Y.; Cheng, J.: RANSAC-based DARCES: A New Approach to Fast Automatic Registration of Partially Overlapping Range Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 11, pp. 1229 –1234 (1999).
4. Chen, H.; Flann, N.S.; Watson, D.W.: Parallel genetic simulated annealing: a massively parallel SIMD algorithm. *IEEE Transactions on Parallel and Distributed Systems*, Vol. 9, No. 2, pp. 126-136 (1998).
5. Davis, L. D.; Mitchell M.: *Handbook of genetic algorithms*. Van Nostrand Reinhold (1991).

6. Holland, J. H.: Adaptation in Natural and Artificial Systems. *University of Michigan Press, Ann Arbor* (1975).

7. Horn, B. K.: Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, Vol. 4, No. 4, pp. 629-642 (1987).

8. Kohonen, T.: Self-organizing formation of topologically correct feature maps. *Biological Cybernetics*, Vol. 43, pp. 59–69 (1982).

9. Lee, L.; Wang, L.; Chen, S.: Temperature prediction and TAIFEX forecasting based on high-order fuzzy logical relationships and genetic simulated annealing techniques. *Expert Systems with Applications*, Vol. 34, No. 1, pp. 328-336 (2008).

10. Pulli, K.: Multiview Registration for Large Data Sets, Stanford University. Proceedings. *Second International Conference on 3-D Digital Imaging and Modeling*, pp. 160-168 (1999).

11. Sappa, A. D.; Spetch, A. R.; Devi, M.: Range Image Registration by using an Edge-Based Representation. *In Proceedings of th 9th International Symposium on Intelligent Robotic Systems* (2001).

12. Stein, F.; Medioni, G.: Structural Indexing: Efficient 3-D Object Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 2 (1992).

13. Tavakolpour, A. R.; Mat Darus, I. Z.; Tokhi O., Mailah T.: Adaptive simulated annealing genetic algorithm for system identification. *Engineering Applications of Artificial Intelligence*, Vol. 9, No. 5, pp. 523-532 (1996).

14. Tsenov, A.: Simulated Annealing and Genetic Algorithm in Telecommunications Network. *Proceedings of the 7-th IEEE International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services - TELSIKS*, Vol. 2, pp. 555 - 558 (2005).

15. Wong, K. P.; Wong, Y.W.: Genetic and genetic/simulated-annealing approaches to economic dispatch. *IEEE Proceedings Generation, Transmission and Distribution*, Vol. 141, No. 5, pp.507-513 (1994).

Authors' Information

Fernando Ortega, Javier San Juan, Francisco Serradilla, Dionisio Cortes

Dpto. de Sistemas Inteligentes Aplicados, Universidad Politécnica de Madrid, Crta. de Valencia, Km 7. 28031 Madrid, Spain

{fortegarequena, javier.sanjuan, francisco.serradilla, dio.cortes.fernandez}@gmail.com