An Evolutionary Optimization Method for Parameter Search in 3D Points Cloud Reconstruction

Vitoantonio Bevilacqua[1,3,*] , Fabio Ivona[1,2],
Domenico Cafarchia[1], and Francescomaria Marino[1,4]

[1] Dipartimento di Ingegneria Elettrica e dell'Informazione, Politecnico di Bari,
via Orabona 4 – 70126 Bari – Italy
[2] Trait d'Union srl,
via R. Redi 3 – 70124 Bari – Italy
[3] eBIS srl,
via G. Petroni 25 (15/F4)– 70125 Bari – Italy
[4] APIS Apulia Intelligent Systems srl,
via P. Fiore 26 – 70125 Bari – Italy
bevilacqua@poliba.it

Abstract. Reconstruction of 3D laser scanned point clouds may generate a mesh characterized by a high number of triangles. Unfortunately, in Computer Aided Design environments neither a simple triangle reduction, nor decimation filters are feasible for mesh optimization, because of their intrinsic errors.

In this paper we show how Genocop III can be effectively used to reconstruct a point cloud bounding the error under a certain threshold. Moreover, we define an optimized algorithm for evaluating the reconstruction error, that exploits AABB-trees and pre-computation and provides a useful metric to the genetic algorithm.

Keywords: 3D Point Cloud, Reconstruction, Decimation, Hole Filling, Genetic Algorithms, Evolutionary Computation, Reconstruction Error, AABB-trees.

1 Introduction

Laser scanning of 3D surfaces allows capturing huge points cloud datasets that can be used in a Computer Aided Design (CAD) environments. After a preliminary data-cleaning and registration phase, a digital representation of the original surface can be computed through a process of surface reconstruction that generates a polygonal mesh, usually made of triangles. However, reconstructing a surface for further use in architectural CAD software introduces two more requirements:

- reconstructed mesh triangles count should be as lowest as possible;
- maximum error caused by reconstruction should be bounded by a pre-defined threshold.

* Corresponding author.

D.-S. Huang et al. (Eds.): ICIC 2013, LNCS 7995, pp. 601–611, 2013.

Usually, a surface optimization algorithm aims at reducing the triangles count to a specific value, while binding the consequent error under a specified threshold. In order to make a surface reconstruction algorithm feasible for architectural CAD environments, instead, we are mainly interested in minimizing the triangles count, while maintaining the maximum reconstruction error under a specified threshold.

Such a result is usually obtained by carefully searching and setting reconstruction and post-processing parameters. In this paper we investigate how to exploit genetic algorithms in order to automatically set the parameters, guaranteeing a low complexity mesh, paying a feasible error. Despite of the high efficiency of modern reconstruction techniques, an external hint is still required in order to point out the parameters used to generate and simplify the output mesh: our main contribution is to define an efficient and automatic algorithm to find the best ratio between reconstruction error and triangle count. Additionally, our algorithm allows the integration of different reconstruction, decimation and hole filling methods, helping to find the best configuration for the reconstruction system.

Moreover, since to define a way to compute the reconstruction error is not a trivial task (and it turns to be even more difficult, when executed within a genetic algorithm iteration), we propose a way to solve the 3D distance problem by means of a fast 2D algorithm, optimized for a huge number of iterations, which takes advantage of search trees and point-triangle distance techniques.

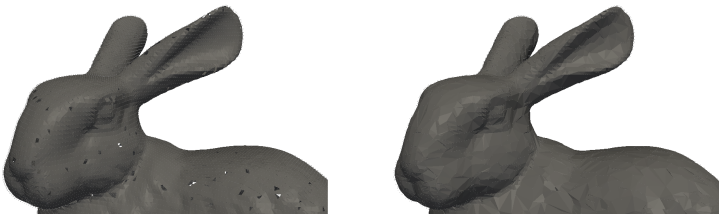

Fig. 1. Mesh reconstruction and optimization from 71787 to 24161 triangles

The paper is organized as follows. In section 2, we outline some mesh reconstruction, hole filling and decimation procedures that have included in our algorithm. Genocop III, an evolutionary genetic algorithm that handles non-linear constraints is then shortly recalled. In section 3, we describe the implementation of our algorithm for 3D surfaces reconstruction optimization. In section 4, we define a metric to provide an accurate estimation of the error generated by the reconstruction and decimation process. Experimental results, carried out both on syntethical benchmark and real world point clouds, are presented in section 5, evidencing the ability of the system to optimize the reconstructed mesh by finding the best parameter set. Conclusions are pointed out in section 6.

2 Background

Reconstruction and post-processing, starting from a points cloud, can be realized with a great number of algorithms, each characterized by strengths and weaknesses.

The choice of the algorithms and their parameters is usually committed to human operators, and this phase may require an intensive tuning step, in order to obtain an output mesh that minimizes the triangles count by satisfying the maximum error requirements.

2.1 3D Reconstruction

A typical surface reconstruction algorithm takes as input noisy points clouds that do not encode information on the original surface topology. We have considered the CPU-time and the quality of reconstruction as discriminating factors for the 3D reconstruction algorithm to adopt in our surface reconstruction procedure. This choice is due to the subsequent use of a genetic algorithm, which typically suffers time steps. In the following, we will explain the techniques that underlie the main steps of our surface reconstruction procedure. They are preliminary to the adoption of the genetic algorithm: in particular, after the reconstruction phase, the hole filling and the decimation filtering phases are necessary in order to achieve a low triangles count.

In [1], Hoppe proposes an approach based on a function that estimates the signed distance of each point from an unknown surface S. The key operation to define the distance function is to associate an oriented tangent plane with each point. The tangent plane T_i associated with the point x_i is built by a point o_i (center) and a normal vector n_i. Under this assumption, the distance from an arbitrary point p_k of the cloud to the plane T_i equals to:

$$dist_{T_i}(p_k) = (p_k - o_i) \cdot n_i$$

The subsequent phase aims to find geometrically close points, by checking if their corresponding tangent planes are consistently oriented. Starting from these ones, the distance function of p_i to an unknown surface can be computed using the oriented tangent planes: first, the algorithm finds a tangent plane T_i whose center o_i is the closest to p_i. The tangent plane is a local linear approximation of the surface S, so the signed distance $f(p)$ can be computed as the distance of p_i from the tangent plane T_i. The real surface can be described as the zero-set of the signed distance function. In a last stage, a marching cubes based contour tracing algorithm is used to approximate the zero set with a triangular mesh. The algorithm leads to the creation of an extremely dense initial mesh, that needs to be optimized in order to be feasible in a computer graphics environment. In order to achieve this result, Hoppe defines an automatic optimization procedure that aims to reduce the mesh complexity. The main target is to find a simplicial complex K and a set of vertex positions V that define a new mesh $M' = (K, V)$ by minimizing an energy function $E(K,V)$:

$$E(K,V) = E_{dist}(K,V) + E_{rep}(K) + E_{spring}(K,V)$$

where E_{dist} (distance energy) equals to the sum of the quadratic errors from the cloud to the mesh, and is calculated by identifying the nearest triangle to each point

and computing their distance; E_{rep} (representation energy) is introduced in order to penalize meshes with a high number of vertexes, and is proportional to a penalization factor defined by the user; E_{spring} (spring energy) is useful to ensure the convergence of the optimization algorithm.

In [2], Kazhdan et al. face the reconstruction problem by defining a 3D indicator function χ, defined as 1 for the points inside the surface and 0 for those outside. As the indicator function gradient is a vector field that has a non-zero value only in proximity of the surface, computing the indicator function reduces to the search for the scalar function χ whose gradient best approximates the vector field $\boldsymbol{V}$ defined by the points cloud. Applying the divergence operator, this variational problem is transformed into a standard Poisson problem of *computing the scalar function χ, whose Laplacian equals to the divergence of the vector field*

$$\mathbf{V}: \quad \delta_\chi \equiv \nabla \cdot \nabla_\chi = \nabla \cdot \mathbf{V}.$$

A different approach to the reconstruction problem is defined by the algorithm Greedy Surface Triangulation [3], based on the concept of surface growing. The mesh generation algorithm proceeds incrementally by searching, for each point P_i, a k-neighborhood made by the k nearest point from P_i within a sphere of radius $r = \mu \cdot d_0$, where d_0 is the distance from P_i to his closest point, and μ is a user specified constant. The neighborhood is then projected on a plane that is approximately tangential to the surface formed by the neighborhood. Projected points, whose visibility is occluded from P_i by the mesh edges are removed, and the remaining ones are triangulated, obtaining the final mesh. As this greedy approach proceeds incrementally, without deleting edges, the new surface grows directly leading to the final mesh, avoiding the memory consumption of the two other approaches.

2.2 Hole Filling and Decimation

The reconstruction process may cause the presence of holes in the mesh, so a hole filling filter is required in order to reduce the reconstruction error during the subsequent decimation phase. As in [4], the adopted hole filling algorithm proceeds by detecting boundary edge rings and associates neighborhood of points. Then, the neighborhood is projected on a tangent plane, and new vertexes are interpolated. In the last step, new triangles are computed with the Moving Least Squares algorithm [5].

Fig. 2. Mesh reconstruction and optimization from 93836 to 20795 triangles

Afterward, triangles decimation is performed. Decimation is based on the approach described in [6], that chooses which vertexes can be removed by classifying them in simple, complex and boundary. A simple vertex is surrounded by a complete triangle cycle, and can also be identified either as an interior vertex (if it is part of two edges that generate a large angle between the triangles they belong to, named feature edge) or a corner vertex (if it belongs to one or more than two feature edges). A boundary vertex is surrounded by a half cycle of triangles and a complex vertex is surrounded by a complete cycle of triangles and belongs to a triangle not in the cycle. Every vertex that is not a complex nor a corner vertex, can be removed and put into a priority queue, ordered by crescent values of the error implied by their removal. Finally, the queue is processed removing each vertex and re-triangulating the generated holes. This process is repeated until an optimization target is met.

2.3 Genetic Algorithms and Genocop III

Genetic Algorithms [7][8] are complex adaptive procedures, aimed to solve optimization problems in several real world applications, and are based on natural evolution principles. They work by selecting the best solutions for a given optimization problem, recombining them to build new generation, and converging towards the best solution. Genetic algorithms usually follow these main steps:

1. an initial set of possible solutions is defined;
2. each solution is evauated, and the best ones (in the sense of a given fitness function) are selected;
3. a new set of solutions is defined by manipulating the best solutions of the previous set. By this way, a good solution has better chances to reproduce itself, and to continue the evolution process. New solutions are obtained through mutation and crossover operations;
4. if a maximum iteration count is met, or the algorithm reaches an optimum solution, the optimization process ends. If not, the iterative process continues from step 2.

Usually, each solution (phenotype) is codified as binary code in chromosomes made of set of bits (genes). The evaluation of each phenotype is made through a fitness function that describes its attitude to solve the problem. Genetic Algorithms play a main role in the process of defining the best set of reconstruction and decimation parameters. Additionaly, they ensure that the parameter search converges towards the best solution faster than other optimization techniques. We have chosen to adopt Genocop III [9][10] (GEnetic algorithm for Numerical Optimization of COnstrained Problems), which supports nonlinear constraints, and is based on the concepts of coevolution and repair algorithms.

3 Optimized Reconstruction

Our target is to find a parameter set that allows to obtain a reconstructed mesh characterized by the lowest triangles count, and leads to an error no higher than the defined

error threshold. This can be formalized as an optimization problem with a single non-linear constraint, the maximum error.

In order to exploit the computational efficiency of Genocop III, we have defined two functions that can represent the evaluation function and the non-linear constraint. Chromosomes modeling is a propaedeutic step to the entire optimization process, and allows to apply mutation and crossover operators, in order to combine the best individuals and to obtain a population that converges towards the optimization target. The optimization process starts from an initial population random solutions. Each individual is uniquely identified by a specific value set for reconstruction and post-processing parameters, needed for the mesh generation. Each parameter has its own domain, defined in Genocop III configuration file. The evaluation function returns the triangles count of the reconstructed mesh. Therefore, this forces the algorithm to compute a reconstruction function *M* for each reference point. Given a reconstruction and decimation parameter set, tied to the current reference point $\overline{R}$, the evaluation function can be defined as:

$$eval(\overline{R}) = M(\overline{R}).triangle\_count$$

In order to bind the reconstruction error below the threshold, we have defined a non-linear constraint that takes advantage of the error function described in section 4. For a search point $\overline{S}$, each point of the original cloud is evaluated, measured its distance from the reconstructed mesh, and obtained the maximum quadratic error:

$$e^2_{\max}(\overline{S}) = \max\left\{dist(x_i, M(\overline{D}))\right\}$$

$e^2_{\max}$ is then used by Genocop III to evaluate if a search point $\overline{S}$ can be feasible and, thus, entered in the reference population.

3.1 Optimization Workflow

The optimization algoritm can search for the best reconstruction parameter set P_R, as well as the best decimation parameter set P_D. The optimization process is modeled as a unique execution of the genetic algorithm, in which each search point $S = \{P_R, P_D\}$ contains reconstruction and decimation parameters. At each evaluation, the algorithm launches the mesh reconstruction function and computes the decimation filter. The main target of this optimization problem is to find a parameter set $\left(\overline{P}_R, \overline{P}_D\right)$ such that

$$M(\overline{P}_R, \overline{P}_D).triangle\_count = \min_{(P_R,P_D)}\left\{M(P_R, P_D).triangle\_count\right\}$$

$$\wedge$$

$$\max_i\left\{dist\left(x_i, M\left(\overline{P}_R, \overline{P}_D\right)\right)\right\} < user\_defined\_error$$

4 Error Computation

While filtering reconstructed meshes, our optimization algorithm needs an estimation of the output surface quality. In order to make the generated mesh feasible for a computer aided design environment, a reconstruction algorithm must guarantee that the maximum error deriving by the decimation process does not exceed the specified threshold; under this assumption a simple error estimation, as the one used in common decimation algorithms, is not satisfactory. To face this problem, we have defined an error computation function that could be used by the genetic algorithm to check whether the generated population satisfies the maximum error constraint. This error metric is defined as the greatest distance between the cloud points and the generated geometry, i.e. the maximum deviation of the reconstructed mesh from the input point cloud. In order to compute the distance of a single point from the mesh, the error algorithm searches for the nearest triangle through an AABB-tree and a "point-triangle distance" is then computed.

As described by Jones in [11], the point-triangle distance could be obtained by projecting the point, say P, onto the plane of triangle T, and evaluating the position of the projection P_T. If P_T lies inside the triangle, the length $|PP_T|$ is the distance of P from T. If instead P_T falls outside the triangle, the point-triangle distance equals to the distance from P to the closest edge or vertex to P_T, depending of the projection position.

Starting from the optimized 2D method proposed by Jones, we implemented a point-triangle distance algorithm, that converts the problem into a two dimensional one and exploits pre-computation to achieve better computational performances. Our algorithm pre-computes a translation and a rotation matrix, to place the triangle so that V_1 lies on the origin, V_2 lies on the z axis and V_3 lies in the yz plane.

By this way we can reduce a 3D distance calculation into a bi-dimensional one, avoiding cross and dot products, and lowering the total number of computations needed to obtain the point-triangle distance. Transformation matrices are applied to translate and rotate the triangle and then are applied again to the point. P is then projected onto the plane of the triangle by setting its x coordinate to zero, obtaining P'.

Even though this process introduces an additional complexity to the distance computation process, transformation matrices allow to reduce the number of steps needed to obtain the distance, with computation times 94% lower than a matrix-free approach.

P' position is evaluated by computing its barycentric coordinates (u,v) respect of the translated triangle $V_1V_2V_3$. P' lies inside the triangle if

$$0 \le u \le 1, \quad 0 \le v \le 1, \quad u + v \le 1$$

In this case, the point-triangle distance is $d = |P.x|$. If P lies outside the triangle, the point-triangle distance equals to the distance of P from the nearest of the triangle edges. For the edge V_1V_2, its distance from P is computed by calculating the normalized projection of P' on it:

$$t = \frac{\overline{P'V_1} \cdot \overline{V_1V_2}}{\left\|\overline{V_1V_2}\right\|^2}$$

If $0 \le t \le 1$, the projection P'' lies within the segment $\overline{V_1V_2}$ and the point-triangle distance is $\sqrt{\left\|\overline{P'P''}\right\|^2 + \left|P.x\right|^2}$. If the projection P'' lies outside the triangle, the point-triangle distance equals instead to the distance from P to the nearest vertex (V_1 if $t < 0$ or V_2 if $t > 1$). Following the same process, the distance of P from V_1V_3 and V_2V_3 can be computed.

4.1 Error Computation Optimization

This phase requires two steps. A first step aims at reducing the point triangle distance CPU-time by applying a pre-computation process in order to reduce redundant calculations. A subsequent step allows a high decrease of the total number of distance evaluations by exploiting a binary search tree. By this way, for each point, the nearest triangle is identified and unnecessary distance computations are removed from the error estimation process.

Pre-computation: As more than one point usually share the same nearest triangle, the error algorithm can be optimized by pre-computing the transformation matrices as they depend on the triangle vertices, and remain constant when changing P coordinates. Additionally, barycentric coordinate equations

$$u = \frac{\left(V_2.y - V_3.y\right)\left(P'.z - V_3.z\right) + \left(V_3.z - V_2.z\right)\left(P'.y - V_3.y\right)}{\left(V_2.y - V_3.y\right)\left(V_1.z - V_3.z\right) + \left(V_3.z - V_2.z\right)\left(V_1.y - V_3.y\right)}$$

$$v = \frac{\left(V_3.y - V_1.y\right)\left(P'.z - V_3.z\right) + \left(V_1.z - V_3.z\right)\left(P'.y - V_3.y\right)}{\left(V_2.y - V_3.y\right)\left(V_1.z - V_3.z\right) + \left(V_3.z - V_2.z\right)\left(V_1.y - V_3.y\right)}$$

contain only four terms, depending on P', that need to be calculated at each iteration. Other terms are constant and can be reduced to a constant factor, reducing the complexity of the computation.

Search trees: Another optimization factor is introduced by a nearest triangle search algorithm. During the pre-computation step, the mesh is organized upon a hierarchic tree (AABB-tree [12]) whose leaf nodes contain a single triangle each. The partitioning process works by iteratively subdividing the space, binding each triangle to the partition that contains the center of its Axis Aligned Bounding Box (AABB). The subdivision process continues until the whole search tree is defined. Because of the AABB-tree depth, a hint mechanism has been adopted in order to allow to begin the search from an intermediate node, instead of the root node. A kd-tree [13] is built together with the AABB-tree, and is used to identify the group of triangles that likely

contains the nearest one from the computed point. Because of the construction technique used, the triangles group identified by the kd-tree search is contained in the same AABB-tree node. By this way, it is possible to start the AABB-tree search from an intermediate node, reducing the computational complexity of the search algorithm.

Using search trees, our algorithm can evaluate the maximum error by computing a single distance for each point of the cloud, reducing the total number of point-triangle distance computations.

5 Test and Results

The optimization procedure has been verified using the reconstruction algorithm Greedy Projection Triangulation [3], because it is faster and more efficient than the other solutions analyzed in section 2. Even if we focused our study on a set of three reconstruction methods, our optimization algorithm ensures a high compatibility with every reconstruction and post-processing technique, and can adapt itself to the user requests. The algorithm has been then tested on benchmark point clouds, managing to achieve a good triangle count reduction, without exceeding the maximum error threshold. An initial execution of the algorithm on the Stanford Bunny dataset, shows the optimization rate in respect of the maximum error imposed (Fig. 3).

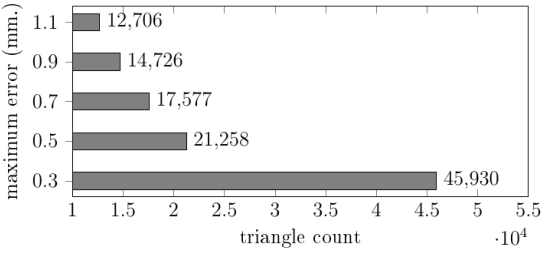

Fig. 3. Optimization rate

The maximum error of 0.3 mm is a lower threshold. Under this error value, our algorithm cannot optimize the mesh without breaking the precision requirement. Results for an error threshold of 0.5 mm are shown in Figure 1. Similar tests have been performed on different point cloud datasets, with increasing complexity. Our algorithm has been proved to be able to generate an optimized mesh with a low triangle count. Figure 2 evidences the presence of some reconstruction errors, due to the lack of points in some areas of the original cloud. The optimization algorithm interpolated the missing surface and applied the decimation filter in order to reduce by 80% the number of triangles. Figure 4 shows the reconstruction and decimation process starting from a high density point cloud. As it can be noticed, the reconstructed surface quality is high, with a maximum error lower than 0.2% of the point cloud width.

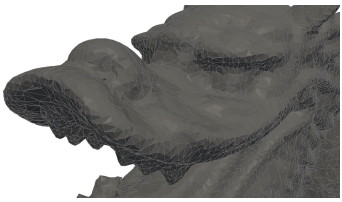

Fig. 4. Mesh optimization from 437645 points to 98857 triangles

A final test has been executed on a real world Trait d'Union company case project consisting of a 65506 points cloud scanned from the 3x6x5 m^3 environment of the *Auditorium Unità d'Italia, Isernia (Italy)*. The resulting dataset is characterized by not equally distributed points, huge areas without any information and a strong noise caused by the presence of a building yard. Starting from the point cloud, a very dense mesh has been reconstructed, obtaining 518626 triangles. The decimation algorithm managed to reduce the triangles count up to 186132, with a maximum error of 5 cm (Fig. 5).

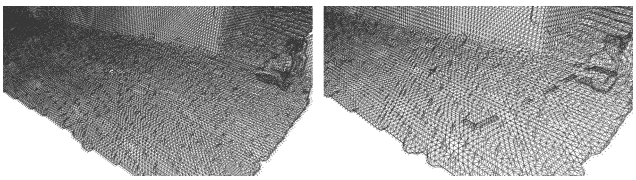

Fig. 5. Mesh reconstruction and optimization from 518626 to 186132 triangles

6 Conclusions

We have shown how genetic algorithms can be exploited in order to automatically obtain optimized mesh reconstruction parameters. The reconstruction process can fit within a computer aided design environment, defining an algorithm that complies with precision and triangle count constraints. Mesh optimization has proven effective with both low and high resolution point clouds, even if the resulting mesh quality is tied to the number of points in the original cloud and the maximum distance between them. In practical use, surface sampling with laser scanner introduces an intrinsic error that cannot be avoided, and it has to be taken into account, when defining the algorithm target error threshold. The main problem we had to deal with, was the high computational complexity of reconstruction algorithms that, in a genetic algorithm, become a strong factor of time consumption and get even worse when the number of points in the cloud increases. Starting from these assumptions, it is possible to define some areas of future research and development:

- further optimization the error computation algorithm with a better exploitation of search trees;
- point cloud partitioning, in order to execute the algorithm on smaller datasets and merge the optimized meshes;

- implementing a reconstruction and decimation algorithm oriented for an execution in a genetic algorithm;
- 3D geometric feature extraction and modeling of surfaces by using in a combined way genetic algorithms and Hough transform [14];
- porting the algorithm in a parallel environment (e.g., GP-GPU, FPGA).

Acknowledgements. This study was supported by the Italian PON FIT Project called "Sviluppo di un sistema di rilevazione della risonanza (SS-RR) N° B01/0660/02/X17" - Politecnico di Bari and AMT Services s.r.l. – Bari – Italy.

References

1. Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W.: Surface reconstruction from unorganized points. In: Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques, pp. 71–78 (1992)
2. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the 4th Eurographics Symposium on Geometry Processing, pp. 61–70 (2006)
3. Marton, Z.C., Rusu, R.B., Beetz, M.: On fast surface reconstruction methods for large and noisy point clouds. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 3218–3223 (2009)
4. Wang, J., Oliveira, M.M.: A hole-filling strategy for reconstruction of smooth surfaces in range images. In: Proceedings of the 16th Brazilian Symposium on Computer Graphics and Image Processing, pp. 11–18 (2003)
5. Lancaster, P., Salkauskas, K.: Surfaces generated by moving least squares methods. Mathematics of Computation 37(155), 141–158 (1981)
6. Schroeder, W.J., Zarge, J.A., Lorensen, W.E.: Decimation of triangle meshes. ACM SIGGRAPH Computer Graphics 26(2), 65–70 (1992)
7. Goldberg, D.E., Holland, J.H.: Genetic algorithms and machine learning. Machine Learning 3(2), 95–99 (1988)
8. Bevilacqua, V., Mastronardi, G., Piscopo, G.: Evolutionary approach to inverse planning in coplanar radiotherapy. Image and Vision Computing 25(2), 196–203
9. Michalewicz, Z., Nazhiyath, G.: Genocop III: A co-evolutionary algorithm for numerical optimization problems with nonlinear constraints. In: Proceedings of IEEE International Conference on Evolutionary Computation, vol. 2, pp. 647–651 (1995)
10. Sappa, A.D., Bevilacqua, V., Devy, M.: Improving a genetic algorithm segmentation by means of a fast edge detection technique. In: Proceedings of IEEE International Conference on Image Processing, ICIP, pp. 754–757 (2001)
11. Jones, M.W.: 3D distance from a point to a triangle. Technical Report, Dpt. of Computer Science, Univ. of Wales Swansea (1995)
12. Van der Bergen, G.: Efficient collision detection of complex deformable models using AABB trees. Journal of Graphic Tools 2(4), 1–13 (1997)
13. Bentley, J.L.: Multidimensional binary search trees used for associative searching. Communications of the ACM 18(9), 509–517 (1975)
14. Bevilacqua, V., Casorio, P., Mastronardi, G.: Extending Hough Transform to a Points' Cloud for 3D-Face Nose-Tip Detection. In: Huang, D.-S., Wunsch II, D.C., Levine, D.S., Jo, K.-H. (eds.) ICIC 2008. LNCS (LNAI), vol. 5227, pp. 1200–1209. Springer, Heidelberg (2008)