Efficient Reconstruction From Scattered Points

Helton Hideraldo Bíscaro
University of São Paulo, Brazil
EACH-USP
Av. Arlindo Bettio, 1000.
São Paulo - SP - Brazil
CEP: 03828-000
Tel.: 55 (11)-3091-1020
heltonhb@usp.br

Abstract

Most algorithms that reconstruct surface from sample points rely on computationally demanding operations to derive the reconstruction, beside this, most of the classical algorithm use a kind of three-dimensional structure to derive a two-dimensional one. In this paper we introduce an innovative approach for generating two-dimensional piecewise linear approximations from sample points in $\mathbb{R}^3$ that simplify significantly the numerical calculation and the memory usage in the reconstruction process. The approach proposed here is an advancing front approach that uses rigid movements in the three-dimensional space and a bidimensional Delaunay triangulation as the main tools for the algorithm. The principal idea is to use a combination of rotations and translations in order to simplify the calculations and avoid the three-dimensional structure used by the most of the algorithms. Avoiding those structures, this approach can reduce the computational cost and numerical instabilities typically associated with the classical algorithm reconstructions.

Keywords: Algorithm, Reocnstruction, Rigid Movements, three-dimensional structures, Delaunay triangulation.

1 INTRODUCTION

Given a set of samples P extracted from a smooth closed surface S in $\mathbb{R}^3$, the reconstruction problem consists in reconstruct F, a piecewise linear approximation of S, using the points of P. The surface F must be equivalent to S topologically and as close as possible to S.

In the last decades, surface reconstructions have been focus of extensive investigation not only because the number of practical applications in engineer and virtual museums but also by the challenges that need to be faced. In general only the three-dimensional coordinates of the points are known. Despite of lack of information about the topology and geometry, several algorithms has been proposed to solve this problem [8, 7, 9]. Some of the existing methods can even ensure a correct reconstruction as long as an adequate sampling rate is employed, such as those by Amenta *et al.* [2, 3]. However, in spite of considerable theoretical advances many algorithms fail to accomplish a successful reconstruction in practical situations.

In general, algorithms in literature use three-dimensional structures as Delaunay triangulations, or a kind of "immersion" three-dimensional space to derive a two-dimensional reconstruction. This paper introduces an advancing front approach, called LDT (Local Delaunay Triangulations) which runs entirely in two-dimensions. The main idea is to start from a boundary edge e and use the n nearest neighbors of one of end points of e to build a two-dimensional Delaunay triangulation in order to choose the better triangle to be glued in e. Avoiding those three-dimensional structures, not only the calculations are simplified, but also the amount of memory used is considerably reduced.

Prior to introducing the LDT algorithm, this work discuss related work in Section 2 and introduce some mathematical fundamentals required to lay out the proposed approach in Section 3. In Section 4 the reconstruction algorithm is described. Reconstruction results with LTD are given in Section 5. Finally, conclusions and further work are addressed in Section 6.

2 RELATED WORK

Surface reconstruction from sample points has deserved considerable attention from researchers in both Computer Graphics and Computational Geometry. The problem became popular after the paper by Hoppe et al. [23], who presented an algorithm for reconstructing the surface as the zero set of a signed distance function. However, that approach is unable to capture fine surface details. A related algorithm was developed by Curless and Levoy [14] that is more effective in capturing surface details; nevertheless, it relies on additional information than just the sample points. An alternative

WSCG 2010 conference proceedings, ISBN 80-903100-7-9
WSCG'2010, February 1 – February 4, 2010
Plzen, Czech Republic.

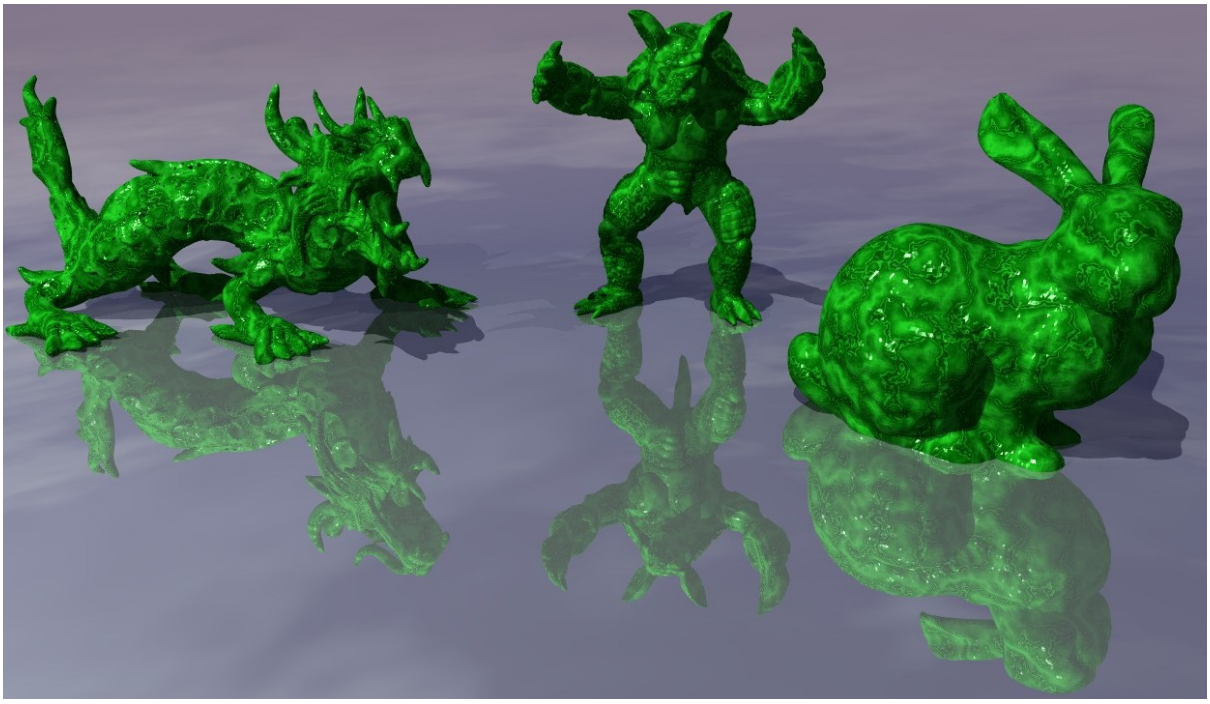

Figure 1: Models reconstructed with LDT algorithm

approaches for reconstructing a surface from the zero set of a distance function have been proposed. Carr et al. [13], for example, employ radial basis functions to approximate the signed distance. Their algorithm, though computationally expensive, can handle gaps and capture fine model details. Ohtake et al. [29] and Alexa et al. [1] use local fitting by employing partition of unit and moving least-squares approximation to estimate the approximating surface. The ability of handling large data sets is a major strength of such implicit approaches. However, the surfaces produced do not interpolate the given samples, which may be undesirable in some applications.

Researchers in Computational Geometry adopted a different approach towards the problem, some of them have proposing reconstruction algorithms based on a Delaunay complex generated from the sample points. The rationale behind such algorithms is to sculpt the surface from the Delaunay complex; others have proposed advancing fronts approaches. Boissonnat [11] proposed the first Delaunay based reconstruction algorithm, which operates by removing tetrahedral and triangles that violate certain geometrical conditions. Unfortunately, it is applicable only to surfaces of genus zero. The α-shape algorithm [17] starts with the Delaunay tessellation of the sample points and removes all simplices that are not contained in an empty ball of radius $\frac{1}{\alpha}$. The α-shape is simple to implement, but it works properly only on evenly sampled point sets, as a single α value applies to the whole data set. Teichmann and Capps [33] introduced a density scaled α-shape to handle this problem. Nonetheless, their approach requires the normal vectors at the sample points. The Crust, by Amenta and Bern [2], is the first three-dimensional algorithm with theoretical guarantees of reconstruction. For a suitably sampled object it computes a piecewise linear surface approximation that is homeomorphic and geometrically close to the original one. The Crust handles non-evenly sampled point sets and requires little user intervention during reconstruction. A drawback is that the geometrical calculations required to compute the Voronoi vertices introduce numerical instabilities. Furthermore, the algorithm has high computational cost because it builds two Delaunay tessellations, one to compute the Voronoi vertices and a second one to generate the Crust. The Cocone algorithm, by Amenta et al. [4], is an elegant and fast simplification of the Crust that holds the same theoretical guarantees. However, in practical applications it generates undesirable holes in the reconstructed surface. This problem has been solved by Dey and Goswami in the Tight Cocone algorithm [15]. Nonetheless, unlike its predecessor Tight Cocone does not capture internal components. Moreover, it requires pole estimates, cell labeling and, in some cases, triangle size estimates are also necessary. Power Crust [5] also improves on the Crust algorithm. It computes a piecewise linear approximation of a smooth surface employing a weighted Voronoi diagram called Power Diagram. Power Crust is also theoretically guaranteed to generate a correct reconstruction under proper conditions, and its computational performance is superior to that of the Crust. But it still faces numerical instability problems due to the geometrical calculations required to construct the Power

Diagram. Kolluri et al. [25] introduced the Eight Crust algorithm for reconstructing a watertight surface from noisy point cloud data. Starting from the Delaunay tessellation it uses a variant of spectral graph partitioning to decide whether each tetrahedron is inside or outside the original object. The reconstructed surface consists of the set of triangular faces shared by both internal and external tetrahedral. The spectral partition makes local decisions based on a global view of the model and therefore the algorithm can ignore outliers, patch holes and under-sampled regions. The high computational cost is still a major disadvantage.

The ball pivoting algorithm by Bernardini et al. [9] is very simple and fast. Three points form a triangle if a ball of user-specified radius touches them without containing any other point. Starting from a seed triangle, the ball pivots around an edge – i.e., it revolves around the edge while keeping in contact with the edge's endpoints until it touches another point, forming another triangle. The process proceeds until all reachable edges have been tried, and then it starts over from another seed triangle, stopping when all points have been considered. The process can be repeated with a ball of larger radius to handle uneven sampling densities. A major advantage of ball pivoting is that it does not compute the Delaunay tessellation of the sample points. On the other hand, it is user-dependent and needs the normals at the samples. Advancing front strategies have been employed in reconstruction algorithms by several authors, such as Schreiner et al. [30] and [31], but computational implementation of such methods can be quite intricate.

Edelsbrunner [16] derived an algorithm for fitting a surface to a set of sample points that relies on classical Morse theory. Although it relies on a topological background, topology is employed just to deduce the geometrical calculations. Another approach that uses Morse theory, in its discrete version is the work of Bíscaro *et al.* [10] which uses a discrete Morse function defined in a three-dimensional Delaunay triangulation to guide the reconstruction process. Also, the main drawback of this work is the three-dimensional structure required to extract a two-dimensional one.

Finally, Gopi *et al.*[20] has proposed a similar approach that uses a local Delaunay triangulation. However, their approach selects a set of candidate points which might be possible neighbors of a vertex in the final triangulation using a kind of sample criteria. They also compute the local Delaunay triangulation in the tangent plane without using any kind of simplification in its computation.

In fact, most of the classical algorithms derive the reconstruction from a subset of the three-dimensional Delaunay tessellation. This approach avoid to construct a three-dimensional structure to derive a two-dimensional piecewise linear approximation of the surface. Avoiding this immersion space, the algorithm presented here reduces the amount of ram memory used in the process as well as number of geometrical calculations. Another advantage of avoiding a three-dimensional Delaunay triangulation is absence of sliver tetrahedrons, which is a classical problem in three-dimensional triangulations.

3 BASIC CONCEPTS

This Section introduces the basic concepts and the terminology used in the remainder of the text.

A Delaunay triangulation for a set P of points in $\mathbb{R}^n$ is a triangulation of $DT(P)$ such that no point in P is inside the circumsphere of any simplex in $DT(P)$. In the plane, each vertex has on average six surrounding triangles; also, this triangulation maximizes the minimum angle. Compared to any other triangulation of the points, the smallest angle in the Delaunay triangulation is at least as large as the smallest angle in any other [21, 18].

Let S be a smooth closed surface in $\mathbb{R}^3$, i.e., S is C^1-continuous and divides $\mathbb{R}^3$ into open solids. A ball B is said to be empty (with respect to S) if its interior contains no point of S. The set of centers of the maximal empty balls touching S in at least two points make up the medial axis of S. The local feature size of a point s in S, denoted $lfs(s)$, is the distance from s to the medial axis of S. An important property of $lfs(\cdot)$ is that $lfs(p) \leq lfs(q) + |pq|$, where $|pq|$ is the distance between p and q. A set of points $P \subset S$ is an r-sample of S if the distance from any point $s \in S$ to the closest point in P is at most $r \times lfs(s)$. In this case S is said to be r-sampled; in general, good results in reconstructions are achieved for $r \leq 0.1$.

Quaternions (four numbers) are a kind of number system that extends the complex numbers. A quaternion number $q = (w, x, y, z)$, or correspondingly, $w + ix + jy + kz$, where hold the following identities; $i^2 = j^2 = k^2 = -1, ij = k - ij$ and $w, x, y, z \in \mathbb{R}$ [22]. They also provide a useful mathematical notation for representing and rotations of objects in three dimensions. When compared to Euler angles, they are simpler to compose and have an advantage of not present the problem of "gimbal lock". Also, they are more numerically stable a more efficient than rotations matrices. To represent a rotation of an angle θ around the axe n, a unit vector, is enough to define the quaternion $q = (cos(\frac{\theta}{2}), sin(\frac{\theta}{2})n)$.

This work also need an efficient and effective way of find the n nearest neighbors of a three-dimensional point p. To accomplish this, the work of Lin and Yang [27] was used. Their work offer a high accuracy nearest neighbor search by their ANN-Tree (Approximate Nearest Neighbor Tree) which is a tree based structure that works for arbitrary dimension.

Another important calculation present in this work is the angles between two vectors. According to Jonathan

Shewchuck [32], given two vectors with the same origin r and s, the best way of calculate the angle between r and s is to use the formula $tan(\theta) = \frac{2A_f}{\langle r,s \rangle}$, where A_f is the area of the triangle with sides r and s. The computation of A_f can be done making $A_f = \frac{|r \times s|}{2}$, where $r \times s$ is the cross product of r and s.

This paper uses the Hausdorff distance to compare the meshes generated by the LDT algorithm and the meshes of the classical algorithms. Hausdorff distance is a generic technique that defines a distance between two nonempty sets; and has been used as an efficient tool to evaluate distances between three-dimensional meshes [6].

In next section, this paper presents details of the algorithm developed in this work.

4 ALGORITHM

The algorithm LDT uses the normal vector at each sample point. There are several strategies to estimate this vector, but is important to include, in such estimation, the impact of the point's distance. The influence of the sample points must be inversely proportional to its distance. This work uses weighted principal component analysis (WPCA). The weight average of a point $p \in \mathbb{R}^3$ is given as follow:

$$M(p) = \sum_{i=1}^{n} \frac{w_p(p_i)\, p_i}{\sum_{i=1}^{n} w_p(p_i)} \tag{1}$$

where n is the number of the nearest neighbors of p, the function $w_p(x)$ specifies the influence of the point x in point p. According to Levin [26], a good choice is $w_p(x) = e^{\frac{\|x-p\|}{H^2}}$, where H estimates the local density in p, $H = \sum_{i=1}^{n} \frac{\|p_i - p\|}{n}$. The 3×3 covariance matrix C for a point p if given by:

$$C = \begin{pmatrix} p_1 - M(p) \\ p_2 - M(p) \\ \vdots \\ p_{n-1} - M(p) \\ p_n - M(p) \end{pmatrix}^T \begin{pmatrix} p_1 - M(p) \\ p_2 - M(p) \\ \vdots \\ p_{n-1} - M(p) \\ p_n - M(p) \end{pmatrix} \tag{2}$$

Let $\lambda_1 \leq \lambda_2 \leq \lambda_3$ be the three eigenvalues of C, and α_1, α_2 and α_3 the three associated eigenvectors. Jolliffe, in his work [24] establish that α_3 is the direction of greatest variance in a neighborhood of p, α_2 represents the direction of second greatest variance and α_1 the direction that minimizes the variance. As the set of points P is a subset of the surface S, the geometric interpretation is that α_2 and $\alpha 3$ approximates the main directions of the tangent plane at p and α_1 approximates the normal direction.

```
   Data: A set of samples P ⊂ R^3
1  for each p ∈ P do
2  |  Approximate the normal vector in p
3  end
4  Find a initial triangulation F ;
5  Store in E the boundary edges of F ;
6  while E ≠ ∅ do
7  |  Remove e from E;
8  |  if e still is a boundary edge then
9  |  |  f ← FindNewFace(e);
10 |  |  F ← F ∪ {f};
11 |  |  Add to E the boundary edges of f;
12 |  end
13 end
14 return F
```

Algorithm 1: Algorithm LDT - Local Delaunay Triangulation

The main idea in the LDT algorithm is to execute an advancing front approach to achieve the reconstruction. This advancing front technique uses a two-dimensional Delaunay to get the next triangle from a boundary edge. The pseudo-cod 1 shows the main loop of the algorithm developed here. In the step 2, the weighted principal component analysis is used to approximate the normal vectors in the samples points. The initial triangulation (step 4 in the algorithm 1) is acquire choosing an initial point p, projecting its n nearest neighbors in its tangent plane, computing the Delaunay triangulation in the plane and re-projecting the triangulation in the surface. The two-dimensional Delaunay triangulation was implemented using the only the first and the second coordinates of the samples. Considering this, the algorithm must rotate p and its neighbors such that the tangent plane in p coincide with the XY plane. By doing this rotation, the projection operation is expressively simplified. The Figure 2 illustrates this initial step, showing a normal vector in an initial point of a paraboloid, the blue points are the nearest neighbors of the initial point and the orange plane is the tangent plane where the neighbors are projected.

The set E store the boundary edges of the triangulation F and can be interpreted as a list of active edges that guide the reconstruction process. The main loop of the algorithm is repeated while E is not an empty set. It is worth to mention that when an edge e is removed from E, it is possible that e is not a boundary edge anymore. Also is possible that a new face f returned in the step 9 of the algorithm 1 has no boundary edges.

The pseudo-code 2 illustrates the procedure to expand the frontier of F, and is a variation of the procedure to achieve the initial triangulation. The idea is also to project the neighbors in the tangent plane π, execute a two-dimensional Delaunay DT triangulation with the

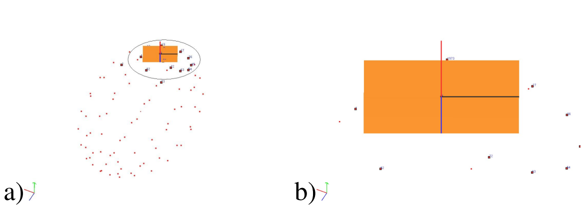

Figure 2: a) Set of samples of a 3D object b) Zoon view of the initial point.

Data: A boundary edge e.
1 Let p be one of the end points of e;
2 Rotate p and its n nearest neighbours to align the normal in p whith the Z axe and the edge e with Y axe ;
3 Find p_e, the opposite vertex to e;
4 Project in the tangent plane only the vertex p_i such that $p_{e_x} * p_{i_x} \leq 0$;
5 Find DT, a two-dimensional Delaunay triangulation with the projected vertices ;
6 Find f, the triangle of DT that contain e as a boundary edge ;
7 **return** f

Algorithm 2: Algorithm FindNewFace

project points, to choose from DT, the face f that has e as boundary edge, and re-project f in the surface.

To ensure that the projection of e appear in the local triangulation, consider $f_e \subset F$ the face of F containing e, and p_e the vertex of f_e opposite to e. When the vertex p, which is one of the end points of e, and its nearest neighbors are rotate to align the edge e with the Y axe, the x coordinate of p_e is either positive or negative depending of its relative position. After that, only the neighbors that has x coordinate with opposite signal when compared with p_e are projected in π. This procedure is enough to ensure that the projection of e appear in the boundary of the Delaunay triangulation DT. It is worth to mention that at this point of the algorithm (step 4 of the algorithm 2), only the boundary vertex in the neighborhood of p or vertices that are not contained in a face are considered to be projected in the tangent plane π.

Two steps of a paraboloid reconstruction can be seen in the Figure 3 a) and b). The distinct face represents the last face glued in the mesh and the wider edge represents the first edge in the list E of active edges. In the Figure 3 c) the complete reconstruction is showed. The figure 4 exhibit a local Delaunay triangulation's example for a set of sample points, and again, the distinguished face is the one captured to be re-projected in the surface.

Although this algorithm does not need to handle sliver tetrahedral, which is a very common problem in sculpturing techniques, it is possible that the algorithm

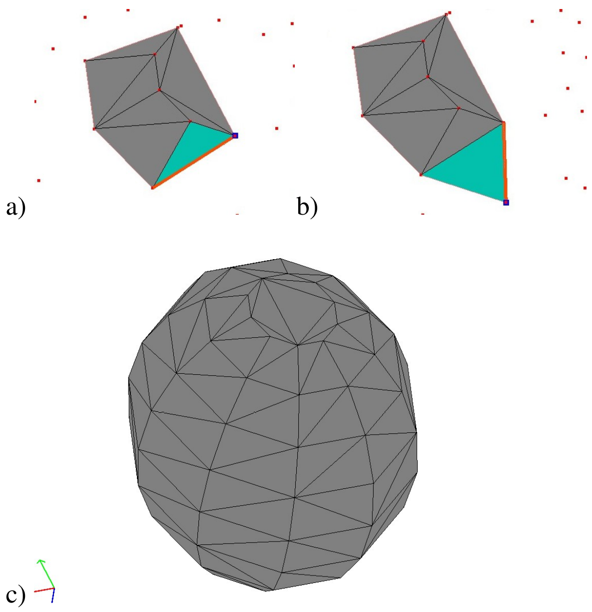

Figure 3: Two steps in the reconstruction of a paraboloid

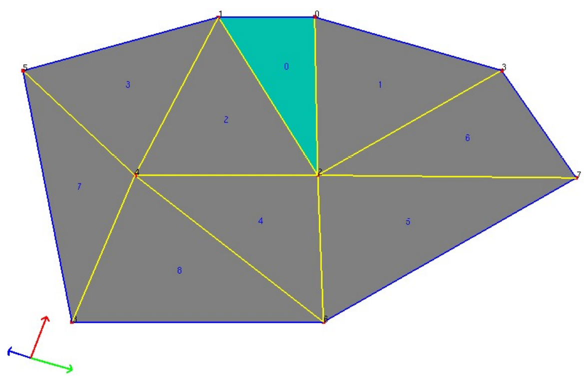

Figure 4: Local Delaunay triangulation to a set of sample points

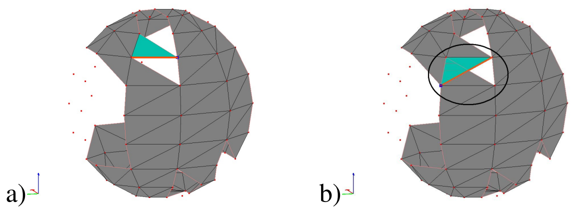

Figure 5: Small dihedral angles

glue faces with small dihedral angles as is showed in the Figure 5 The Figure presents two consecutive steps of the paraboloid reconstruction. However, according to the work of Mederos *et al.* [28], the dihedral angle between two adjacent faces approximates to π when the sample rate increases. To avoid this problem, a dihedral angle calculation, given by the work of Jonathan Shewchuck [32], must be done before glue a new face in the mesh.

4.1 Discussions

There are some crucial points in the LDT algorithm. The estimation of normal vectors in the samples points for instance, plays a crucial role in all reconstruction process (algorithm 1 step 2). Of course that the LDT algorithm does not have intention of reconstructs arbitrary surfaces with arbitrary sampling rate. In order to achieve a good normal estimation in all samples; is acceptable that a minimum sampling rate be respected. However, this is a theoretical study that will be subject of a future work. Another consideration, is about the projection effect over the algorithm's results. According to Amenta's work [3], for an adequate sample rate, in general, a $r-$sampled surface with $r \leq 0.1$, the correct reconstructions lies in a subset of the Delaunay complex of the samples points. Therefore, respecting this sampling condition, for an arbitrary sample point p, its neighbors must lie close to the tangent plane in p; not causing ample movements in the projection operation as well.

It is also worth to mention that in the initial triangulation (cod:algorithm step 4) no glue operation is needed. Therefore, the Delaunay triangulation computed I this step can be re-projected directly in the output surface.

The next section presents some of results obtain with the LDT algorithm.

5 RESULTS

This section shows some examples of models reconstructed with LDT algorithm as well as some comparisons with classical algorithms in the literature. For the comparisons was used in-house implementations, based on CGAL [12], of the Crust and Power Crust developed as part of a master dissertation project [19]; the TSR implementation was part of a previous work [10] and the original implementations of Cocone and Tight Cocone were kindly provided by Tamal Dey. The reconstructions were performed on a dual Pentium 4 with 3 GHz and 1GB RAM.

The figure 6 give an idea of the quality of the mesh generated by the algorithm LDT in a reconstruction of a bitorus. The Figures 7 and 8 show additional reconstructions examples, the dragon model is rendered with a jade texture and the hand model with a stone texture, and the Figure 10 shows the Lucy model reconstructed from a large data set (921085 points).

In the table 1 the usage of memory, in Kbytes, of some classical algorithm is exhibit. The algorithms are Crust, Power Crust, Cocone Tight Cocone, TSR and that of LDT, for a set of standard sample sets, identified in the top Table line (models shown were generated with LDT). The Crust and the Power Crust Algorithm produces no output to the Isis model, the fourth in the table. The Figure 9 represents the running times, in seconds, of three traditional reconstruction algorithms,

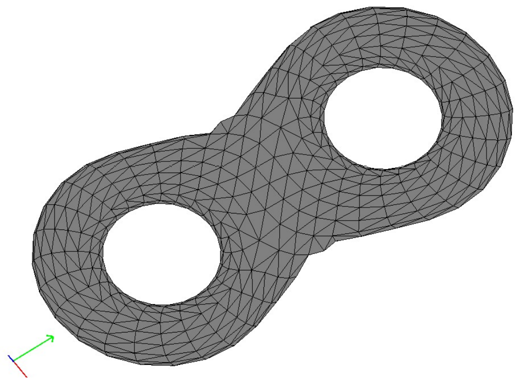

Figure 6: Mesh generated with LDT algorithm

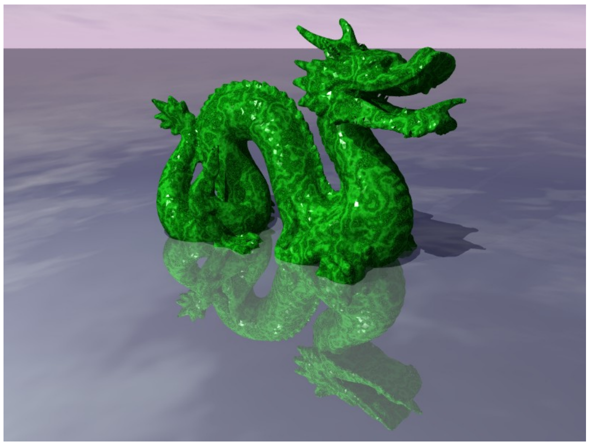

Figure 7: Dragon Model generated with LDT algorithm and rendered with a jade texture

Figure 8: hand Model generated with LDT algorithm and rendered with a stone texture

Cocone, Tight Cocone , TSR and LDT. As the table 1 reveal, the LDT algorithm, due its optimizations and its advancing front approach, lean to use less memory than the others. One observes that the running times were improved, particularly when the models are bigger than 50,000 points. One can see in Figure 9 that the LDT algorithm is faster than the classical algorithms compared with it, especially when reconstruct larges data sets.

	 9697 pts	 35947 pts	 54707 pts	 187644 pts
Cr	21.90	53.13	72.21	No output
PC	41.94	79.50	110.54	No output
Co	15.10	51.24	78.98	265.32
TC	19.20	66.10	101.68	343.50
TSR	20.65	71.28	111.72	376.19
LDT	12.48	38.75	59.17	192.22

Table 1: Usage of memory in k bytes to reconstruct the models shown Legend: Cr - Crust; PC - Power Crust; Co - Cocone; TC - Tight Cocone; TSR - Topological Surface Reconstructor; LDT - Local Delaunay Triangulation

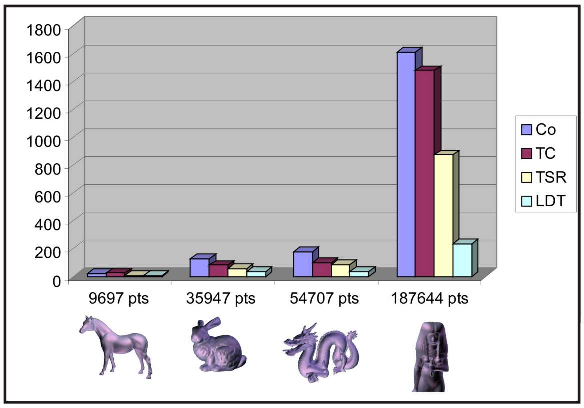

Figure 9: Running time in seconds to reconstruct the models shown Legend: Co - Cocone; TC - Tight Cocone; TSR - Topological Surface Reconstructor; LDT - Local Delaunay Triangulation

The table 2 exhibits the Hausdorf distances between the LDT' meshes and meshes generated by other classical algorithms (cocone, Tight-cocone and TSR). The distances are quite small; suggesting that the output meshes are very similar.

	 9697 pts	 35947 pts	 54707 pts	 187644 pts
Co	0.003235	0.001485	0.213477	0.000561
TC	0.002139	0.001181	0.446078	0.000525
TSR	0.002480	0.001018	0.006002	0.000074

Table 2: Hausdorff distance between the meshes generated with LDT algorithm and the follow ones : Co - Cocone; TC - Tight Cocone; TSR - Topological Surface Reconstructor

Figure 10: Lucy model reconstructed with LDT

6 CONCLUSION AND FUTURE WORK

This work introduces an innovative approach, called LDT - Local Delaunay Triangulation, to reconstructing piecewise linear approximations of surfaces in $\mathbb{R}^3$ that are defined by set of samples. The approach present here is an advancing front approach which make use of a two-dimensional Delaunay triangulation to choose the adequate triangle to be glued in the mesh. The main advantage of this kind of technique is to avoid the use of three-dimensional structures when the goal is to derive a two-dimensional one. Principal component analysis is used to estimate the normal vector in the samples points, and rigid movements are used to optimize the projections operations.

By avoiding those three-dimensional structures, the LDT algorithm improves not only the running times, but also the amount of memory used in the reconstruction process, which enable it to reconstruct models with considerable quantity of points, as showed in the lucy model (Figure 10).

Unfortunately, was not possible to compare the LDT algorithm with the one developed by Gopi *et al.*[20], which is another two-dimensional approach that uses Delaunay triangulation. Basically, the Gopi' approach uses a sample criteria to select the candidate points in computation of the Delaunay triangulation. The comparison of the two techniques must be subject of a future work.

Another step to be analyzed is the possibility of substitute the two-dimensional Delaunay triangulation by another kind of calculation, which cam makes this algorithm even faster. Another possibility for future work is to produce theoretical guarantees of the reconstruction, that is, to investigate for which value of r the LDT produces a correct reconstruction of a $r-$sampled surface.

ACKNOWLEDGEMENTS

The authors are grateful to Tamal K. Dey for providing his implementations of the Cocone and Tight Cocone; and to João Paulo Gois for his implementation of Crust and Power Crust.

REFERENCES

[1] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. Silva. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):3–15, 2003.

[2] N. Amenta and M. W. Bern. Surface reconstruction by voronoi filtering. In *Symposium on Computational Geometry*, pages 39–48, 1998.

[3] Nina Amenta, Sunghee Choi, Tamal K. Dey, and N. Leekha. A simple algorithm for homeomorphic surface reconstruction. *International Journal of Computational Geometry and Applications*, 12(1-2):125–141, 2002.

[4] Nina Amenta, Sunghee Choi, Tamal K. Dey, and N. Leekha. A simple algorithm for homeomorphic surface reconstruction. *International Journal of Computational Geometry and Applications*, 12(1-2):125–141, 2002.

[5] Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust, unions of balls, and the medial axis transform. *Computational Geometry*, 19(2-3):127–153, 2001.

[6] Nicolas Aspert, Diego Santa-Cruz, and Touradj Ebrahimi. Mesh: Measuring errors between surfaces using the hausdorff distance. In *Proc. of the IEEE International Conference in Multimedia and Expo (ICME) 2002*, volume 1, pages 705–708, Lausanne, Switzerland, August 2002.

[7] D Attali. *r*-regular shape reconstruction from unorganized points. *Computational Geometry Theory and Applications*, 10:239–249, 1998. Elsevier.

[8] Chandrajit L. Bajaj, Fausto Bernardini, and Guoliang Xu. Automatic reconstruction of surfaces and scalar fields from 3d scans. *Computer Graphics*, 29(Annual Conference Series):109–118, 1995.

[9] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, 1999.

[10] Helton Hideraldo Bíscaro, Antonio Castelo Filho, Luis Gustavo Nonato, and Maria Cristina Ferreira de Oliveira. A topological approach for surface reconstruction from sample points. *Vis. Comput.*, 23(9):793–801, 2007.

[11] J D Boissonnat. Shape reconstruction from planar cross-sections. *Computational Vision Image*, 44:1–29, 1988.

[12] Jean-Daniel Boissonnat, Olivier Devillers, Sylvain Pion, Monique Teillaud, and Mariette Yvinec. Triangulations in cgal. *Computational Geometry - Theory and Applications*, 22:5–19, 2002.

[13] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3d objects with radial basis functions. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 67–76. ACM Press, 2001.

[14] B Curless and Levoy M. A volumetric method for building complex models from range images. In *Siggraph*, pages 303–312, 1996.

[15] Tamal K. Dey and Samrat Goswami. Tight cocone : A water-tight surface reconstruction. Technical Report OSU-CISRC-12/02-TR31, The Ohio State University, december 2002.

[16] Hebbert Edelsbrunner. Surface recosntruction by wrapping finite point set in space. *Discrete and Computational Geometry. The Goodman-Pollack Festschrift*, pages 379–404, 2003.

[17] Herbert Edelsbrunner and Ernst P. Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics*, 13(1):43 – 72, 1994.

[18] Fortune. Voronoi diagrams and delaunay triangulations. In *Computing in Euclidean Geometry, Edited by Ding-Zhu Du and Frank Hwang, World Scientific, Lecture Notes Series on Computing – Vol. 1*. 1992.

[19] João Paulo Gois. Reconstrução de superfícies a partir de nuvens de pontos. -in portuguese. Master's thesis, Universidade de São Paulo - Instituto de Ciências Matemáticas e Computação., 2004.

[20] M. Gopi, S. Krishnan, and C. T. Silva. Surface reconstruction based on lower dimensional localized delaunay triangulation. In M. Gross and F. R. A. Hopgood, editors, *Computer Graphics Forum (Eurographics 2000)*, volume 19(3), 2000.

[21] Leonidas Guibas and Jorge Stolfi. Primitives for the manipulation of general subdivisions and the computation of voronoi. *ACM Trans. Graph.*, 4(2):74–123, 1985.

[22] W.R. Hamilton. *Elements of Quaternions*. Chelsea Publishing Company, third edition, 1969 - The original was published in 1866.

[23] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. *Computer Graphics*, 26(2):71–78, 1992.

[24] Ian T. Jolliffe. *Principal components analysis*. Springer Us, 2002.

[25] Ravikrishna Kolluri, Jonathan Richard Shewchuk, and James F. O'Brien. Spectral surface reconstruction from noisy point clouds. In *Sigraph*, pages 11–22, 2004.

[26] D. Levin. Mesh-independent surface interpolation. *Geometric Modeling for Scientific Visualization*.

[27] King-Ip Lin and Congjun Yang. The ann-tree: An index for efficient approximate nearest neighbor search. In *DASFAA '01: Proceedings of the 7th International Conference on Database Systems for Advanced Applications*, pages 174–181, Washington, DC, USA, 2001. IEEE Computer Society.

[28] Boris Mederos, Luiz Velho, and Luiz Henrique de Figueiredo. Moving least squares multiresolution surface approximation. In *SIBGRAPI 2003*, 2003.

[29] Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. Multi-level partition of unity implicits. *ACM Trans. Graph.*, 22(3):463–470, 2003.

[30] John Schreiner, Carlos E. Scheidegger, Shachar Fleishman, and Cláudio T. Silva. Direct (re)meshing for efficient surface processing. *Computer Graphics Forum*, 25(3):527–536, 2006.

[31] Andrei Sharf, Thomas Lewiner, Ariel Shamir, Leif Kobbelt, and Daniel Cohen-Or. Competing fronts for coarse–to–fine surface reconstruction. In *Eurographics*, pages 389–398, Vienna, september 2006.

[32] Jonathan Richard Shewchuk. Adaptive precision floating-point arithmetic and fast robust geometric predicates. *Discrete & Computational Geometry*, 18:305–363, 1996.

[33] Marek Teichmann and Michael Capps. Surface reconstruction with anisotropic density-scaled alpha shapes. In David Ebert, Hans Hagen, and Holly Rushmeier, editors, *IEEE Visualization '98*, pages 67–72, 1998.