Draping Curves on Surfaces for Quad-Meshing of Architectural Surfaces

Sai Kit Fan[1], Ajay Joneja[2] and Shawn Wang[3]

[1,2,3]Hong Kong University of Science & Technology, {iesaikit, joneja, shawnw}@ust.hk

ABSTRACT

Modern architectural projects increasingly use free form curved surfaces. These are eventually fabricated out of a large set of simpler pieces, or tiles that are more economical to manufacture and maintain. The most common tiling scheme is quad-meshing. Typical tiling approaches work in two stages: an initial pattern of lines is generated to yield an initial tiling, followed by an adjustment stage that optimizes the precise geometry of each tile. In this paper, we introduce an approach to address the first stage: draping of aesthetic 2D curves onto 3D surfaces in order to create initial tiling patterns. We test our approach by using a common aesthetic curve, the Archimedean spiral. Several examples from real-world projects are used to illustrate our method.

Keywords: draping, architectural geometry, quad-meshing, surface re-meshing.
DOI: 10.3722/cadaps.2013.293-306

1 INTRODUCTION

Over the past couple of decades, there has been a remarkable increase in the number of landmark architectural projects across the world. This increase has manifested itself in the form of many iconic skyscrapers, as well as other notable large-sized buildings. Some beautiful examples include the Beijing National Stadium (a.k.a. The Birdĝs Nest), the Al Raha Race Track in Abu Dhabi (Fig. 1) and the Qunli Museum in Harbin, China (Fig. 2). Firms led by leading architects such as Frank Gehry and Zaha Hadid are increasingly employing free-from surfaces in their designs. Examples of such curved shapes appear in different parts of buildings, e.g. curtain walls of skyscrapers (National Holdings HQ in Abu Dhabi), canopy (the Venetian Casino in Macau) and roofs (the old USPTO building, Washington D.C.).

Fig. 1: Al Raha Race Track, Abu Dhabi

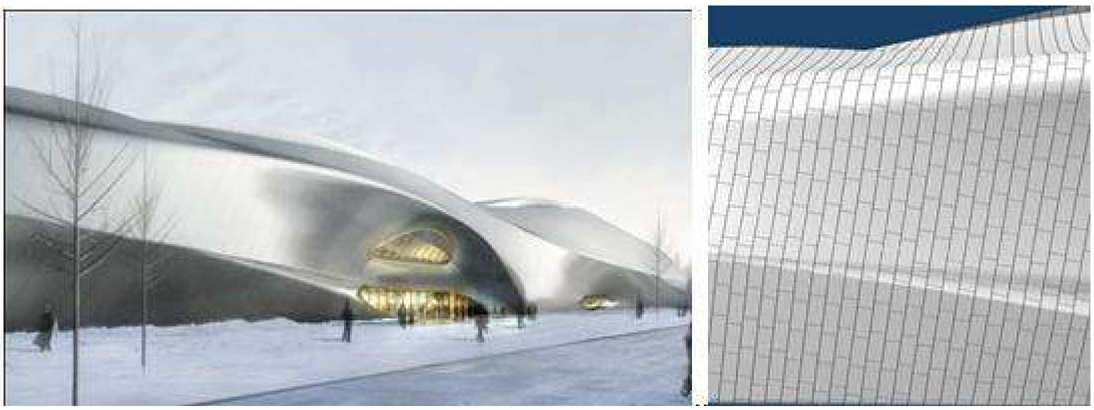


Fig. 2: (a) The Qunli Museum, Harbin (b) Close up showing skin composed of quad-panels.

Practically, such surfaces are fabricated by assembling a series of (almost) regular shaped tiles. In other words, to economically manufacture such a surface, it must first be approximated by a tessellation. Ideally, the tessellation should possess some properties that help to reduce the fabrication costs: for example, each facet should be planar, or failing that, developable. It is even better if the facets are isotropic. Such tessellation often imposes a particular look to the building, manifested in the form of patterns of lines; it is also desirable that these ôpatternsö and the ölinesö composing them should contain some aesthetic qualities. Some of these qualities are quantifiable (e.g. fairness of a curve is often related to how smooth it is in space).

Subdivision of curved surfaces into isometric pieces has been an important engineering problem for several decades. An early application was in Finite Element and Finite Difference Methods George 98]; later, it also found various applications in Computer graphics, for example, see [Alliez 03, Alliez 07, Boissonant 01, Garland 98]. The problem has arisen in a slightly different flavor in modern architecture design [Eigensatz 10, Fu 10, Singh 10]. These researchers addressed the rationalization of meshes, solving for K-set tiling approximations of surfaces; the objective is to find the smallest integer K such that each tile of the tessellation belongs one of K classes, and each member of a class is geometrically equivalent. The idea is that a tessellation with small K, say less than 100, is relatively economical to fabricate. In [Singh 10], the authors propose a method for K-isotropic triangulations, using a two stage algorithm. In the first stage, the initial triangulation is clustered using a variant of k-means clustering, resulting in a representative triangle, or exemplar, for each cluster. The second stage is a global optimization where vertices are allowed to move in order to conform to one of the k exemplars. In [Fu 10], the problem of K-set tiling of quad-mesh approximations of the input surface is considered. In this work, the input is a quad mesh, and the initial stage, as in [Singh 10] follows a K-means clustering of the initial mesh. The subsequent global optimization, which is also achieved by moving vertices, is set up via introduction of penalties for shared edge length mismatches as well as for the degree of violation of fairness of internal vertices. The notion of fairness introduces an analytical means for controlling the aesthetics of the meshing, which is an important consideration in architecture. Another group that has made several practical and theoretical contributions in this area is led by Pottmann [Eigensatz 10, Pottmann 07, Pottman 07sig, Pottmann 10]. Among their significant contributions are techniques for generation of quad- and hexagonal meshes. In particular, an algorithm for creating optimal near-planar quad meshes is introduced in [Eigensatz 10]. In a more recent work, they proposed an approach that optimizes quad meshes by perturbation of the mesh vertices; each quad is fit (within a user-defined tolerance) to the original surface by means of an efficient six-degree-of-freedom optimizer. They also investigated quad meshing of surfaces using geodesic patterns [Pottmann 10], with application in skin/curtain-wall design. Pottmann also investigated the use of variable sized hex-meshes to tessellate surface of buildings designed using sets of revolute surfaces [Pottmann 10]. However, this work does not address the issue of standardization (or isotropy) of the meshes. In practice, mesh standardization has significant economic and aesthetic significance: if each panel is geometrically distinct, the subsequent panel construction as well as assembly is prohibitively expensive. Furthermore, since arbitrarily curved surfaces are not developable, therefore the next cheaper solution adopted in practical construction is to use some restrictions of piecewise developable paneling, e.g. if each panel can be approximated (to within a user-specified tolerance) to a conical surface.

One issue that remains somewhat unexplored is that of the generation of the initial pattern of lines that defines the aesthetic look of the building, but also heavily constrains the subsequent optimization process (see also [Eigensatz 10]). As far as we know, there are only a few high level approaches that have been explored in this regard: (a) use isoparametric curves after attempting to re-parameterize the surface, (b) using networks of principal curvature lines, (c) using geodesic curve patterns (for example, [Pottmann 10] describes a method using level sets to generate patterns of geodesic curves to subdivide a surface), or (d) projection/draping of curve patterns from 2D onto the surface (e.g. an initial solution in the Qunli project was generated by draping a planar line pattern onto the surface, see Figure 3). In our experience, solutions using principal curvature lines tend to be erratic, especially on surfaces that may have some regions that are nearly flat. On the other hand, the latter approach suffers because (a) the initial curve is chosen in a rather ad hoc fashion, and as we progress along the surface, there is little control on the way that geodesic offset curves will lie, and (b) computing these patterns is relatively slow, so this format is not really convenient for iterative design to try out several alternatives.

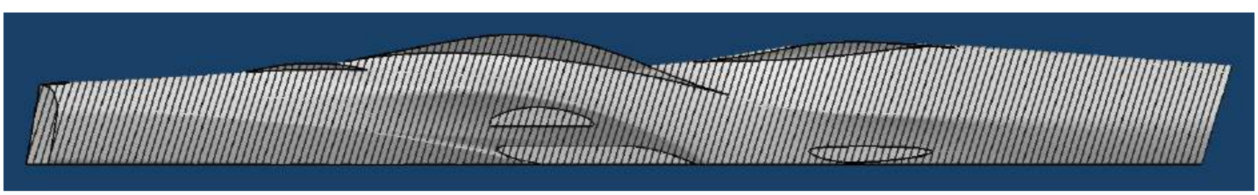


Fig. 3: Initial pattern of curves for subsequent panelization.

In this paper, our main goal is to provide a relatively fast mapping approach to generate panelization patterns on the input surfaces, which allows the user to quickly test out various alternative design and select perhaps one or a few for further optimization. In particular, the paper makes the following contributions:

(1) We propose a novel method to panelize a curved surface by draping patterns of 2D curves onto it;

(2) We propose and demonstrate that classical aesthetic curves can be used to generate interesting and practical patterns for new designs

The rest of this paper is organized as follows. In the next section, we give the details of our approach. The proposed approach has been programmed, and tested using several real-world examples. The results of some of these tests are shown in Section 3. Finally, we close off the paper with some discussions.

2 METHODOLOGY

We briefly outline our approach before giving the details. We implement most of the algorithm in discrete space, for reasons of computing efficiency. The input surface is initially triangulated, using a reasonably fine tolerance (a good practical value is 0.1x the maximum allowed tolerance of the tessellation). The triangulated surface is then mapped to a 2D equivalent shape that minimizes the isometric distortion. The desired mesh pattern is then designed by superposing it onto the 2D image of the surface. This pattern is then mapped back to the 3D domain (one can think of this as the inverse map of the 3D to 2D flattening). The only time consuming step in this entire process is the construction of the 2D development of the surface **ð** and since this operation is required only once, at the outset, therefore it is quite convenient for the designer to rapidly test out various configurations of the meshing. Once a pattern is selected, we can post-process it to optimize the mesh quality to minimize the fabrication costs, e.g. using any of the approaches referenced above.

2.1 Generating the 2D Mapping (near) Isometric Map of the 3D Surface

Surface development is an important problem with various engineering applications. Several approaches have been proposed in the past, most of which begin with a triangulated approximation of the input smooth surface, and employ an optimization model that minimizes some measure of the distortion induced by the parameterization or flattening. McCartney et al treated each edge of the triangulation as a spring and generated a flattened surface that minimized the spring energy [McCartney 99]. However, they would impose darts and gussets in regions where distortion was high. Their approach was enhanced later by considering a mechanics-based model that included strain and shear energy in [Tam 2007]. Faux and Pratt [Faux 79] note that the geodesic curvature can be used for surface development. This observation was used in several subsequent papers [Manning 80, Bennis 91, Azariadis 00], each using progressively sophisticated optimization models to achieve flattening.

An alternative approach, which we found to be somewhat more efficient and robust, was based on the notion of conformal maps and proposed first in [Levy 02]. Conformal maps preserve local isometry; thus the mapping f: S(u, v) → Sô(x, y), satisfies the property: N(u, v) X $\partial f/\partial u = \partial f/\partial v$, where N(u, v) denotes the unit normal of S at (u, v). Let f^{-1}: S'(x, y) → S(u, v) be the inverse map of f. The inverse function theorem from differential geometry states that $J(f^{-1}) = [J(f)]^{-1}$, i.e. the Jacobian of f^{-1} is merely the matrix inverse of the Jacobian of f. Therefore to minimize the conformal error in the discrete setting, we can use an objective function (using the complex algebra notation expressing (u, v) as u + *i*v), as:

Minimize $C = \sum \left| \frac{\partial f^{-1}}{\partial x} + i \frac{\partial f^{-1}}{\partial y} \right|^2$, where A_i is the area of the i-th triangle in the approximation.

Assuming that the triangulation is fairly dense, one can then assume that the parameterization varies linearly within any triangle, which allows the formulation to be formulated into a matrix form that is solved robustly, as long as the initial shape is not too contorted. We use an implementation of Levy's approach that is available via CGAL [CGAL] in this project.

2.2 Designing Archimedean Spiral Patterns in Parameter Space

In this paper, we mainly focus our attention on generation of patterns of Archimedean spirals. Written in 2D polar coordinates, (r, θ), as: $r = a + b\theta$, this curve has an interesting property of maintaining constant width (= b) between successive turns of the spiral. This is particularly convenient for our application, since it yields panels that will maintain (nearly) constant width, and are therefore more amenable to size standardization. Before proceeding to the details of our approach, it is interesting to note that there are other classical aesthetic curves that may also be used to generate other patterns on surfaces, e.g. the so called Log-aesthetic curves [Miura 06]. If we parameterize a curve by arc length, s, and denote the curvature of the curve by ρ, this class of curves is expressed by the formula: $\frac{1}{\rho^{a-1}} \frac{ds}{d\rho} = c$, where a and c are constants. This family of curves includes logarithmic spirals, clothoids and involutes. Clothoidal curves, where the curvature varies linearly with the arc length may be specially interesting for our application, since the notion of controlled curvature change is related to that of fairness commonly used in generating smooth curves on surfaces. How such curves can be used to generate nearly standardized quad-meshing panels will be explored by us in the future. For now, we return our attention to Archimedean spirals. We propose two different approaches for panelization below.

Method 1. Spiral with Radial spines:

This approach is interesting if the center of the spiral is located in the interior of the parameterized shape (see Figure 4). The user specifies the center point, and the nominal width (corresponding to the coefficient b in the equation of the spiral) of the panels, and the angular separation between the radial arms. This is sufficient to generate the pattern using the spiral and a series of radial rays emanating outwards from the center of the spiral, and intersecting with the spiral to yield the quad mesh vertices. Subsequently, the curved segments of the spiral on each panel are then replaced by line segments to yield the quad-mesh.

Method 2. Two intersecting Spirals

In this case, the user is allowed to place two spirals, each with the center point located relatively far from the envelop of the parameterized shape. Additionally, the user specifies the pitch, b_1, and b_2, for the two spirals respectively. By proper location of the centers (for example, if the median of the parameterized surface is located at the origin, the center of the first spiral may be located at (p, 0), and that of the second one at (0, q) for some large values p and q. The resulting spiral patterns intersect to give a quad-mesh, and again, by replacing the spiral segments bounding each panel by corresponding line segments between the end points, we get a quad mesh (see Figure 5). Note that by changing the relative pitch of the spirals, the user can control the aspect ratio of the rectangles in the mesh pattern, from squares to long rectangles.

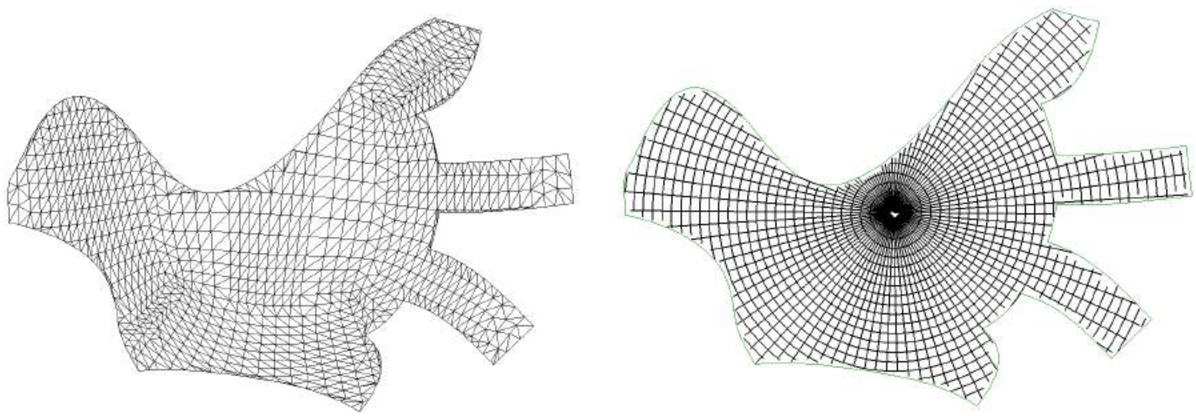


Fig. 4: (a) 2D parameterization of a 3D surface (b) A quad-meshing pattern using method 1

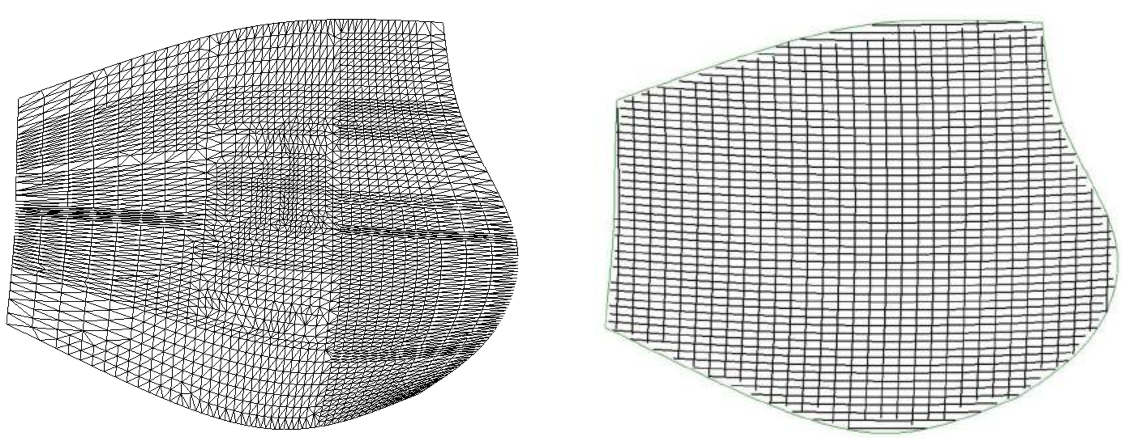


Fig. 5: (a) 2D parameterization of a 3D surface (b) A quad-meshing pattern using method 2; due to space limitations, the centers of the two Archimedean spirals are not shown here.

2.3 Inverse Mapping of the Patterns to the 3D Surface

An important objective for our approach is to provide a responsive mechanism for (almost) real-time feedback to the designer during the design process. As can be seen from the previous sub-section, the process of overlaying the specified patterns of mesh over the 2D parameterization is fairly quick. Furthermore, the process of lifting the resulting quad meshes back to the 3D domain is also quite efficient. Here we outline the main procedure:

(i) We maintain a one-to-one correspondence between the triangles in the 3D and the 2D domains. Since the parameterization method is designed to disallow any triangle overlaps, therefore this map is well defined. The triangulation is stored in a hierarchical data structure using Kirkpatrickös scheme [Kirkpatrick 83].

(ii) For each vertex of the quad mesh in the 2D map, we locate the corresponding triangle containing it. This can be achieved optimally in O(logn) time using Kirkpatrickös algorithm. Here, n refers to the number of triangles in the original approximation of the surface.

(iii) We then compute the Barycentric coordinates of the point in the triangle; use these coordinates to find the equivalent 3D coordinates in the corresponding triangle in the 3D domain.

We only need to perform step (i) once, at the outset, and after the parameterization. Step (iii) requires constant time for each vertex. Therefore, to map a given pattern back to 3D requires O(m logn) time, where m is the number of vertices in the quad-mesh. Notice that the tolerance used in the triangular approximation (we use a value of 1mm for typical surfaces that can be up to 500-2000 m^2) is typically much smaller than the tolerance between the final panel and the original surface (between 10-20mm). Therefore, the number of quad panels in the approximation is typically smaller than the number of triangles by an order of magnitude. In other words, we usually have $m \approx 0.1n$, and therefore the inverse mapping is fairly efficient.

2.4 Evaluating the Quality of the Quad Mesh

The chief goal of our approach is to provide a tool for generating initial, aesthetic designs of quad panels on architectural surfaces. However, it is also important for such a design to be realized economically in practice. As discussed in the introduction, a useful way to achieve fabrication economy is to approximate each panel by one that is fabricated cheaply. Ideally, we would like to use all planar panels of the same size **ð** but this is impossible to achieve while using reasonable sized panels, except perhaps for polyhedral building shapes. The next best approximation is to approximate all panels by easily manufactured developable shapes: in particular, the most versatile mechanism is to use cone-approximations (it is relatively cheap to construct flexible fixtures that bend flat sheets to arbitrary conical surfaces). To evaluate if a design made by our approach will be economical, we use the following method: we approximate each quad-panel created by using our method to a cone. The statistics of the tolerance on this approximation over the entire panelization indicates whether the design can eventually be optimized into an economically viable one. The cone fitting is described briefly below.

(i) Let v1,,v4 denote the vertices of a quad panel in the 3D domain, with the line segments joining them denoted by (v1, v2), ..., (v4, v1), as shown in Figure 6. We project these edges onto the designed surface to obtain the corresponding 4-sided panel surface.
(ii) A randomly distributed sample of 200 points is generated within this panel.
(iii) These points, in addition with the four vertices, are used as sample points to fit a cone by a simple least squares regression (the distance of the point from the cone is computed as shown in Figure 6. Note that the regression requires solving a minimization over six variables (four real numbers locate cone axis, e.g. by the x and y coordinates where it crosses the XY plane, and the latitude and longitude angles; in addition, we need two real numbers to locate the cone point and the cone angle).

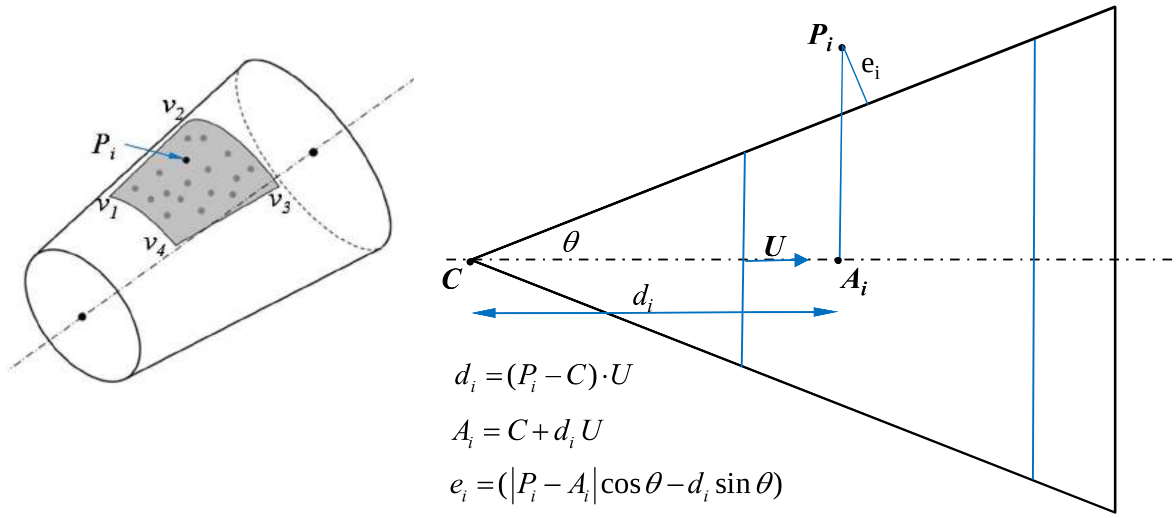


Fig. 6: Approximating the surface corresponding to a quad-mesh facet to a cone-shaped panel.

Figure 6 shows the schematic of the cone fitting. Here we assume that the cone parameters to be solved for are the cone center C, the unit vector along the cone axis U, and the cone half-angle θ. The figure shows the expression for the error of a sample point Pi from the cone. The cone fitting is done by minimizing the sum of the squared error of the samples by using the built-in solver in Microsoft excel, which is directly linked from the CAD system via an API interface.

3 IMPLEMENTATION AND EXAMPLES

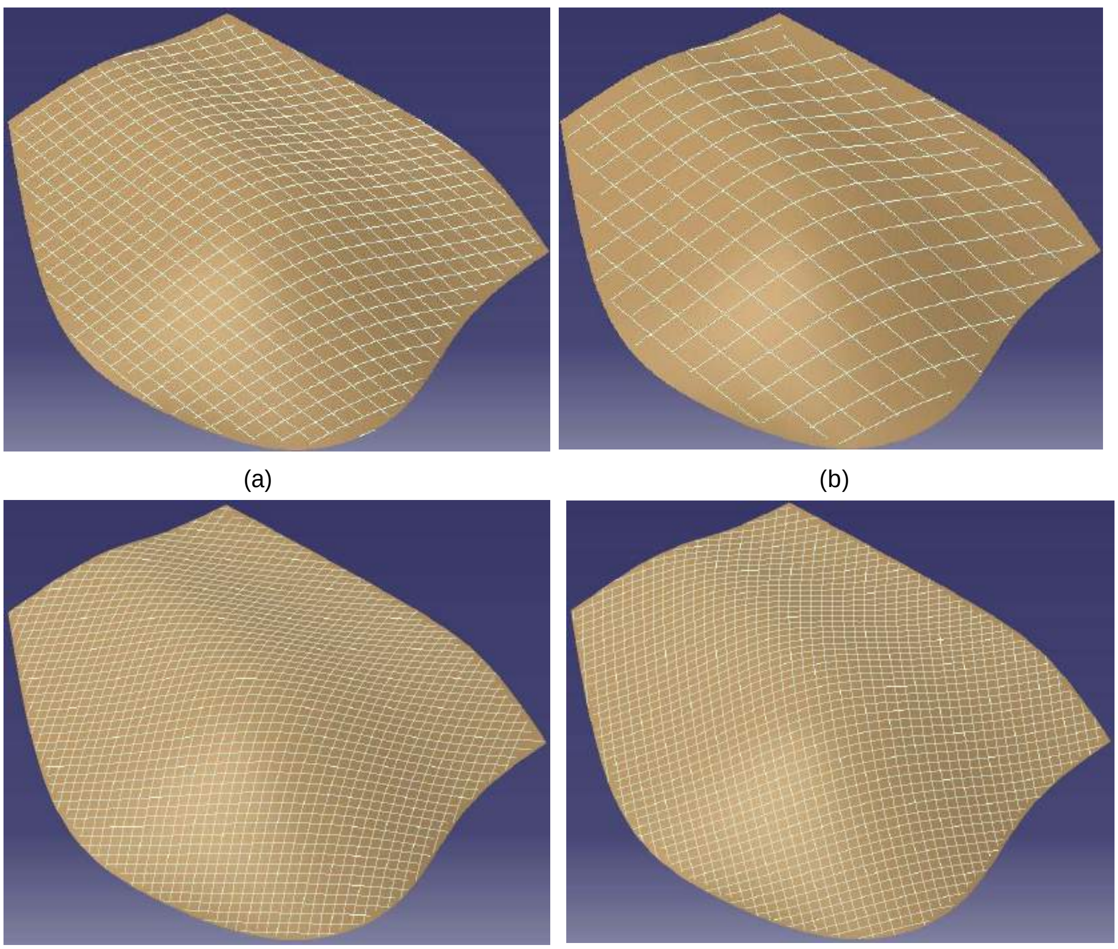
The approach described above was implemented as a functional module linked from a commercial CAD system. The architecture of our program is composed of four functional modules, as described below.

(i) The first module is implemented within a commercial CAD system, CATIAÊ. This module is used to display and interactively modify the designed surface. At this stage, the user is required to specify a tolerance value, τ, that will eventually be used in approximating the designed surface by approximating, developable quad panels. A simple API program written using the VBA programming language interface of CATIA creates the initial triangulated approximation, which is merely an STL mesh of the input surface. The tolerance used for the STL approximation is usually 0.1τ. The STL data is converted by the API into a standard mesh format, OFF and output into a text file.

(ii) The design interface is programmed using C++, and incorporates the 3D to 2D parameterization function from an open source library, CGAL [CGAL]. The module imports the OFF file, and uses the CGAL functions to create the 2D parameterization. It also creates the mapping between the 3D and 2D triangles, and the data structures required for point location. A simple graphical user interface is developed using OpenGL to allow the user to input the spiral pattern details, including the center point and the pitch for each spiral. The resulting 2D quad-mesh is displayed overlaid on the parameterized shape, and the user can interactively modify the mesh patterns in the 2D mapping. The module computes the 3D coordinates and outputs the mesh topology (i.e. the vertices and edges) to a simple text file.

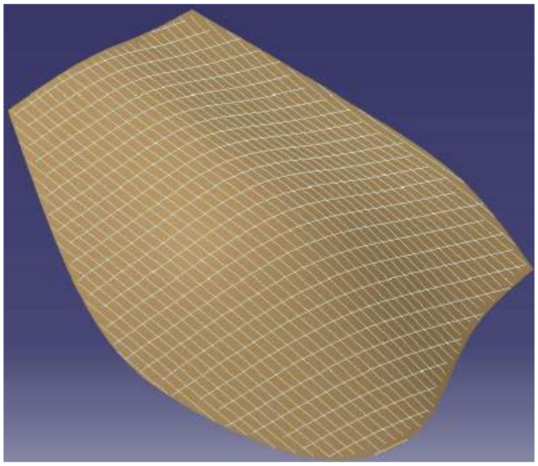
(iii) The third module is a simple API program written within CATIA that imports the mesh output by the designer, creates the corresponding vertices and line segments within CATIA, and displays the resulting quad-mesh to the user.
(iv) The final module is a mesh evaluation function. For each face of the quad-mesh, the corresponding edges are projected onto the original surface. A sample of 200 randomly distributed points is computed from the surface, and a simple single-cone surface is fitted onto the panel as described above. The module then displays the quality of the mesh in terms of the tolerance achieved by this quad-panel approximation.

Figures 7 show some examples of input surfaces, and the corresponding quad-meshes designed using different patterns. In each case, we also show the fitting statistics, indicating that our design tool indeed can provide a fairly good initial design for further optimization of the mesh (e.g. to improve the percent of panels that are within tolerance, or to reduce the number of unique classes in a K-set tiling).

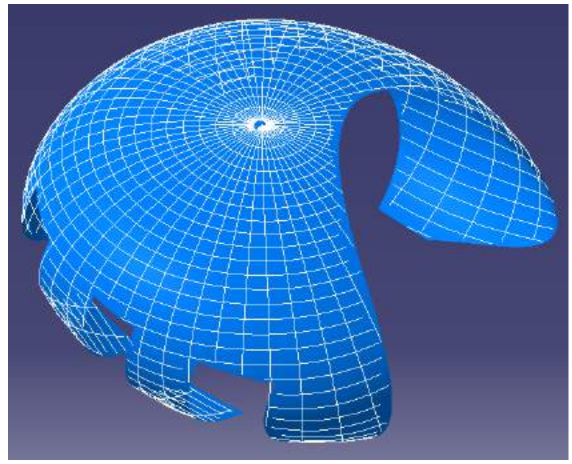


(a) (b)

(c) (d)



(e)



(f)

Fig. 7: Examples of generating quad mesh patterns using our method; (a)-(e) show the same surface meshed using variations of the spiral pitch and orientation; (f) shows an example of using the single spiral and radial lines approach.

We examine briefly the effectiveness of our approach. The time performance for the examples shown above is fairly good; the only stage that requires a run time of the order of a few minutes is the parameterization, which only needs to be performed once for each shape. All the other steps can be executed in a few seconds, and therefore we have a fairly convenient and almost real-time design tool available for testing a large variety of mesh patterns. Obviously, the aesthetics of a building can be changed dramatically by using different meshing patterns, so our simple operator can be a useful design tool. Since there is some uneven distortion due to the parameterization process, we also measure the curvature error along the spirals.

Finally, we evaluate how well our approach performs by using two quantitative measures. Firstly, we would expect that since the parameterization is not ideally isometric, the 3D realization of the curves have some distortion from the ideal, 2D shape beyond what is introduced by the draping. We measure this by measuring the angular error at each vertex. Figure 8 illustrates how this is computed: basically the two edges incident on the vertex and belonging to the same spiral are projected to the tangent plane of the designed surface, and the angle between them is compared to the angle between the corresponding edges in the planar mesh approximation at the corresponding point. In all the examples we have tested, this angular error was found to remain fairly low, with a maximum error being under 0.5 degrees.

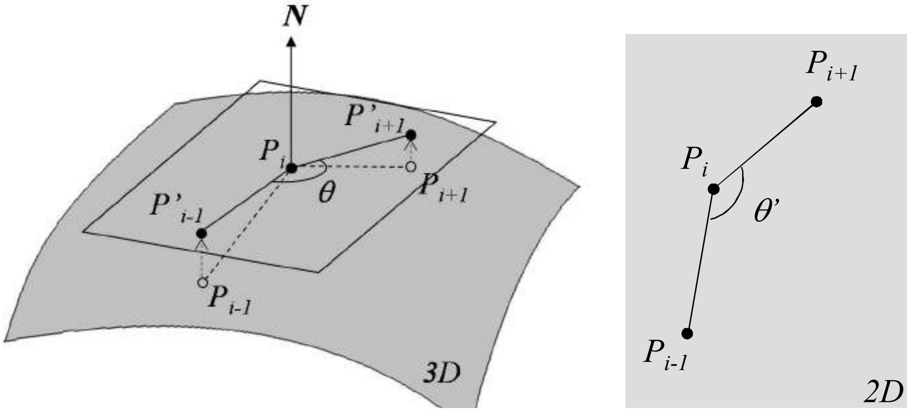
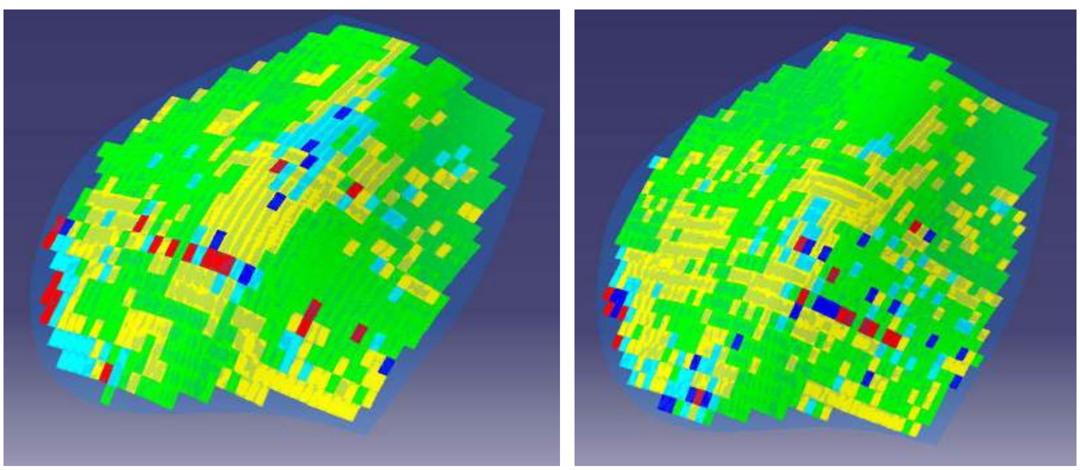
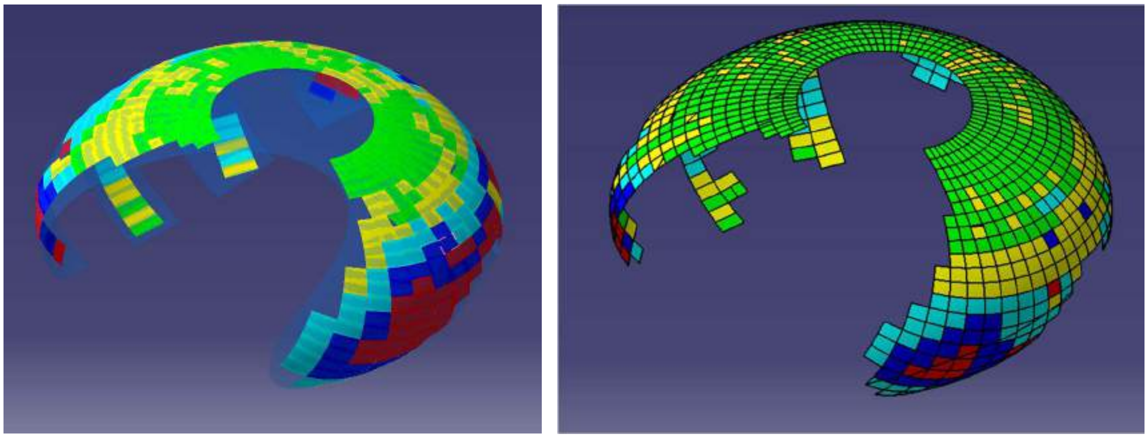


Fig. 8: Computing the angular error.

Secondly, we measure the manufacturability of the mesh pattern as described in the previous section. As mentioned earlier, the quad-mesh generated by our approach can be used as an input that can be further refined and optimized in order to improve the manufacturability. If the initial mesh is of very poor quality, the optimization will yield a solution that distorts the panel patterns too far from the initial design, or will require too many doubly curved (i.e. non-developable) panels. The figure below shows the results for several examples.



(a) (b)



(c) (d)

Fig. 9: Different quad panel designs, with panels approximated by cones. The different panel colors indicate different levels of maximum error between the approximating cone and the original surface.

Panel color (tolerance) →	Green (τ < 5mm)	Yellow (5< τ < 10)	Cyan (10< τ <15)	Blue (15< τ <20)	Red (τ > 20)
Example ↓					
(a)	445	228	78	8	19
(b)	684	315	101	30	37
(c)	194	147	112	65	63
(d)	495	240	102	40	22

Tab. 1: Entries in the cells indicate the number of panels that fall within the indicated tolerance level.

4 DISCUSSION

In this paper, we explore the problem of generating aesthetic quad meshes on architectural surfaces. We introduce the idea that classical 2D aesthetic curves can be used to generate interesting and practical shape patterns on surfaces. This expands the earlier ideas of using geodesic offset patterns or lines of principal curvature. We provide a convenient mechanism for designers to create these patterns quickly, and simple tools to measure the quality of these patterns (in terms of deviation from the pure curve aesthetic, as well as in terms of potential for optimization with a view to design for manufacture). The operators that we developed are easy to program, and efficient enough to allow almost real time testing of various design alternatives. Several real world examples of architectural surfaces were presented demonstrating the effectiveness of our technique.

5 ACKNOWLEDGEMENTS

The authors wish to thank Mr Billy Wong Chi Pan, Technical Director of GT Asia for data, consultation and advise. We also thank architects Ma Yansong, Yosuke Hayano and Qun Dand of MAD studio, Beijing for their kind permission to use the images of the beautiful Qunli Museum. The research in this paper is supported by UGC GRF grant # 614309.

REFERENCES

[1] Alliez, P.; Verdiere, E.; DeVillers, O.; Isenbure, M.: Isotropic Surface Remeshing, Intõl Conf on Shape Modeling and Applications, 49-58, 2003.

[2] Alliez, P.; Ucelli, G.; Gotsman, C.; Attene, M.: Recent advances in remeshing of surfaces, Shape Analysis and Structuring, Mathematics and Visualization. ed. L. de Floriani and M. Spagnuolo, Springer, Berlin, 2007.

[3] Azariadis, P.N.; Aspragathos, N.A.: Geodesic curvature preservation in surface flattening through constrained global optimization, Computer-Aided Design, 33, 581-591, July 2000, http://dx.doi.org/10.1016/S0010-4485(00)00102-0

[4] Bennis, C.; Vezien, J. M.; Iglesias, G.: Piecewise surface flattening for non-distorted texture mapping, Computer Graphics, 25(4), 1991, 237-246, http://dx.doi.org/10.1145/127719.122744

[5] Boissonnat, J.-D.; Cazals, F.: Coarse-to-fine Surface Simplification with Geometric Guarantees, Eurographics proceedings, 490ð499, 2001.

[6] Computational Geometry Algorithms Library (CGAL), www.cgal.org

[7] Eignesatz, M.; Kilian, M.; Schiftner, A.; Mitra, N.; Pottmann, H.; Pauly, M.: Paneling Architectural Freeform Surfaces, ACM Transactions on Graphics, 29(4), 2010.

[8] Faux, I.D.; Pratt, M.J.: Computational Geometry for Design and Manufacture, England, Ellis Horwood Limited, 1979, Ch.4, 116-120.

[9] Fu, C-W.; Lai, C-F.; He, Y.; Cohen-Or, D.: K-set tileable surfaces, ACM Transactions on Graphics, 29(4), 2010, doi: 10.1145/1778765.1778781

[10] Garland, M.; Heckbert, P.: Simplifying Surfaces with Color and Texture using Quadric Error Metrics, IEEE Visualization Conference Proceedings, 263ð269, 2008.

[11] George, P. L., Borouchaki, H., Eds.: Delaunay Triangulation and Meshing Application to Finite Elements, HERMES, Paris, 1998, ISBN-10: 2866016920

[12] Kirkpatrick, D. G.: Optimal search in planar subdivisions, SIAM Journal on Computing 12(1), 1983, 28ð35, http://dx.doi.org/10.1137/0212002

[13] Levy, B.; Petitjean, S.; Ray, N.; Maillot, J.: Least Squares Conformal Maps for Automatic Texture Atlas Generation, ACM Transactions on Graphics, Procs. of SIGGRAPH 02, 21(3), 362-371 , doi. 10.1145/566654.566590

[14] Manning, J.R.: Computerized pattern cutting, Computer-Aided Design, 12(1), 43-47, 1980, http://dx.doi.org/10.1016/0010-4485(80)90217-1

[15] McCartney, J.; Hinds, B.K.; Seow, B.L.: The flattening of triangulated surfaces incorporating darts and gussets, Computer-Aided Design, v.31, 249-260, 1999, http://dx.doi.org/10.1016/S0010-4485(99)00025-1

[16] Miura, K. T.: A general equation of aesthetic curves and its self-affinity, Computer-Aided Design and Applications, 3(1ð4), 2006, 457ð464.

[17] Pottmann, H.; Asperl, A.; Hofer, M.; Kilian, A.: Architectural Geometry, Bentley Institute Press, 2007, ISBN-10: 193449304X

[18] Pottmann, H.; Huang, Q.; Deng, B.; Schiftner, A.; Kilian, M.; Guibas, L.; Wallner, J.: Geodesic Patterns, ACM Transactions on Graphics, 29(4), 2010, doi: 10.1145/1778765.1778780

[19] Pottmann, H.; Liu, Y.; Wallner, J.; Bobenko, A.; Wang, W.: Geometry of multi-layer freeform structures for architecture, ACM Transactions, Procs of SIGGRAPH 2007, Graphics 26/3, doi: 10.1145/1276377.1276458

[20] Singh, M.; Schaefer, S.: Triangle Surfaces with Discrete Equivalence Classes, ACM Transactions on Graphics 29(4), 2010, doi: 10.1145/1778765.1778783

[21] Tam, A.; Joneja, A.; Tang, K.; Yao, Z.: A Surface Development Method with Application in Footwear CAD/CAM, CAD & Applications, 4(1-4), 2007.