MODELING PLANAR 3-VALENCE MESHES

A Thesis

by

OZGUR GONEN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2007

Major Subject: Visualization Sciences

MODELING PLANAR 3-VALENCE MESHES

A Thesis

by

OZGUR GONEN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Co-Chairs of Committee,	Ergun Akleman
	Vinod Srinivasan
Committee Members,	Richard Burt
Head of Department,	Mark Clayton

December 2007

Major Subject: Visualization Sciences

ABSTRACT

Modeling Planar 3-Valence Meshes. (December 2007)

Ozgur Gonen, B.A., Istanbul Bilgi University

Co–Chairs of Advisory Committee: Dr. Ergun Akleman
Dr. Vinod Srinivasan

In architectural and sculptural practice, the eventual goal is constructing the shapes that have been designed. Due to fabrication considerations, shapes with planar faces are in demand for these practices.

In this thesis, a novel computational modeling approach to design constructible shapes is introduced. This method guarantees that the resulting shapes are planar meshes with 3-valence vertices, which can always be physically constructed using planar or developable materials such as glass, sheet metal or plywood. The method introduced is inspired by the traditional sculpture and is based on the idea of carving a mesh by using slicing planes. The process of determining the slicing planes can either be interactive or automated.

A framework is developed which allows user to sculpt shapes by using the interactive and automated processes. The framework allows user to cut a source mesh based on its edges, faces or vertices. The user can sculpt various kinds of developable surfaces by cutting the parallel edges of the mesh. The user can also introduce interesting conical patterns by cutting different vertex, edge, face combinations of the mesh.

ACKNOWLEDGMENTS

First of all, I would like to thank my parents for their endless support through-out the completion of my masters degree. Their valuable support has always motivated me along the way.

I would like to thank Dr. Ergun Akleman for being my committee co-chair and for his innovative contributions to my research. I would also like to thank my other committee co-chair, Dr. Vinod Srinivasan, and committee member, Dr. Richard Burt, for their critiques and support.

I would like to thank the VIZA 657 3D Computer Aided Sculpting class fall'07 students for their effort in designing shapes using my implementation and especially David Morris for his constructive feedback.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION AND MOTIVATION

In architecture and sculptural modeling, we eventually want to build the shapes that we have designed. This leads us to consider fabrication issues that might be challenging due to the physical limitation of fabrication material and the possible restrictions of the fabrication process.

With the current improvements in rapid prototyping, most of the manifold polygonal models can be printed in 3D. However this process is limited to a single fabrication material. It is also impossible to print models at larger scales.

The most common fabrication materials used in architectural free form design such as glass, sheet metal or plywood are suitable for constructions at large scale. Although these materials can be bent into curved panels, it is highly cost effective to use them as planes. This essentially limits the models to be meshes with planar faces.

A few approaches have been presented to address this limitation address this limitation. One of them is the concept of conical meshes, which are quadrilateral meshes with planar faces [1]. The use of planar faces makes it suitable for the design of freeform glass structures. Cutler [2] developed an algorithm that remeshed a NURBS or subdivision surface into planar panels by an iterative clustering method.

The methods above suggest a planar approximation to a given source mesh by the designer. However, in this approach, the designer's control on the final product is mostly dominated by the remeshing algorithm. In this thesis, as opposed to developing an approximation algorithm for creating planar meshes, a computational method

---

The journal model is *IEEE Transactions on Automatic Control.*

for designing shapes with planar faces is directly introduced. The designer is more involved in the design process in this method.

The method in this thesis is inspired by traditional sculpture and is based on the idea of removing chunks of mesh out of an initial mesh. The mesh is sculpted by cutting slices off the mesh by a using a slicing plane. This method ensures that all the new faces introduced are planar faces with valence-3 vertices.

The method in this thesis is implemented in TopMod software [3, 4]. The implementation presents an intuitive framework for the designer to sculpt planar meshes with the provided artist tools. Using this framework it is very easy to come up with interesting planar meshes.

An innovative feature of the method is the variety of patterns that can be introduced in the design, which could not be obtained by a regular remeshing algorithm. This gives the designer freedom to break the regular triangular or quadrilateral patterns that are common in glass structures. Using this method, it is also possible to model various D-forms, which are a recently invented type of developable sculptures.

CHAPTER II

BACKGROUND

In this chapter, various types of planar meshes relevant to this research is mentioned. Some remeshing methods to create planar meshes are also briefly explained.

A. Triangulation

The simplest method of converting an arbitrary mesh into a planar mesh is triangulation. Any arbitrary surface can easily be represented by triangles. This method is relatively easy and included in most of the modeling packages. The modern roof of the British Museum in London is an example for a regular triangulation applied to a curved surface (see Figure 1a). The surface of the roof is tiled with triangles with vertices of valence 6.

(a) (b)

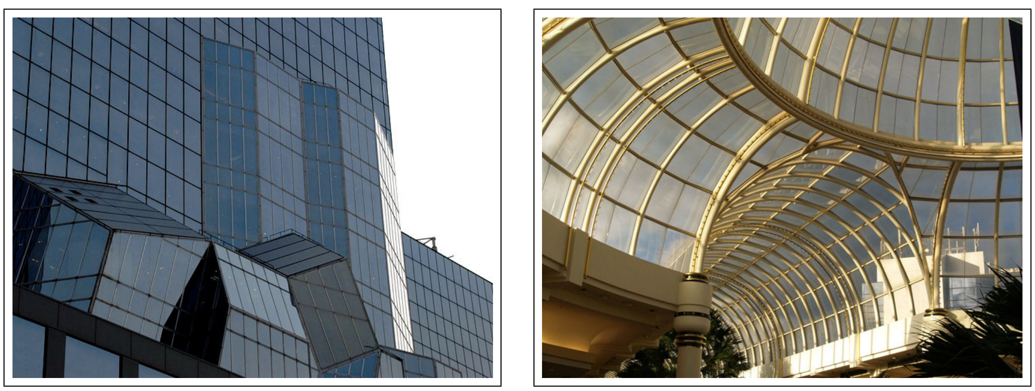
Fig. 1. (a) Example for a regular triangulation: British Museum, London [5]. (b) Example for an irregular triangulation: Fiera Milano, Italy [6]

Irregular triangulation can be used to create more complex surfaces. Fierra Milano by Jorg Schlaich demonstrates a striking usage of this approach as shown in

Figure 1b where the building structure is defined by the flexibility of an irregular pattern. The continuous ribs that carry the forces from the roof down to foundation are formed by the carefully arranged edges of triangles.

B. PQ Meshes

The study of quad meshes with planar faces, called PQ meshes were first systematically addressed by R. Sauer [7], as summarized in his monograph on difference geometry. PQ meshes are observed as a discrete counterpart of conjugate curve networks on surfaces.



(a) (b)

Fig. 2. Examples of glass constructions with planar quads: (a) Kyoto Terminal Station, Japan [8]. (b) Chadstone Mall, Australia [9]

PQ meshes appear in the mathematics literature under the name of quadrilateral meshes, which actually means that they are quad meshes with the additional property that all the quads are planar [10]. They are also suitable for freeform architectural construction (see Figure 2). Gehry Partners [11] and Schlaich Bergermann and Partners [12] argue that freeform glass structures with planar quadrilateral facets are preferable over structures built from triangular facets or non-planar quads and their work also shows a few simple ways to construct quad meshes with planar faces.

There have been other contributions to quadrilateral remeshing which do not try to achieve planarity. Alliez [13] presented the computation of quad meshes based on smoothed principal curvature lines. The faces of these meshes are not exactly planar, however it is expected that the are at least approximately planar.

C. Developable Surfaces

Mathematically, a developable surface is a surface with zero Gaussian curvature [14]. It is a surface that can, by definition, be flattened onto a plane without distortion (i.e. stretching, compressing, tearing). Oppositely, it can be made by transforming a plane (i.e. folding, bending, rolling, cutting, and gluing).

All developable surfaces embedded in 3D space are ruled surfaces [14]. A ruled surface is a surface that can be swept out by moving a line in space [15]. A developable surface can be represented as an arrangement of n planar quads in a single row. The surfaces that can be realized in 3D space are as follows:

- *Cylindrical Surfaces* When the ruling is done in parallel, the geometry is called a cylinder. More generally, it is a generalized cylinder where the cross-section is not necessarily a circle and can be any smooth curve.

- *Conical Surfaces* When the rulings pass through a single point in space, the resulting geometry is a cone. The cross section can be any smooth curve.

- *Tangent Surfaces* When the rulings are tangent to a curve in space, the surface type is a tangent surface.

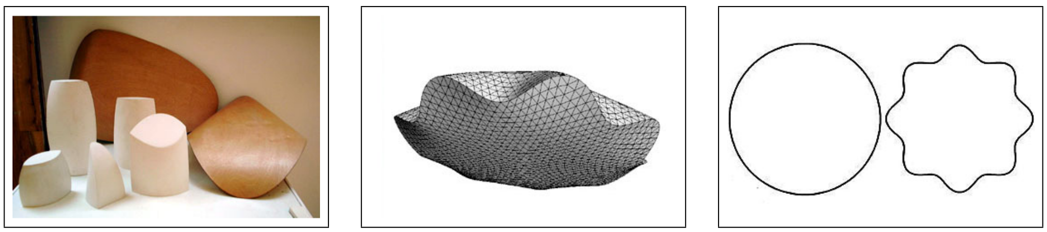
Developable surfaces are useful since they can be made out of sheet metal or paper by rolling a flat sheet of material without stretching it [16]. Most large-scale objects such as airplanes and ships are constructed using un-stretched sheet metal. In ship or airplane design, the problems usually stem from engineering concerns and in engineering design there has been a strong interest in developable surfaces. For instance, modeling packages such as Rhino provide developable surface analysis [16, 17].

Although it is easy to physically construct developable surfaces using sheet metal or paper, it is not that easy to provide computational models to represent developable surfaces. Sun and Fiume developed a technique for constructing developable surfaces [18], but their method is useful only to represent ribbons and is hard to use to represent general developable surfaces. Chu and Sequin introduced developable Bézier patches [19]. Haeberli recently introduced a method to represent a shape with piecewise developable surfaces and implemented it in his Lamina Design Software [20]. The current results seem to be limited but Haeberli's approach has great potential for developable surface design. Mitani and Suzuki introduced a method to approximate any given shape using developable surfaces to create paper models [21]. Because of the approximate nature of their models, there exist gaps between individual pieces and therefore, their method is not suitable for engineering application.

Developable surfaces are frequently used by contemporary architects to design new forms. However, the design and construction of large-scale shapes with developable surfaces requires extensive architectural and civil engineering expertise. Only a few architectural firms such as Gehry Associates have been able to take advantage of the current graphics and modeling technology to construct such revolutionary new forms [11].

1. D-Forms

D-forms are recently introduced sculptures invented by the London based designer Tony Wills [22] and first introduced by John Sharp to the art and math community [23]. D-forms are created by joining the edges of a pair of sheet metal or paper with equal perimeters [23, 22] (see Figure 3).

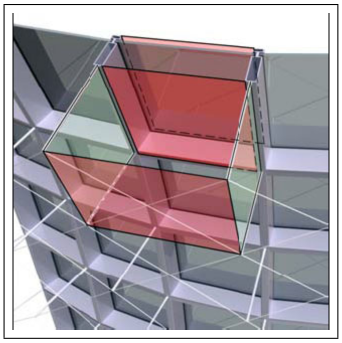


(a) (b) (c)

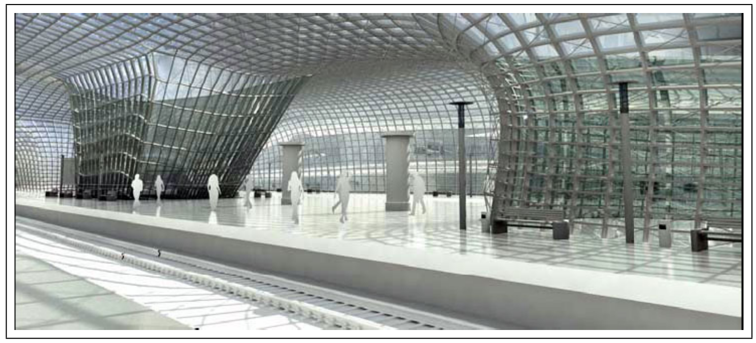
Fig. 3. (a) D-forms examples created by Tony Wills [22]. (b) A wireframe of a D-form. (c) A circle and a rippled shape with equal perimeters to create the D-form in (b).

D-forms can be approximated using thin planar quadrilaterals with valence-3 vertices. The research community has also been exploring D-forms. For instance, Pottman and Wallner introduced two open questions involving D-forms [24, 25]. Sharp introduced anti-D-forms that are created by joining the holes [26]. Ron Evans invented another related developable form called Plexagons [27]. Paul Bourke has recently constructed computer generated D-forms and plexagons [28, 27].

The method introduced in this thesis provides an alternative computational approach to physical D-form construction where the D-forms can be modeled directly and the construction is not limited to two pieces.



(a)



(b)

Fig. 4. (a) Offsetting a conical mesh results in the same connectivity. (b) A possible glass structure which is a conical mesh

D. Conical Meshes

A planar mesh is said to have conical property if all vertices in have the property that offsetting all the face planes incident with the vertex by a constant distance leads to planes which intersect again in a common point [1]. This is equivalent to the property that the planes, consistently oriented via the connectivity of the mesh, are tangent to an oriented cone of revolution.

The most common planar meshes in computer graphics, triangular and quadrilateral meshes, do not guarantee conical property. Wallner [1] introduced a method for approximating smooth surfaces with valence 4 planar quadrilateral meshes that satisfy conical property (see Figure 4).

The method in this thesis introduces planar faces with valence three vertices. Since these vertices satisfy the conical property [29], the shapes designed using this method can be easily constructed at larger scales.

CHAPTER III

SCULPTING WITH PLANES

Planar mesh modeling has been an interesting research area in geometric modeling due to the increasing demand in architectural freeform design. Much of the research in this area focuses on developing algorithms for planar approximation of a given curved surface as in [1] and [2]. In this thesis, I introduce a different approach by allowing the designer to model the planar mesh directly. This method, unlike the previous methods, gives direct control to the designer on the final product.

In this method, the designer is expected to work like a sculptor. The term "sculptor" here is a critical word which guides us to observe the methodology of traditional sculpting. A traditional sculptor creates his work of art by carving a source mass such as stone or marble. This approach is expressed in Michelangelo's famous quote as "I saw the angel in the marble and carved until I set him free"[30]. The same idea can be applied when modeling developable surfaces if the required tools are provided to the designer in the modeling environment. This crucial point is the foundation for the methodology of this research. Based on this,an algorithm that cuts a source mesh by a plane has been developed.

A. Cutting Algorithm

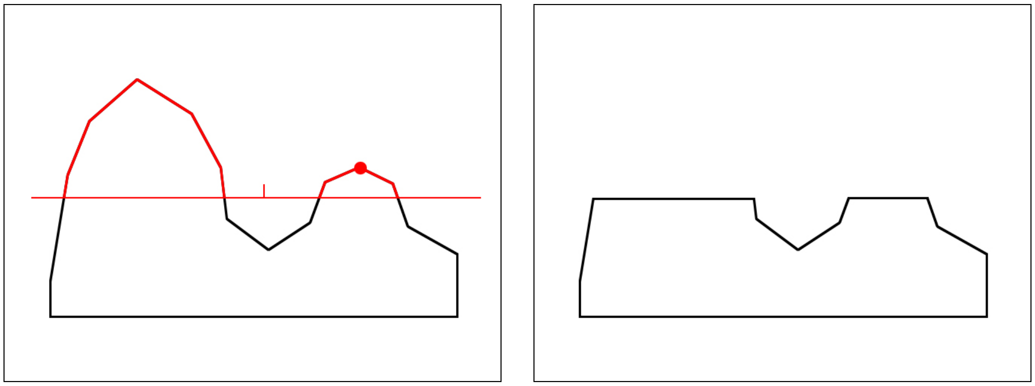
The cutting algorithm is the fundamental algorithm of this work. It is based on the idea of chopping a mesh by a given plane, called "slicing plane" in this context. The slicing plane is defined by a location and a normal vector in world space. The algorithm traverses all the edges of the mesh and tests if the edge is intersected with the slicing plane. The portion of the mesh that remains in the normal side of the plane is deleted.

A slicing plane, determined by a point P_0 and a normal n_0, divides the 3D space into negative and positive regions by the function $f(P_i) = n_0 \cdot (P_i - P_0)$ where P_i is any given point in the 3D space. To test the intersection status of an edge, the value of the function $f(P_i)$ needs to be computed for the position P_i of every vertex v_i. The value of $f(P_i)$ is greater than zero if P_i is on the normal side and less than zero if P_i is in the reverse side of the plane. The decision for intersection status of an edge is done by checking $f(P_i)$ and $f(P_j)$ where P_i and P_j are the positions of the two end vertices of the edge. There are three possible cases for an edge in this situation:

- $f(P_i) > 0$ *and* $f(P_j) > 0$; both vertices are in the normal side of the plane and the edge needs to be deleted.
- $f(P_i) \leq 0$ *and* $f(P_j) \leq 0$; both vertices are in the reverse side of the plane. In this case, the edge is skipped.
- *(* $f(P_i) \leq 0$ *and* $f(P_j) > 0$ *) or (* $f(P_i) > 0$ *and* $f(P_j) \leq 0$ *)*; the vertices lie on the opposite sides of the plane and consecutively the edge is intersected by the plane. In this situation the edge is subdivided into two sub-edges. The new vertex from subdivision is moved along the unit vector $u_{ij} = (P_j - P_i)/ \parallel (P_j - P_i) \parallel$ by amount t to the point $P_i + u_{ij}t$ where t is computed as $n_0 \cdot (P_0 - P_i)/(n_0 \cdot u_{ij})$. This operation moves the new vertex to the slicing plane. The sub-edge which remains in the normal side of the plane is marked for deletion.

When the traversing is completed, the new vertices are connected by edge insertions and the edges that are marked as deleted are deleted. This cutting algorithm cuts the mesh globally since it traverses all the edges as illustrated in Figure 5.

To localize the cutting operation, I have developed an algorithm which starts the traversal from an initial vertex node and traverses the mesh recursively by visiting



(a) (b)

Fig. 5. (a) The default cutting algorithm traverses all the edges and marks all edges above the slicing plane for deletion.
(b) The result is a global cut operation.

each of the neighboring vertices of the current vertex in each iteration. The result becomes a localized cut as seen in Figure 6. Whenever the edge between a neighboring vertex and current vertex intersects with the slicing plane, the edge is subdivided and the traversing is stopped for that node.

B. Determining the Slicing Plane

There exist only two parameters to determine a slicing plane in the 3D space; the point P_0 and the normal n_0 . In this thesis, these two parameters are computed based on the elements of the polygonal mesh such as edges, faces or vertices as described bellow.

1. Edge Based Cutting

When the slicing plane is determined relative to an edge of the mesh, the cutting algorithm operates like a planar edge truncation. The normal of the slicing plane is a

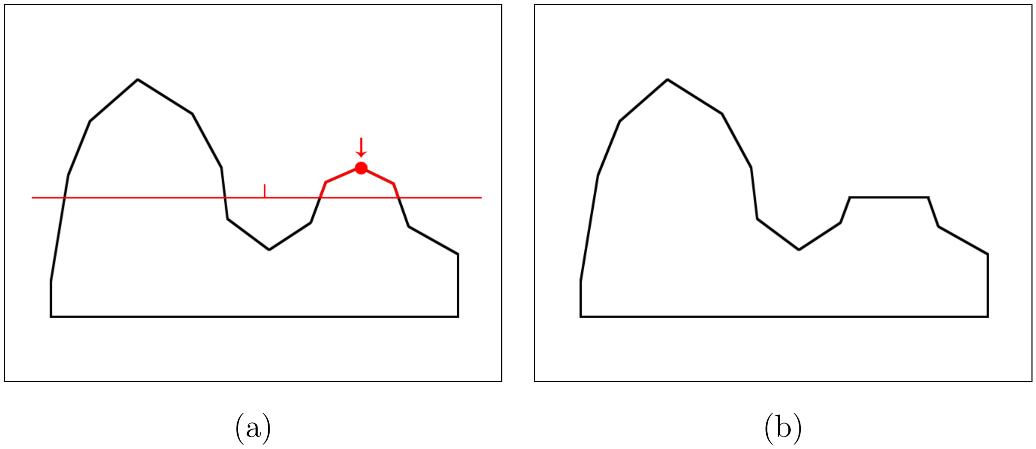


Fig. 6. (a) Local cutting starts traversing the mesh from a given element and continues until the slicing plane is hit.
(b) The result is a local operation that does not touch the rest of the mesh

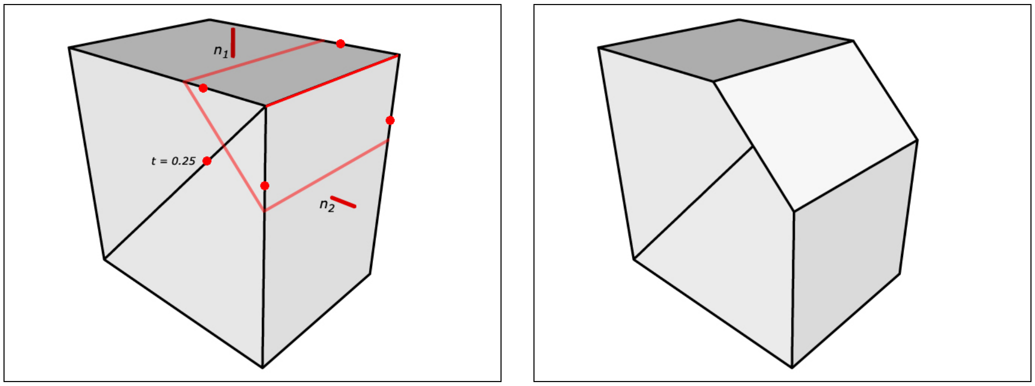
weighted average of the normals of the adjacent faces to the selected edge e_s as shown in Figure 7. The position p_0 of the slicing plane is computed as

$$\sum_{i=0}^{n}(v_i t + v_i'(1-t))/n \tag{3.1}$$

where v_i and v_i' are the two vertices of each neighboring edge e_i to the edge e_s. In the equation, the vertex v_i is the shared vertex between the edges e and e_i. The variable t is an offset value between 0 and 1.

2. Vertex Based Cutting

Determining the slicing plane relative to a vertex is very similar to determining it relative to an edge. The approach is similar to the planar vertex truncation. The location point p_0 is computed the same way as the average of all edge points as illustrated in Figure 8. The vertex normal, which is an arithmetical mean of neighboring face normals, is used as the normal of the slicing plane.



(a) (b)

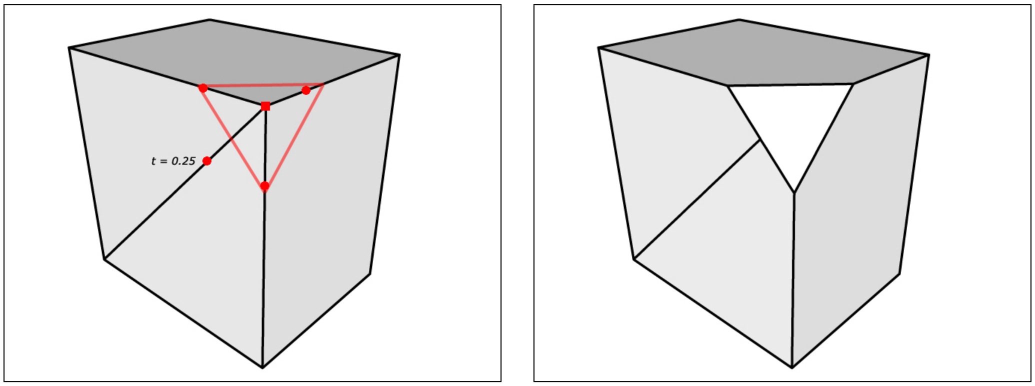
Fig. 7. Planar edge truncation:(a) The position and the normal of the slicing plane is computed based on the selected edge. (b) The selected edge is truncated by the computed slicing plan.

When planar vertex truncation is applied to a vertex, the vertex is truncated to a face which is guaranteed to be planar and will have valence-3 vertices. Therefore this operation is really handy for getting rid of the vertices with valences higher than three in the mesh. It is generally used in combination with planar edge truncation.

3. Face Based Cutting

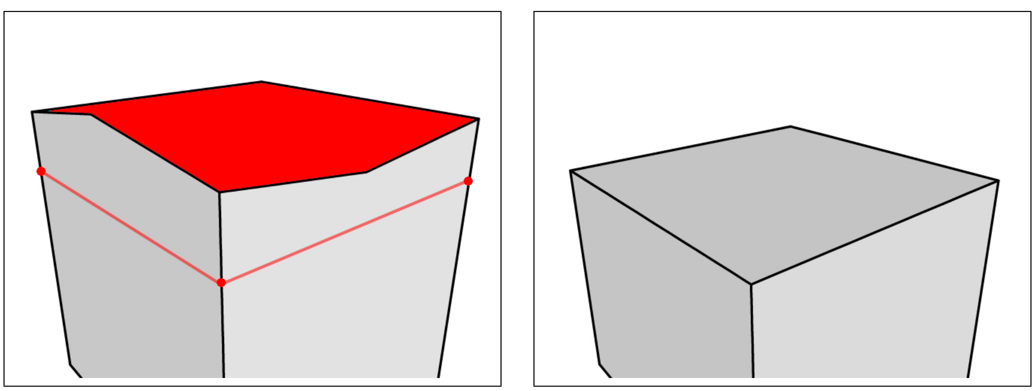
The existing face normal is used as a normal of the slicing plane. The position p_0 is again computed the same way as in the other operations as the average of all edge points. Planar face truncation applied to a face guarantees a planar face with valence three vertices around the face as seen in Figure 9.

This operation can be used for getting rid of non-planar faces. It also cleans up vertices with valences higher than three.



(a) (b)

Fig. 8. Planar vertex truncation: (a) The position and the normal of the slicing plane is determined relative to the selected vertex. (b) The result is a planar vertex truncation.



(a) (b)

Fig. 9. Planar face truncation: (a) The position and the normal of the slicing plane is determined relative to the selected face. (b) The result is a planar face truncation.

CHAPTER IV

METHODOLOGY

The fundamental idea behind the method introduced in this thesis is cutting a source mesh by a slicing plane. Determining the slicing plane becomes the main issue in this methodology. This is analogous to how a sculptor uses his hammer.

Determining the slicing plane is the process of assigning the desired values to the two parameters of the plane point P_0 and normal n_0. Technically, the user can be given free control to adjust these parameters by using interface elements such as sliders or rotation/translation handles. However, custom translation and rotation of the slicing plane may not be comfortable for the end user.

Alternatively, the method introduced in this work determines the slicing plane based on the elements of the mesh such as edges, vertices or faces. This is easily turned into an intuitive interactive process where the user can select the elements of the source mesh. In this approach, first the user is expected to select elements of the mesh such as edges, vertices or faces. Then the slicing planes are computed based on the selected elements. Finally, the cutting operation is performed by using the computed slicing planes.

The process of determining the slicing planes is not restricted to user interaction and can be automated by an algorithm as well.

A. Modeling Developable Surfaces

The interactive process provides a framework for users to design various developable shapes. The interactive use of planar edge truncation plays an important role in carving a developable surface. When the edges in a particular direction are continuously truncated a planar approximation to a smooth developable surface can be obtained. Figure 10 shows how an edge of a cube is smoothed out by truncating it in 3 consecutive iterations.

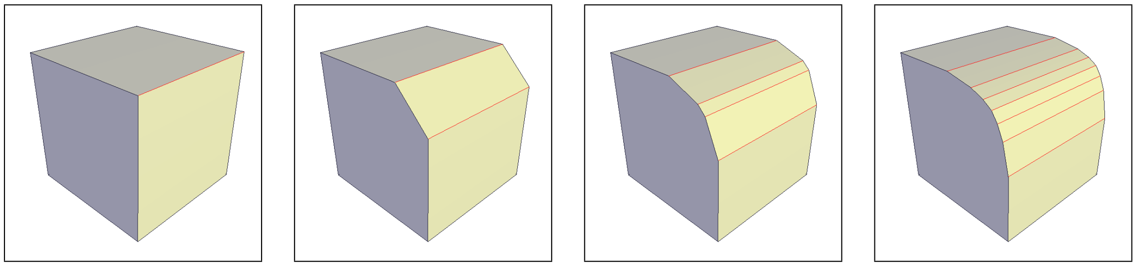


Fig. 10. The edge of a cube is smoothed out by truncating parallel edges.

B. Creating Patterns

Distinctive patterns can be created by selecting a series of edge and vertex combinations from the source mesh by using the interactive process. The best aesthetic patterns are achieved when the elements are selected by following a rule or symmetry in the consecutive iterations. Figure 11 shows an interesting pattern created by using the interactive process as mentioned above.

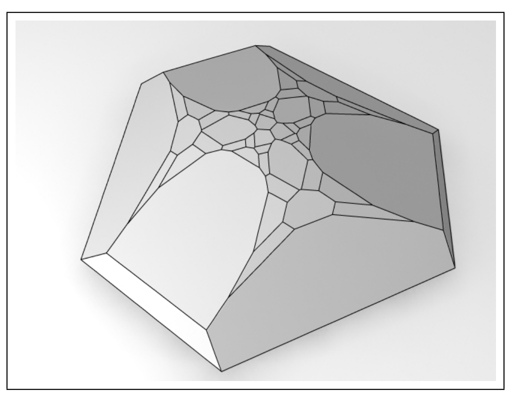


Fig. 11. A pattern created interactively by user

Applying the same type of truncation to all elements of the mesh is another way of creating patterns. Pattern variety can be increased by following different sequences of truncations. The mesh in Figure 12a is generated by applying a honeycomb subdi-

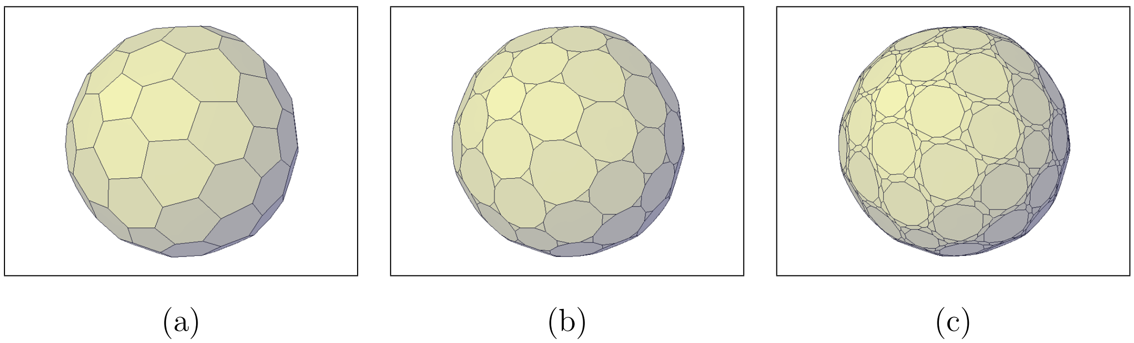


Fig. 12. (a) A honeycomb subdivision and planar face truncation applied to a soccer ball. (b) Planar vertex truncation applied to the shape successively. (c) Planar edge truncation is applied to the shape

vision [31] to a soccer-ball mesh. Planar face truncation is applied to the subdivided mesh to ensure planarity all over the mesh. When planar vertex truncation is applied to this shape, the resulting pattern is as demonstrated in Figure 12b. The pattern in Figure 12c is obtained by applying planar face truncation to the shape in Figure 12b. Other interesting results created by using the same approach are shown in Figure 13.

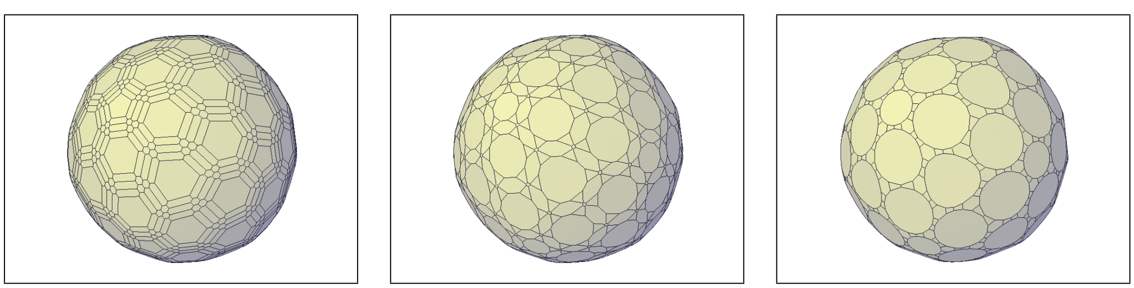


Fig. 13. Other interesting results achieved by using the same approach on the shape in Figure 12a

1. Dual Convex-Hull by Automated Process

Selecting the slicing planes relative to the elements of the mesh is an intuitive and interactive process. However, the selection process may not be restricted to the element based selection methods. The process can be automated as well.

Dual Convex-Hull algorithm is an example of an automated determination of slicing planes. The algorithm generates a planar dual for regular convex-hull of any given point cloud. Since it is a dual shape, triangular faces of a regular convex-hull become 3-valence vertices.

The algorithm takes a set of 3D points as an input. These points can be randomly distributed or be the vertices of a given mesh, thus generating a convex-hull from the points. Afterwards, a cube is generated as the initial mesh, whose faces are in contact with the boundaries of the convex-hull. Next, for all the vertices of the convex-hull, a slicing plane is determined by using the location and normal of each convex-hull vertex. Finally, the initial cube is cut by the computed slicing planes. The process is illustrated in Figure 14.

Dual convex-hull, as the name suggests, always gives a convex result. To create concave shapes, the concave input shape can be subdivided into convex regions and then processed by the algorithm. This subdivision can be both uniform or adaptive. Figure 15 shows the dual convex-hulls of a model. The uniform subdivision is done by a uniform clustering of the input points along one particular axis. For the adaptive subdivision example, I separated the mesh to convex regions manually.

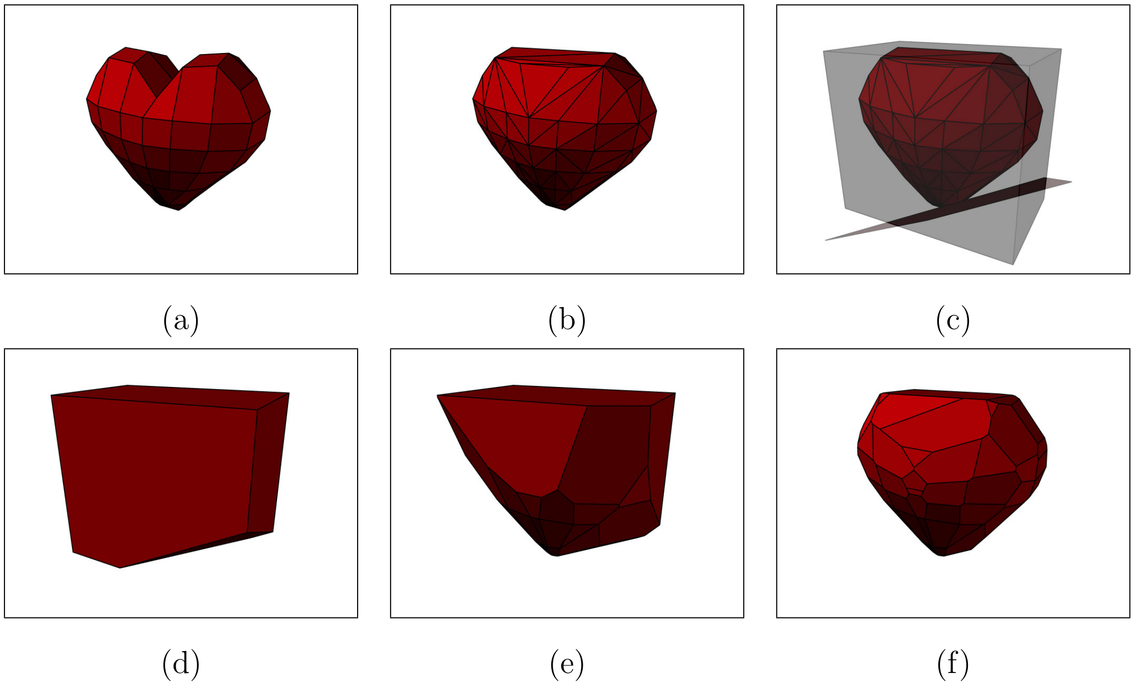


Fig. 14. Step by step dual convex-hull algorithm. (a) Initial mesh. (b) Convex-hull of the initial mesh. (c) Bounding box. (d) Cutting for the first vertex. (e) Cutting for the 20th cutting vertex. (f) Final mesh.

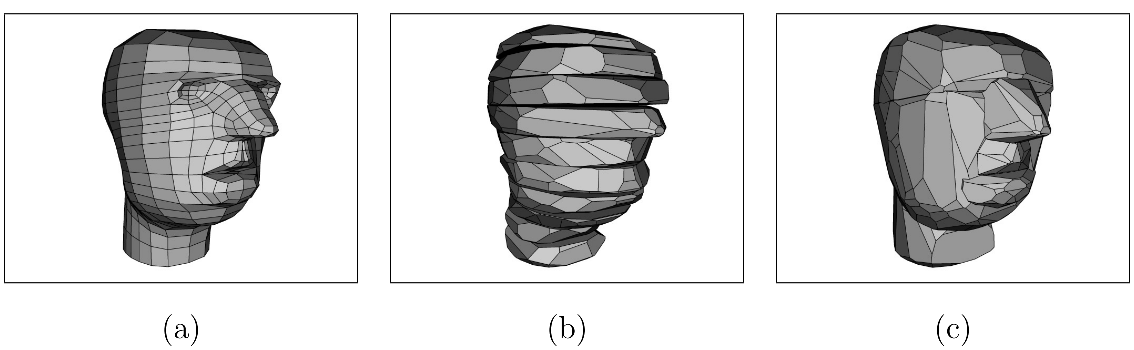


Fig. 15. Dual convex-hull results with subdivison: (a) Input mesh. (b) Dual convex-hull with uniform uniform clustering along y axis. (c) dual convex-hull with manual subdivision.

CHAPTER V

IMPLEMENTATION AND RESULTS

I have implemented the planar truncation operation in the topological mesh modeling software, TopMod [3, 4]. I provide three different tools: *Cut by Edge*, *Cut by Vertex*, and *Cut by Face*. Users can adjust the default parameters of the slicing planes. Figure 16 shows a screenshot from the TopMod software.

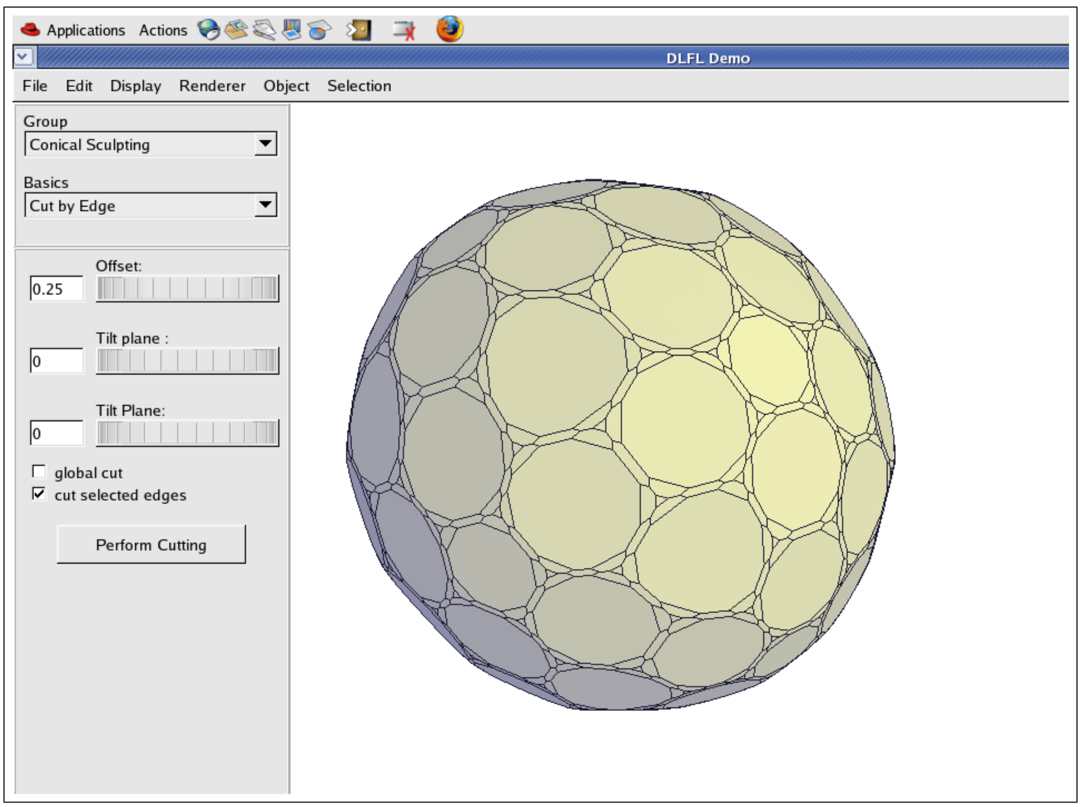


Fig. 16. A screenshot from the implementation of the method in TopMod

The slider controlled "offset" value adjusts the offset amount used in the edge point computations. Therefore it determines how deep the slicing plane is positioned from the selected element.

The “Bent” control enables the user to tilt the slicing plane in by changing weights of the normals of the faces adjacent to the edge.

In the interactive process, the user is expected to mark elements of the initial mesh such as edges, vertices or faces. The marking mode can be switched from the dropdown menu. When the user presses the “perform cutting” button, all the slicing planes are computed based on the marked elements and the respective cutting function (global/local) is called for each slicing plane computed. The cutting function can be optionally chosen from the interface.

The framework enables the user to sculpt developable surfaces in various shapes. The best results are achieved when the truncation tools are used in consecutive iterations. The results of this work can be classified in two main categories: (A) Developable Surfaces (B) Patterns.

A. Developable Surfaces and D-Forms

The basic interactive process enables the user to create various developable shapes. Interactive use of planar edge truncation plays an important role in carving a developable surface.

Unusual variations can be introduced to the developable surface by truncation of edges consecutively in different orientations. Figure 17 shows some interesting surfaces created with this technique.

The framework also provides an alternative computational method to physical D-form [22] construction. Despite its power to construct unusual shapes easily, there are two problems with physical D-form construction. First, the physical construction is limited to only two pieces. It is hard to figure out the perimeter relationships if we try to use more than two pieces. The second problem with D-form construction

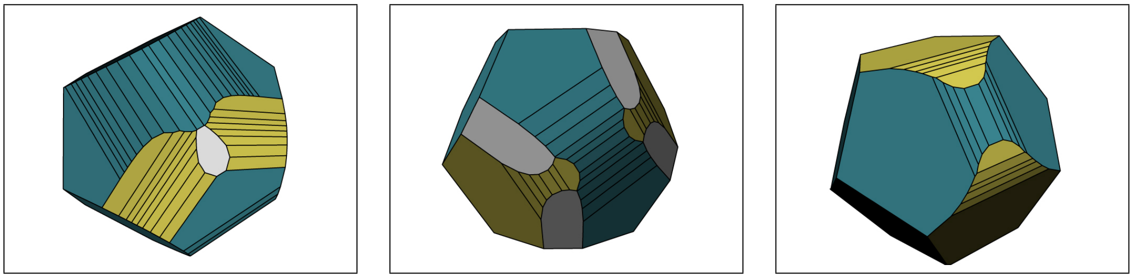


Fig. 17. Some examples of developable surfaces with variations

is that the final shape is not known until it is physically constructed

Using the basic interactive process, it is possible to directly model D-forms which are not limited to two pieces. Another advantage of the method is that since the user actually sculpts the D-forms, the final shape can be visualized in each step before the physical construction.

D-forms designed with this framework can be unfolded using Pepakura, a commercially available polygonal unfolding software [32]. Once unfolded, the pieces can be cut using a laser cutter and glued together to create physical D-forms. Using this method, D-forms that were not known before were created.

Figures 18 and 19 show D-forms sculpted out of a dodecahedron by using our planar truncation operation. The three piece case in Figure 18 is particularly interesting since the long piece touches itself. This suggests that it may be possible to construct a D-form using only one piece, although we have not been able to find one. The D-form in Figure 20 is also interesting in the sense that both unfolded pieces have a **Y** shape. Figure 21 shows some other unusual D-form examples that consist of more than two pieces.

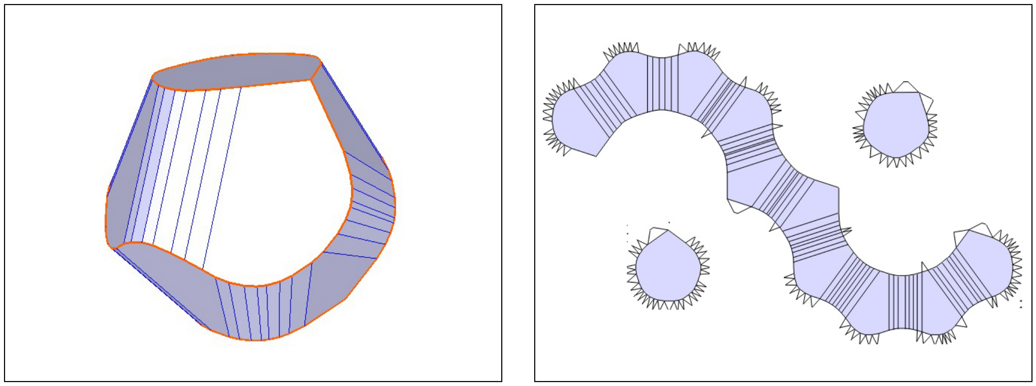


Fig. 18. A dodecahedron constructed from 3 pieces. This D-form is particularly interesting since the middle piece turns and touches itself.

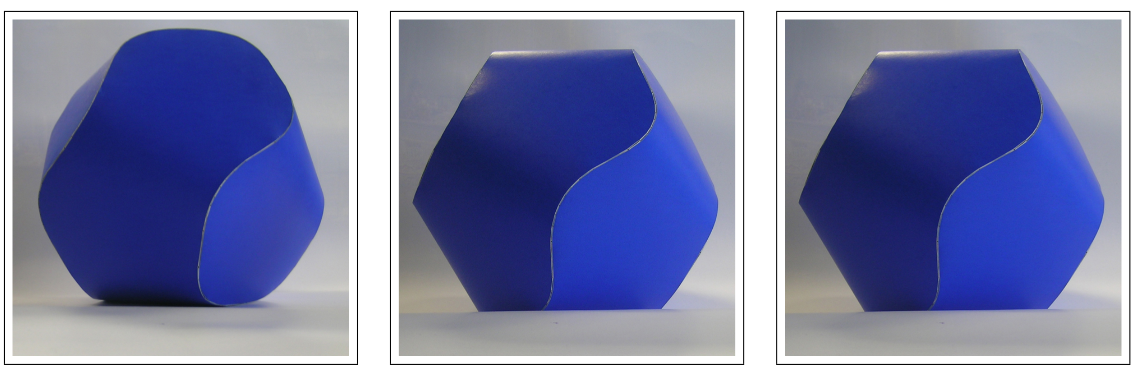


Fig. 19. Three views of a D-form constructed using our method starting from a dodecahedron. This shape is designed using our software by Ergun Akleman. This D-form consists of two pieces. The computer designed and unfolded versions of this D-form are shown in Figure 18. Jonathan Penney combined the unfolded pieces to create final physical D-forms.

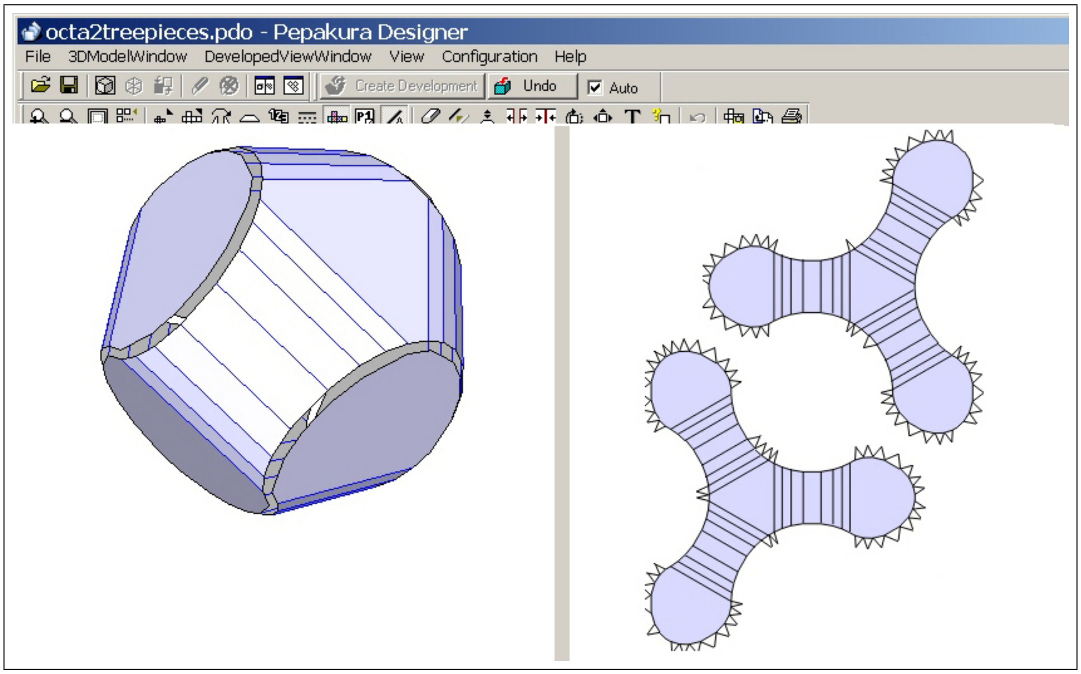


Fig. 20. Unfolding a D-form in Pepakura. This D-form is obtained from an octahedron. Note the **Y** shape of unfolded pieces.

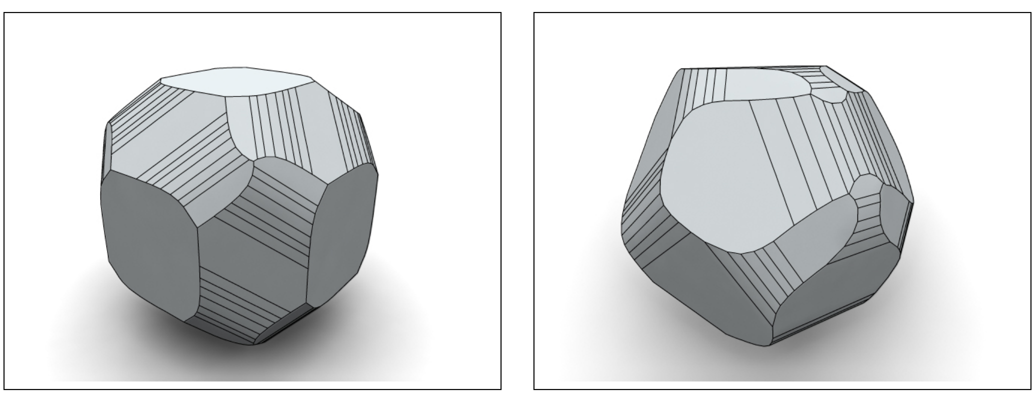
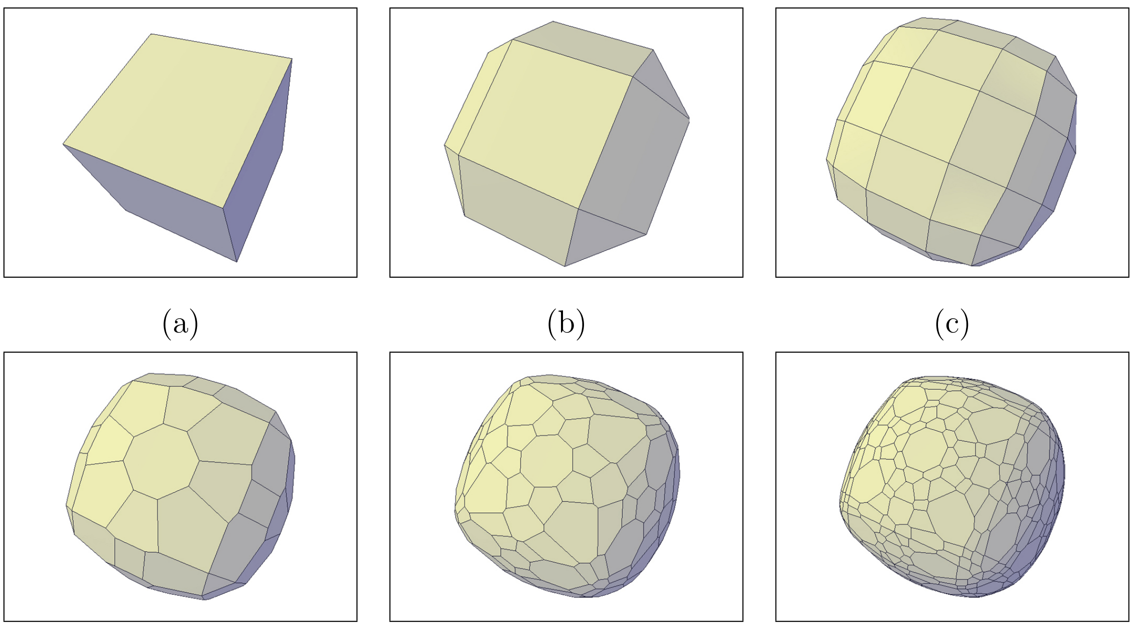


Fig. 21. Some unusual D-forms that consist of more than two pieces.

B. Patterns

Distinctive patterns can be created by selecting a series of edge and vertex combinations from a source mesh. Applying the same type of truncation to all elements of the mesh is another way of creating patterns. Pattern variety can be increased by following different sequences of truncations. Tweaking the parameters like offset value and normal bent will also introduce variations in the results.

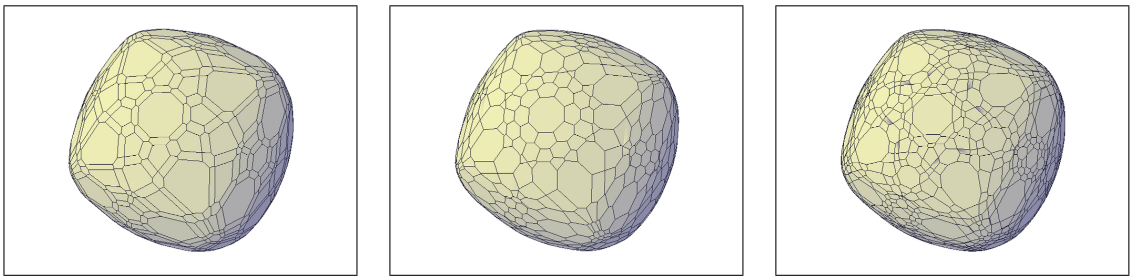


(a) (b) (c)

(d) (e) (f)

Fig. 22. Step by step creation of a pattern: (a-c) Two iterations of Doo-Sabin subdivisions are applied to a cube. (d) Planar face truncation is applied with 0.75 offset value to all faces . (e-f) Planar vertex truncation is applied for 2 iterations with 0.75 offset value.

In Figure 22, starting from a cube, the creation of a pattern using this method is illustrated step by step. Two iterations of Doo-Sabin subdivisions [33] are applied to the cube create an initial mesh. Then planar face truncation is applied to all faces with a 0.75 offset value of edge points. Finally two consecutive iterations of planar vertex truncation is applied to all vertices of the mesh with a 0.75 offset value to create the final pattern.

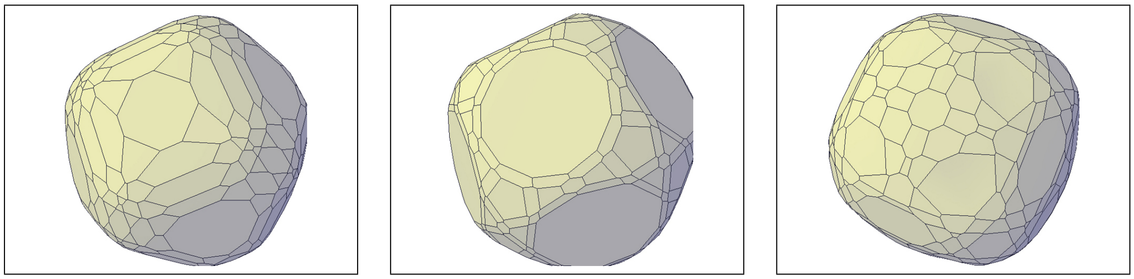


(a) (b) (c)

Fig. 23. Other patterns based on the initial mesh in Figure 22e

Figure 23 demonstrates the variety of patterns that can be achieved by applying different sequences of cutting operations to the same initial mesh. The pattern in Figure 23a is created by applying planar edge truncation to all edges of the mesh in Figure 22e with 0.75 offset value. Applying planar face truncation to all faces of this pattern gives the result in Figure 23b. The pattern in Figure 23c is created by applying two consecutive planar vertex truncations with offset value of 0.75 to all vertices of the shape in Figure 22e as well.

In Figure 24 another initial mesh is prepared by using a honeycomb subdivision scheme [31]. The pattern in Figure 24a is obtained when planar edge and vertex truncations are sequentially applied to a cube that is subdivided using the honeycomb scheme. When planar vertex truncation with offset of 0.5 and planar edge truncation with offset of 0.25 is applied to this shape, the pattern in Figure 24b is obtained. For creating the pattern in Figure 24c, a planar edge truncation with offset of 0.25 and honeycomb subdivision are applied to a cube. Then, planar vertex and edge



(a) (b) (c)

Fig. 24. Some patterns which are based on an initial mesh subdivided by honeycomb subdivision.

truncations with offset of 0.5 are successively applied to the shape.

CHAPTER VI

CONCLUSION AND FUTURE WORK

The new concept of conical meshes are particularly suitable for the design of freeform glass structures. In this thesis, I have presented a computational method for modeling conical meshes directly. The method mostly introduces valence three vertices with planar faces all over the mesh which are proven to be always conical [29]. Moreover valence three vertices are advantageous since they ensure a stable structural analysis that matches the forces in the constructed frame.

The outcomes of this method can be classified in two main directions: when the user marks parallel edges of the mesh in each iteration, the method provides an alternative to physical D-form construction. The computer generated D-forms can be unfolded using commercially available software and cut using a laser cutter. Physical D-forms can be obtained by putting the unfolded metal or paper pieces together. Using this method it is possible to create complicated D-forms that cannot be constructed without a computer. One of the major advantages of our D-forms is that they are created as conical meshes and can therefore be constructed at larger scales even from thick and planar materials like glass or sheet metal.

The user can also introduce interesting conical patterns all over the mesh by selecting the elements in a regular pattern. These aesthetically pleasing patterns can show varieties in shape but can still be perceived as belonging to the same family due to their distinctive look and feel.

A. Future Work

Currently this method works only for convex shapes. The ways of generalizing this method to non-convex shapes and shapes with saddle points can be explored.

The results of the method are heavily dependent on how the slicing plane is determined. An interactive process that uses the elements of the mesh to determine the slicing plane is introduced. The Dual Convex-Hull algorithm is given as an example for an automated process. Yet, other ways of determining a slicing plane can be explored. More interesting results can be achieved with different slicing plane determination algorithms.

REFERENCES

[1] Y. Liu, H. Pottmann, J. Wallner, Y. Y. Tand, and W. Wang. "Geometric Modeling with Conical Meshes and Developable Surfaces," *ACM SIGGRAPH*, pp. 681-689, 2006.

[2] B. Cutler, and E. Whiting, "Constrained Planar Remeshing for Architecture," *Poster Presentation at Eurographics Symposium on Geometry Processing*, Cagliari 2006.

[3] E. Akleman. (2006, November). *TopMod Website*. Available: http://www-viz.tamu.edu/faculty/ergun/research/topology/

[4] E. Akleman, J. Chen, and V. Srinivasan, "A Minimal and Complete Set of Operators for the Development of Robust Manifold Mesh Modelers," *Graphical Models Journal*, vol. 65, no. 2 pp.286-304, 2003.

[5] Ohio State University, (2006, December). *British Museum Photo*. Available: http://rclsgi.eng.ohio-state.edu/cash.36/London/

[6] Hotel Accursio, (2006, December). *Fierra Milano Photo*. Available: http://www.accursiohotel.it/img/location

[7] R. Sauer, *Differenzengeometrie*, Berlin: Springer-Verlag, 1970.

[8] M. Leiter. (2006, November). *Kyoto Bus Terminal Photo*. Available: http://www.tropicalisland.de/japan/kyoto/kyoto.html

[9] Flickr, (2006, November). *Chadstone Mall Photo*. Available: http://www.flickr.com/photos/tiomo/204595031/

[10] E. W. Weisstein, (2006, November). *Quadrilaterals.* Available: http://mathworld.wolfram.com/Quadrilateral.html

[11] F. Gehry, (2006, November). *Freeform Glass Structures.* Available: http://www.gehrytechnologies.com/

[12] J. Glymph, D. Shelden, C. Ceccato, J. Mussel, and H. Schober, "A Parametric Strategy for Freeform Glass Structures Using Quadrilateral Planar Facets," *Acadia ACM*, vol 15, pp. 303-321. 2002

[13] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Levy, and M. Desbrun, "Anisotropic Polygonal Remeshing," *ACM Trans. Graphics* vol. 22, no. 3, pp. 485493, 2003.

[14] E. W. Weisstein, (2006, October). *Developable Surfaces.* Available: http://mathworld.wolfram.com/DevelopableSurface.html.

[15] E. W. Weisstein, (2006, October). *Ruled Surfaces.* Available: http://mathworld.wolfram.com/RuledSurface.html

[16] J. Maertterer, (2006, October). *Developable Surfaces.* Available: http://www.rhino3.de/design/modeling/developable/index.shtml

[17] J. Maertterer, (2006, October). *Developable Surfaces in Marine Design.* Available: http://www.rhino3.de/design/modeling/developable/marinedesign.shtml

[18] M. Sun, and E. Fiume, "A Technique for Constructing Developable Surfaces," in *Proc. Graphics Interface*, Toronto, Ontario, Canada, pp. 176 - 185, 1996.

[19] C. H. Chu, and C. Sequin, "Developable Bezier Patches: Properties and Design," *Computer-Aided Design*, vol. 34, pp. 511-528, 2002.

[20] P. Haeberli, (2006, November). *Lamina Design Software.* Available: http://laminadesign.com/index.html

[21] J. Mitani, and H. Suzuki, "Making Papercraft Toys from Meshes Using Strip-Based Approximate Unfolding," *ACM Trans. Graph.* vol. 23, no. 3, pp. 259-263, 2004.

[22] T. Wills, "D-forms," presented at the *Proc. Bridges*, London, 2006. pp. 134-139.

[23] J. Sharp, "D-forms and Developable Surfaces," presented at the *Proc. Bridges*, Banff, Canada, 2005 pp. 121-128.

[24] H. Pottmann, and J. Wallner, *Computational Line Geometry*, Berlin, Germany: Springer-Verlag, 2001.

[25] E. D. Demaine, and J. O'Rourke, "Open Problems from CCCG 2002," in *Proceedings of the 15th Canadian Conference on Computational Geometry (CCCG 2003)*, pp. 178-181, Halifax, Nova Scotia, Canada, August 11-13, 2003.

[26] J. Sharp, "D-forms: Surprising New 3D Forms from Flat Curved Shapes," Hertfordshire, England: Tarquin, 2005.

[27] R. Evans. (2006, October). *Plexons Created by Paul Bourke.* Available: http://astronomy.swin.edu.au/ pbourke/geometry/plexagon

[28] P. Bourke. (2006, October). *D-Forms.* Available: http://astronomy.swin.edu.au/ pbourke/surfaces/dform/

[29] J. Wallner, Prof. at Institute of Discrete Mathematics and Geometry, University of Technology in Wien, private communication, September 2006.

[30] Quotations Page, (2006, September). *Michelangelo's Quote*. Available: http://www.quotationspage.com/quote/36775.html

[31] E. Akleman, and V. Srinivasan, "Honeycomb Subdivision", in *Proc. 17th International Symposium on Computer and Information Sciences*, pp. 137-141, November 2002.

[32] Tama Software (2006, September). *Pepakura Designer*. Available: http://www.tamasoft.co.jp/pepakura-en/

[33] D. Doo, and M. Sabin, "Behavior of Recursive Subdivision Surfaces Near Extraordinary Points", *Computer Aided Design*,vol 32, no. 10, pp. 356-360, 1978.

VITA

Ozgur Gonen earned the degree of Bachelor of Arts in Visual Communication Design from Istanbul Bilgi University in 2003. He worked as a Multimedia Developer before joining the Master of Science in Visualization Sciences program at Texas A&M University. He mostly worked with Dr. Ergun Akleman during his masters studies. He can be reached at ozgur@viz.tamu.edu.

C418 Langford Center 3137 TAMU College Station TX 77843-3137