Periodic Meshing of Surfaces in CGAL [*]

Vissarion Fisikopoulos

advisor: Monique Teillaud

August 2009

INRIA

[*]These notes were prepared for my internship in Geometrica team at INRIA under the supervision of Monique Teillaud, July-August 2009, France

Contents

List of Figures

1 Introduction

This report is divided in three parts. In the first one there is a brief introduction about Surface Meshing and 3D Periodic Delaunay Triangulations. In the second part there are some notes about design and implementation issues of the Periodic Surface Mesher for CGAL [cga09]. Finally, in examples section there are some visualizations of minimal triply periodic surfaces using Periodic Surface Mesher in CGAL.

2 Preliminaries

2.1 The CGAL Surface Mesher

Assume a Delaunay triangulation dt and surface S in 3 dimensional space.

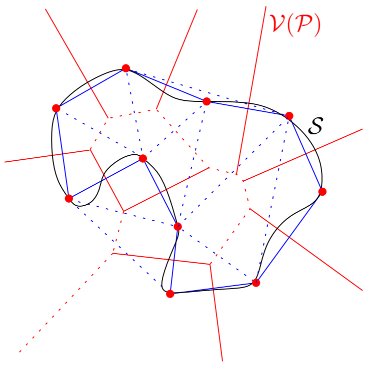
Definition 2.1. Restricted Delaunay triangulation rdt w.r.t. dt and S is a 2 dimensional complex $c2t3$ of facets of dt whose dual Voronoi segments intersect S. These facets called boundary facets.

Definition 2.2. Surface Delaunay ball of a boundary facet f is the ball with center an intersection of f with S circumscribing f.

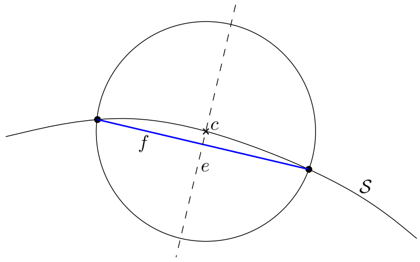
Refinement Criteria.

- angular bound : lower bound to the minimum angle of any boundary facet
- radius bound : upper bound to the radius of the surface Delaunay ball of any facet
- distance bound : upper bound to the distance between the center of a boundary facet and the center of its surface Delaunay ball

Definition 2.3. Bad facet is a boundary facet the does not meet the refinement criteria.



(a′) Restricted Delaunay triangulation



(б′) Surface Delaunay ball

Figure 1: (a) 2d case: The blue edges are the boundary facets and consist the Restricted Delaunay triangulation. The red edges are the dual Voroinoi edges that intersect the surface S. (б) f is the boundary facet of surface S, e the dual Voroinoi edge and c the center of the Surface Delaunay ball.

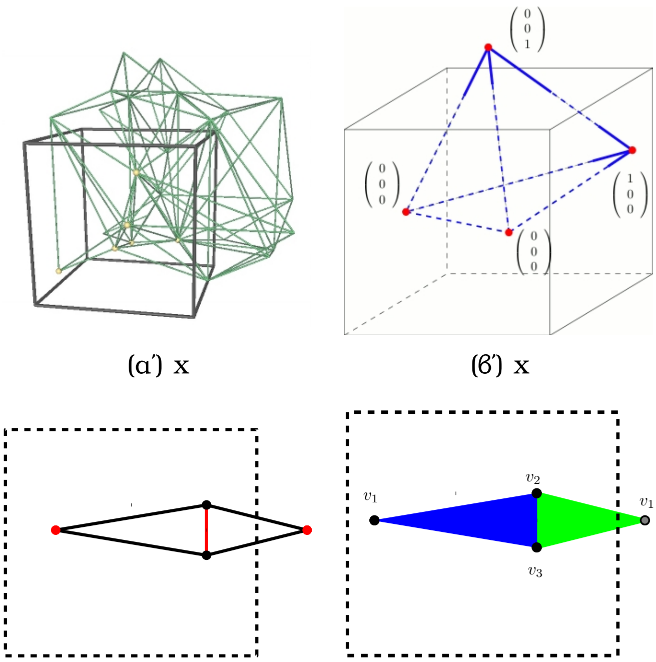
The Algorithm. A initialization step inserts at least 3 points (20 points in the CGAL implementation) laying on the surface to the Delaunay triangulation. These points are constructed by ray shouting from the origin point to a random point on the boundary sphere and compute the intersection with the surface via a dichotomic search.

A refinement procedure refines the constructed restricted Delaunay triangulation w.r.t. the surface until it meets some criteria. These criteria inserts the notion of bad facets of rdt. The algorthm maintains a priority queue storing bad facets and a c2t3 data structure storing the restricted to surface Delaunay triangulation. In each step refines one bad facet by inserting in the triangulation the center of its Delaunay surface ball which is the intersection of the facet's dual Voronoi segment with the surface. It also updates both the triangulation and the bad facets priority queue.

In [BO05] there is a detailed illustration of the theoretical aspects of surface meshing and sampling in which the surface mesher algorithm of CGAL is based.

2.2 The CGAL 3D Periodic Delaunay Triangulation

A 3D Periodic Delaunay Triangulation is a Delaunay Triangulation computed in the periodic space $\mathbb{T}_c^3 := \mathbb{R}^3/\mathbb{Z}^3$. We define as domain



(α′) x (β′) x

(γ′) x (δ′) x

Figure 2: (α) the original domain and the way it stores the triangulation (β) a cell crossing the domain and the offsets of the points (γ) the 2 simplices' intersection is not a simplex (δ) 2 facets with the same vertices but with different offsets

the cube that contains exactly one representative of each periodic element. We compute our triangulations in original domain which is the one that contains the origin point. Each point of the triangulation is represented inside the main domain. For each point we define an offset (o_x, o_y, o_z). It determines the number of periods that the point is far from the original domain. Edges and cells store their points together with an offset for each point. An offset of a point v contains the information about the domain that this point is with respect to the other points of the edge or cell.

When computing in periodic space there might be point sets that have not triangulation in $\mathbb{T}_c^3$ (see figure 2(γ′)).

A solution ([CT09]) to this is either compute in 27-sheeted covering space which means that we dublicate our domain 26 times or insert 36 dummy points at the beginning of the procedure. In this project we choose the second solution. The reason is that the insertion of the dummy points will not affect the result of the mesh because of the refinement steps (the facets that these points introduce will be refined). Note that by inserting this points the property of the surface meshing that all the points of the triangulation are on the surface is not hold any more.

In [CT09] there is the theory of 3D Periodic Delaunay Triangulations used to implement this software package.

3 Periodic Surface Mesher

The goal is to mesh triply periodic surfaces given their implicit functions. We will use the surface mesher package of CGAL with the 3D Periodic Delaunay Triangulations as template parameter.

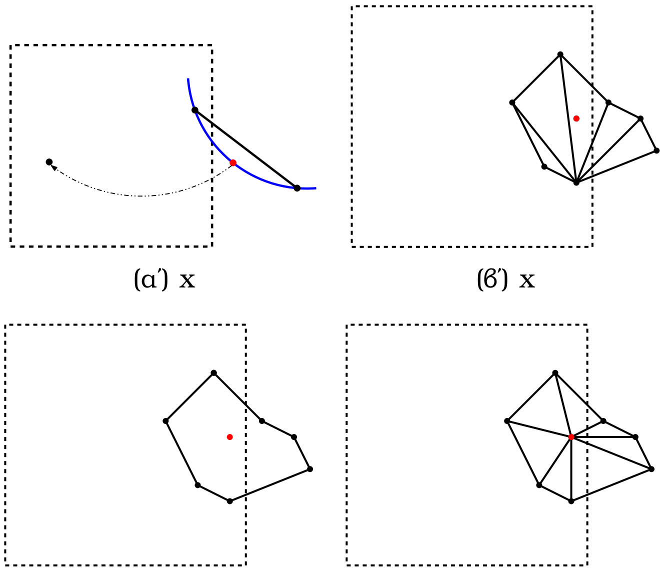
Implementation issues. An new class Periodic_3_Delaunay_ triangulation_3_Surface_mesher is contructed overloading methods from Periodic_3_Delaunay_triangulation_3 class in order to make Periodic triangualtions compatible to the surface mesher (e.g. dimension, is_infinite, dual) and also to provide new functionalities needed for periodic meshing (e.g. insert, insert_in_hole, find_ conflicts). There are also some functions of visualization, that output the triangulation to an .off or .mesh file. The meshing algorithm is implemented using the design of mesher levels described in [RY07]. (Mesher_level.h: contains algorithms, Surface_mesher.h: contains implementations of basic functions)

3.1 Modifications

3.1.1 Point insertion

A point insertion could happend either in the initialization step or at the refinement step. There can be cases that the surface and the boundary facet crosses the domain and the refinement point is outside the domain. In these cases the algorithm should translate these points inside the domain using offsets (see figure 3.1.1 (α)).

Implementation issues. For the initialization step we just overload insert method in order to translate points that are outside the domain inside. For the refinement step we overload insert_in_hole, find_ conflicts methods. For translation we use the canonicalize_point



(α′) x (β′) x

(γ′) x (δ′) x

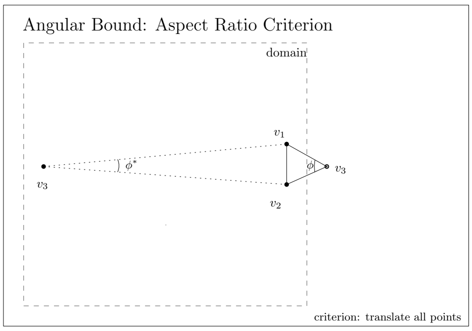
Figure 3: (α) translating a out-of-domain refinement point inside the domain (β)-(δ) keep track of offsets during insertion

function. Note that this implementation is not so efficient because we have to call canonicalize_point many times for the same points.

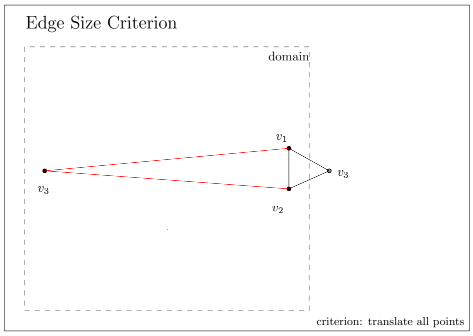
3.1.2 Star Approach

When the algorithm makes an insertion it uses the star approach, so it makes a point location then find the conflicting cells, it makes a hole removing them and finally it makes a star in the hole (see figure 3.1.1 (β)-(δ)) creating new cells. When the cells are removed the information of the offsets is also removed and when the new cells are created the algorithm has to put some offsets to them. All these are solved and implemented at the Periodic Triangulations CGAL package but the surface mesher uses methods directly from the original Triangulations classes. So we had to adjust some code of the periodic triangulations to the methods the mesher uses for point insertion.

Implementation issues. Here we have to overload insert_in_hole and find_conflicts methods. The change is just a movement of the clear offsets procedure from find_conflicts to insert_in_hole.



(α′) Aspect Ratio Criterion



(β′) Edge Length Criterion

Figure 4: The point v_3 is outside the main domain and the geometric computation with its representation inside the domain leads to false results. One can see the two different triangles which correspont to the points v_1, v_2, v_3 with the two different translations of v_3.

3.1.3 Refinement Criteria.

The problem with the mesh criteria is that they use geometric information, which in peoridic case can lead to computations of the wrong geometric quantity. At each criterion there is a facet which has to be checked whether it is bad w.r.t. some lower and upper bounds given to the algorithm (see section 2.1). So each criterion has to be adapted in order to compute the right geometric quantity.

Angular Bound - Aspect Ratio Criterion. In this case the problem is simple. If the facet we want to check is periodic, i.e. two of its vertices points lay in a different domain, the computation of the minimum angle can be false (see figure 4(α′) for an illustration). This is because the facet is a combinatorial object and in the periodic case represents more than one triangle, which is a geometric object. The solution is to translate the points to their periodic coordinates using their offsets stored in the cell. With this translation we construct the right triangle and as a result we take the right angles and edge lenghts.

Edge Length Criterion. The case is similar to the angular bound case because both angles and edges should depend on the triangles. See figure 4(β′) for an illustration. This criterion is not used by the algorithm.

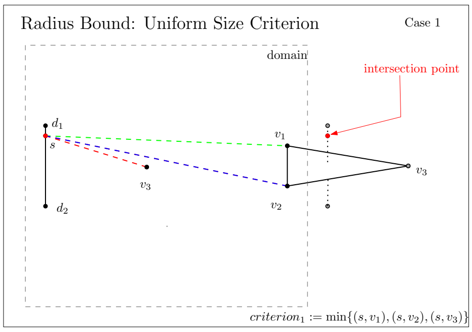
Radius Bound - Uniform Size Criterion. In this criterion the algorithm calculates the dual Voronoi edge of the checking facet and computes an intersection point s of this edge with the surface. This criterion computes the distance af any point of the facet with s in order to find the radius of the surface Delaunay ball of the facet. The dual edge can be periodic so the potential problem arising is that the intersection point s is translated differently than the point v.

One case is when the dual segment is completely outside the domain (see figure 5(α′)) and the facet is crossing it. In this case the dual segment is represented totally inside the domain so the computation of the distance of s to any vertex of the facet can be different than the real distance. But if the facet is included in only 2 periodic domains we can be sure that at least one of its vertices v_3 is translated inside the domain. So the distance of s with v_3 is real one which force us to take as criterion the min$\{(s, v_1), (s, v_2), (s, v_3)\}$.

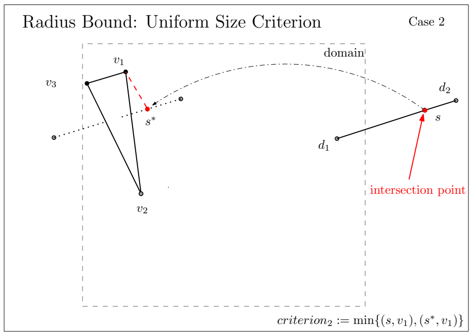
Anot case is the opposite; if the facet is completely inside the domain (see figure 5(β′)) and the dual edge is crossing it. In this case the facet is represented totally inside the domain but the dual segment is represented at the edge of the domain. If s is outside the domain we translate it to s^*. So the new criterion is min$\{(s, v_1), (s, v_2), (s, v_3), (s^*, v_1)\}$. Note that we don't handle the case when s is inside the domain, because both facet and s will be inside the domain. When all the points are inside the domain we can't distinguish between a configuration which gives the right criteria and the one that does not. So a solution to that could be to pass to the criteria more information, for example the points of the dual edge.

Implementation issues. We use two new classes Periodic_criteria with the implementation of the criteria defined above and also Surface_mesh_periodic_criteria_3 which is the same as the original Surface_mesh_criteria_3 instead of loading our new periodic criteria.

Distance Bound - Curvature Size Criterion. The distance bound criterion is similar to the radius bound criterion and we use the same ideas and solutions. The new criterion here is min$\{(s, c), (s, c^*), (s^*, c)\}$ (see figure 3.1.3).

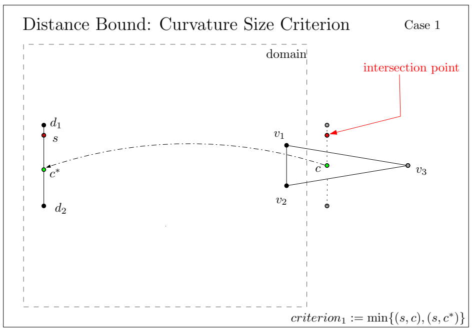


(α′) Uniform Size Criterion: Case 1

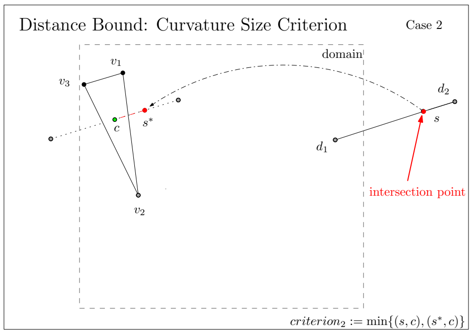


(β′) Uniform Size Criterion: Case 2

Figure 5: Radius Bound



(α′) Curvature Size Criterion: Case 1



(β′) Curvature Size Criterion: Case 2

Figure 6: Distance Bound

4 Examples

In this section there are presented some examples of the use of periodic surface mesher with some known triple periodic minimal surfaces found on [DH]. The visualization was made using the mesh visualization software: medit [Fre01]. We also present some bad examples. Finally we visualize our results by exporting the domain of the surface 8 times constructing a cube of edge 2 emphasizing on the periodicity of these surfaces and on the event that they can actually be ``glued'' together (see figures at the end of this report).

cylinder	$x^2 + y^2 = 0$
diamond	$sinx \cdot siny \cdot sinz$ $+ sinx \cdot cosy \cdot cosz$ $+ cosx \cdot siny \cdot cosz$ $+ cosx \cdot cosy \cdot sinz = 0$
double p	$0.5 \cdot (cosx \cdot cosy + cosy \cdot cosz + cosz \cdot cosx)$ $+ 0.2 \cdot (cos2x + cos2y + cos2z)$
D prime	$1 \cdot (sinx \cdot siny \cdot sinz)$ $+ 1 \cdot (cosx \cdot cosy \cdot cosz)$ $- 1 \cdot (cos2x \cdot cos2y + cos2y \cdot cos2z + cos2z \cdot cos2x))$ $- 0.4$
G prime	$5 \cdot (sin2x \cdot sinz \cdot cosy + sin2y \cdot sinx \cdot cosz + sin2z \cdot siny \cdot cosx)$ $+ 1 \cdot (cos2x \cdot cos2y + cos2y \cdot cos2z + cos2z \cdot cos2x)$
gyroid	$cosx \cdot siny + cosy \cdot sinz + cosz \cdot sinx$
lidinoid	$1 \cdot (sin2x \cdot sinz \cdot cosy + sin2y \cdot sinx \cdot cosz + sin2z \cdot siny \cdot cosx)$ $- 1 \cdot (cos2x \cdot cos2y + cos2y \cdot cos2z + cos2z \cdot cos2x)$ $+ 0.3$
schwarz p:	$cosx + cosy + cosz$
split p	$1.1 \cdot (sin2x \cdot sinz \cdot cosy + sin2y \cdot sinx \cdot cosz + sin2z \cdot siny \cdot cosx)$ $- 0.2 \cdot (cos2x \cdot cos2y + cos2y \cdot cos2z + cos2z \cdot cos2x)$ $- 0.4 \cdot (cosx + cosy + cosz)$

Table 1: List of surfaces

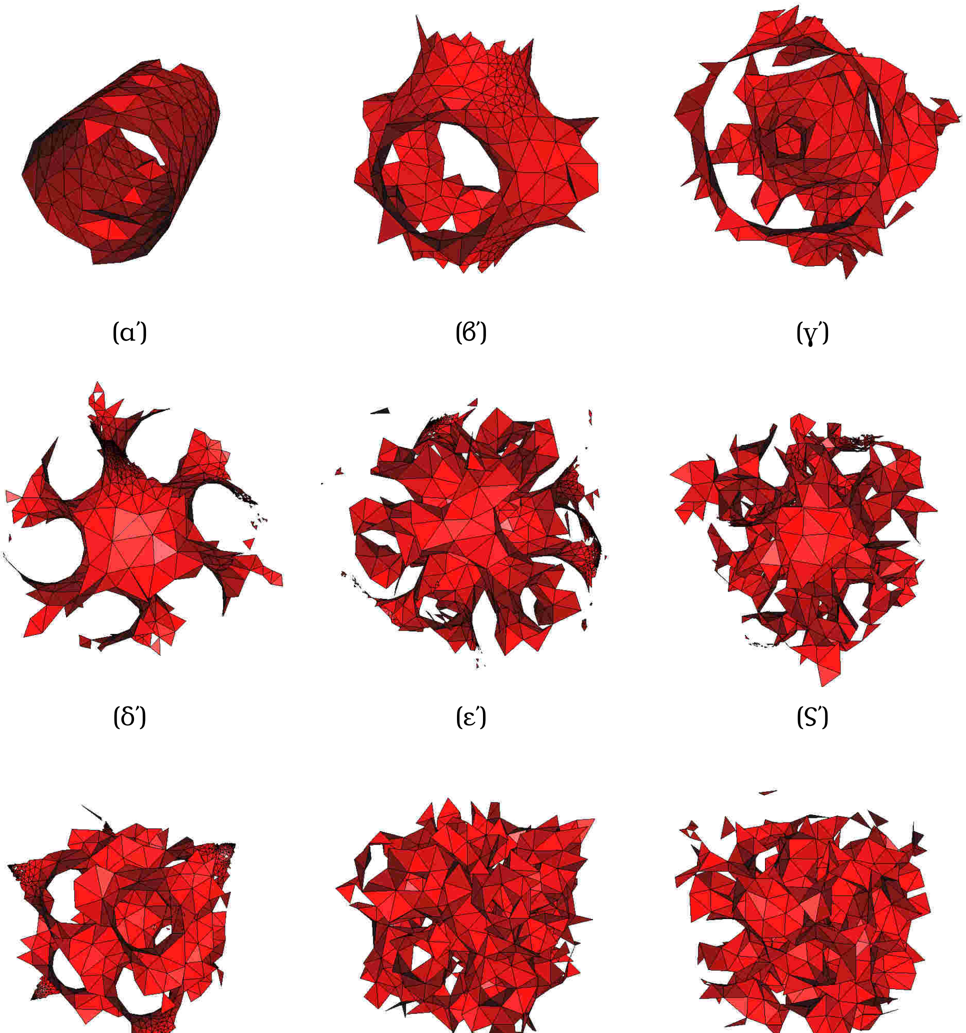


Figure 7: Periodic Minimal Surfaces: criteria: $angular = 30$ $radius = 0.1$ $distance = 0.1$

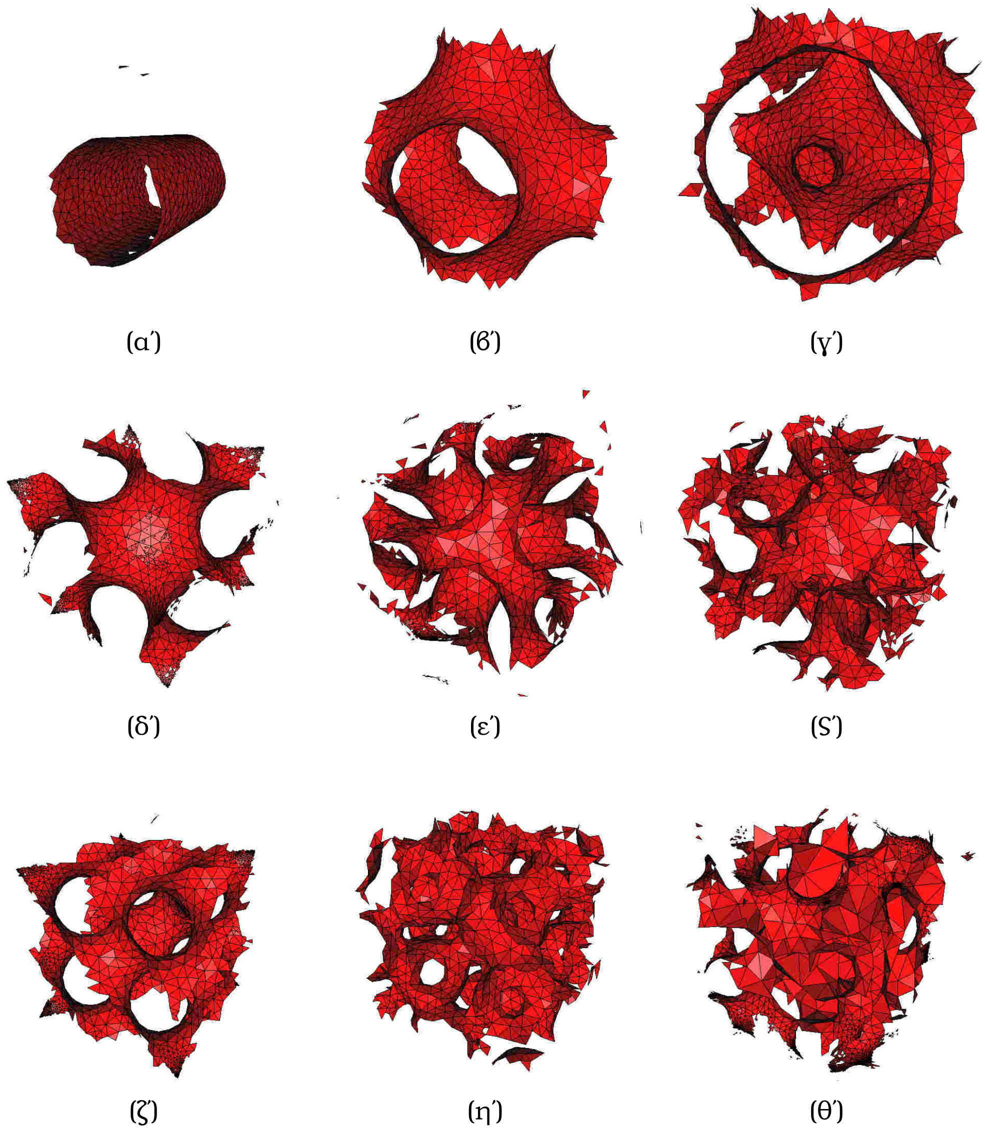
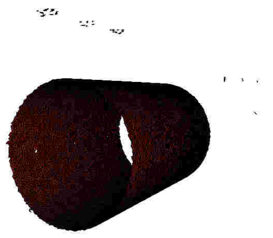
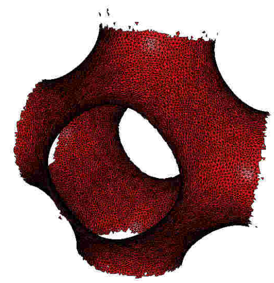


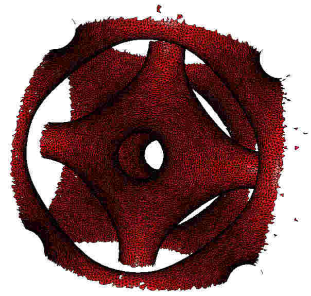
Figure 8: Periodic Minimal Surfaces: criteria: $angular = 30$ $radius = 0.05$ $distance = 0.05$



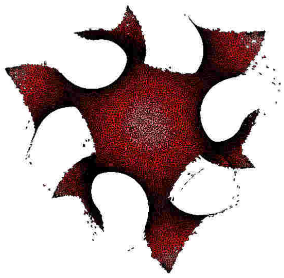
(α′)



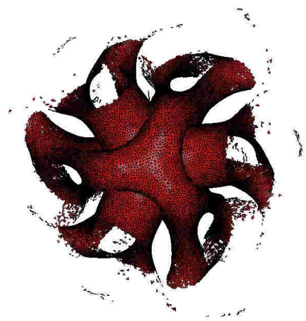
(β′)



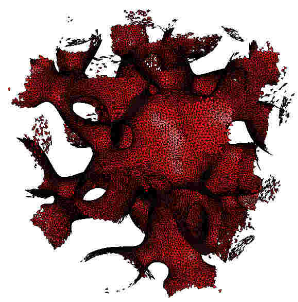
(γ′)



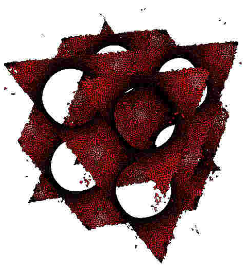
(δ′)



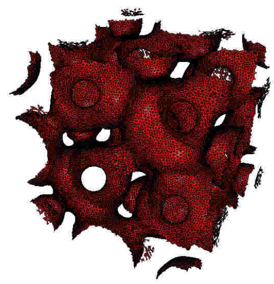
(ε′)



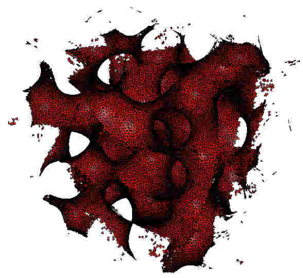
(ς′)



(ζ′)

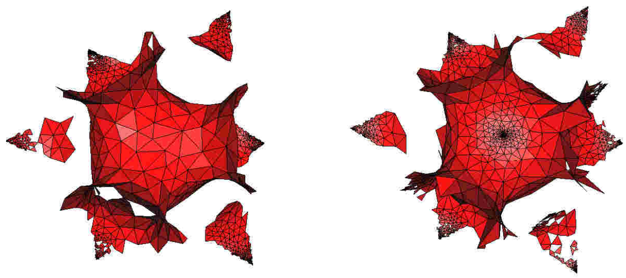


(η′)



(θ′)

Figure 9: Periodic Minimal Surfaces: criteria: $angular = 30$ $radius = 0.01$ $distance = 0.01$



(α′) (β′)

Figure 10: Bad examples: schwarz p surface shifted

5 Conclusion - Future Work

To conclude with, the algorithm terminates for all of the above surfaces so the state that in practice it can mesh many periodic minimal surfaces. This is a first result. On a second phase we have to construct new criteria and prove that the algorithm's termination and correctness. Note that it is sufficient to prove that the new criteria handle all the possible cases because this criteria uses the original criteria of the surface mesher which have a proven termination and correctness [BO05].

Acknowledgements

I would like to thank Monique Teillaud and Manuel Caroli for their guidance and help and also Stephane Tayeb and Dobrina Boltcheva for their help and fruitful discussions about cgal and meshes. Finally I would like to thank all the people in Geometrica team and all the interns for providing me a great working environment.

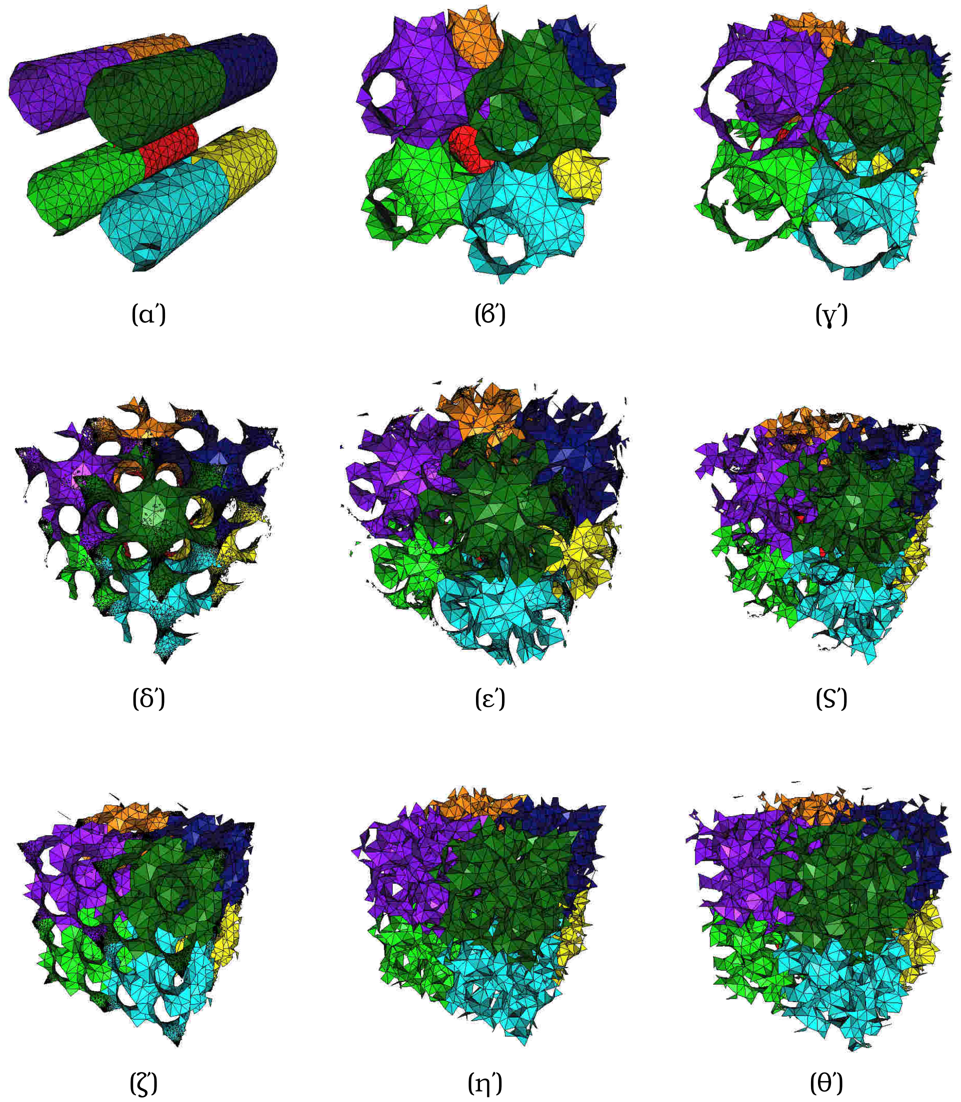
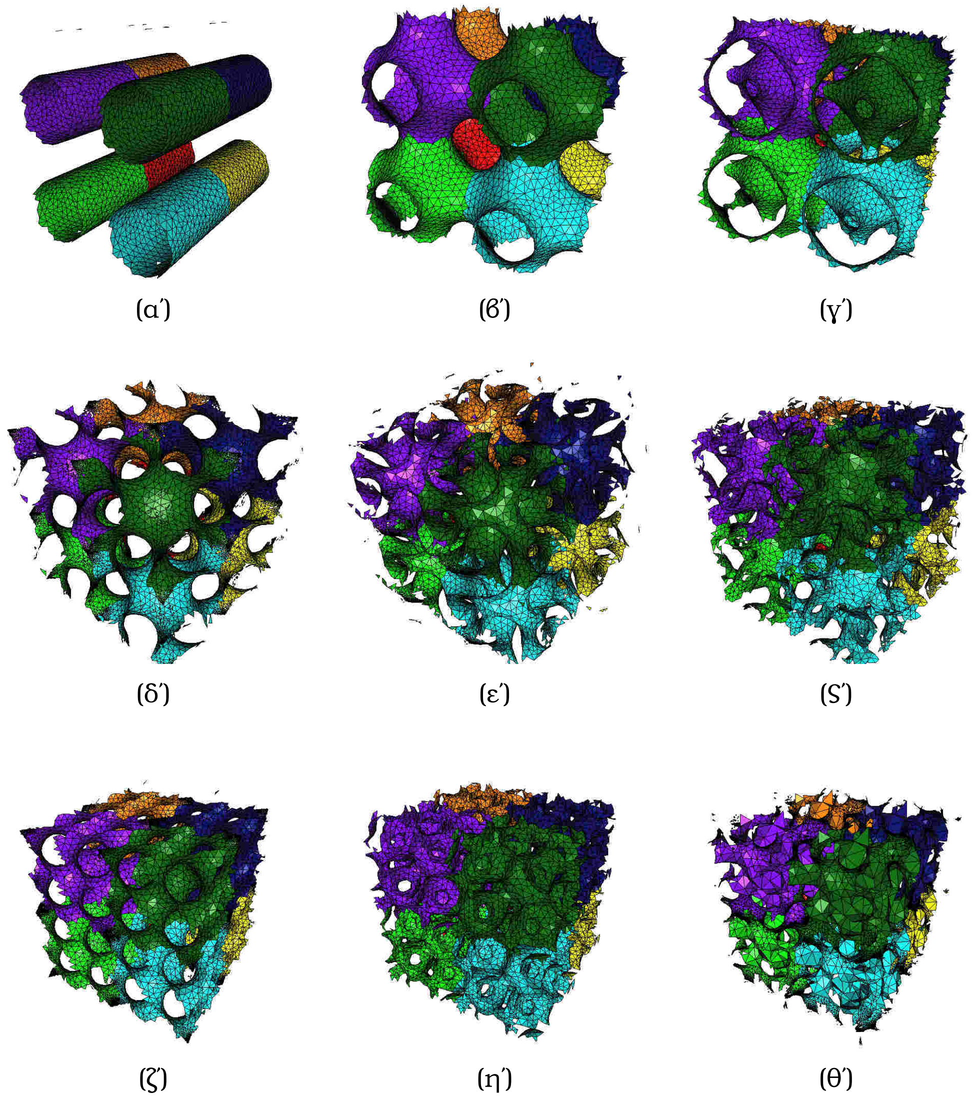


Figure 11: Periodic Minimal Surfaces (8-sheet): criteria: $angular = 30$ $radius = 0.1$ $distance = 0.1$



(α′) (β′) (γ′) (δ′) (ε′) (ϛ′) (ζ′) (η′) (θ′)

Figure 12: Periodic Minimal Surfaces (8-sheet): criteria: $angular = 30$ $radius = 0.05$ $distance = 0.05$

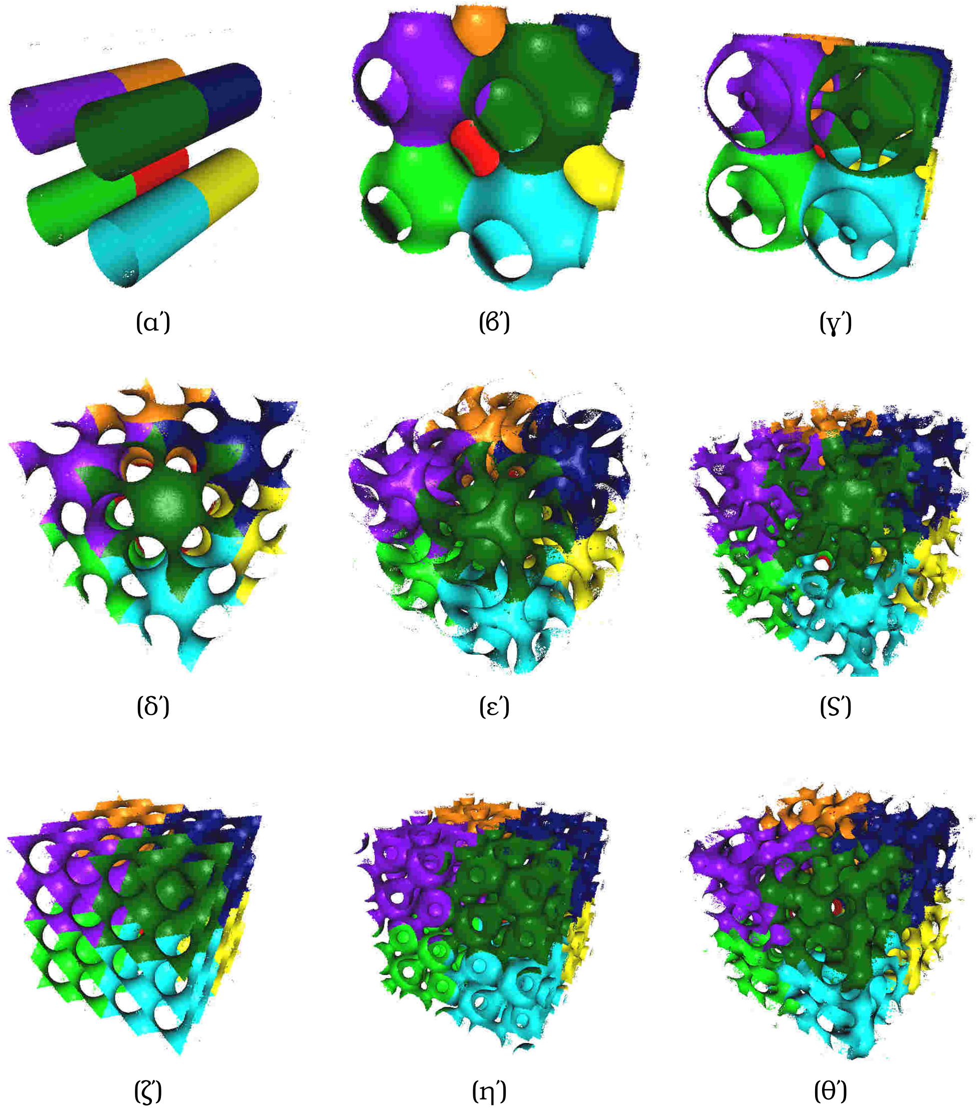


Figure 13: Periodic Minimal Surfaces (8-sheet): criteria: $angular = 30$ $radius = 0.01$ $distance = 0.01$

References

[BO05] Jean-Daniel Boissonnat and Steve Oudot. Provably good sampling and meshing of surfaces. *Graph. Models*, 67(5):405--451, 2005.

[cga09] Cgal, computational geometry algorithms library. www.cgal.org, August 2009.

[CT09] Manuel Caroli and Monique Teillaud. Computing 3D Periodic Triangulations. Research Report RR-6823, INRIA, 2009.

[DH] James T. Hoffman David Hoffman. The scientific graphics project. http://www.msri.org/about/sgp/jim/geom.

[Fre01] Pascal J. Frey. MEDIT : An interactive Mesh visualization Software. Technical Report RT-0253, INRIA, 12 2001.

[RY07] Laurent Rineau and Mariette Yvinec. A generic software design for delaunay refinement meshing. *Geom. Theory Appl.*, 38:100--110, 2007.