Planar Quads in Free-Form Architecture Surfaces

Christian Dimitri, UPC BarcelonaTech

September 2018

This paper will cover the preprocessing techniques for planar quad meshes in architecture free-form surfaces. As a first step, we will covering the problems and objectives behind planar quads for construction, their benefits, their metrics as well as their goals, considering their constraints for a better optimization of the candidate *PQ mesh*. Secondly, we will explain the several preprocessing algorithms that generate a candidate *PQ mesh* ready for optimization. In addition to that, the output will be optimized according to it’s properties qualifying it to be PQ meshes. The last-mentioned are based on scientific papers, and were applied to concrete architectural projects. Combining chapter two and chapter three iteratively, we will be hitting the last chapter of this paper; generating subdivision method algorithm and a quad planarization in order to have a PQ mesh.

# Introduction

Planer Quad meshes have been nearly ubiquitous in architecture and construction. A large body of data structures and geometry processing algorithms based on them has been developed in the literature and adapted in construction of free-form surfaces. This type of re-meshing has many advantages especially the semi-regular ones, and significant progresses were made in quadrilateral mesh generation and processing during the last years. In this paper, we will study four algorithms behind planar quad meshes and their goals in order to fulfill the objectives. We will apply them on four input surfaces having different curvatures.

# Construction

In construction, planar quads should always be planar and their distribution on the mesh is preferably equidistant so that their size does not vary a lot. In the first section, the geometric properties of PQ meshes are introduced as well as their benefits over other ones. Therefore, the metrics and measures are split by type and explained graphically and mathematically.

## PQ Geometric Properties

Figure 1: The hinge is affected by the high Gaussian curvature on the surface of the *Yas Island Hotel By Zaha Hadid* (Leon and Shelden 2011). The difference between *PQ meshes* and *triangle meshes*.

Figure 1: The hinge is affected by the high Gaussian curvature on the surface of the *Yas Island Hotel By Zaha Hadid* (Leon and Shelden 2011). The difference between *PQ meshes* and *triangle meshes*.

A polygon face is planar if and only if its vertices *define a plane*. A triangle face is always planar, however a quadrangular face can be non-planar since the curvature plays a prominent role against the geometric property of a planar quad. Such constraint is a disadvantage for *PQ meshes* over triangular ones. Thus, if the warping height exceeds a certain limit while measuring it, the four vertices of each of the faces should be independent from its neighboring face’s vertex see fig. 1.

Knowing that two parallel vectors in space, enclosed at each point by two other vectors that are not necessarily parallel, form a planar face (Glymph et al. 2004). We Consider each row of faces is a *PQ strip*. *PQ meshes* are composed by vertices with a valence where along each vertex a curve of family A and a curve of family B intersect see (Liu et al. 2006). N-geons can appear with a valence so called singularities.

## Benefits

*Planar quad meshes* may be preferred over *triangular meshes* for construction reasons. In addition, planar quads have the same fabrication and assembling benefits as triangles. The advantages of planar quads meshes for construction over other meshes is that: *PQ meshes* have higher surface to edge ratio than triangles, thus, a lower mullion cost. *PQ meshes* consumes less energy during fabrication.

## Metrics-Measures(Quality)

To have *planar quads*, several measures are mentioned below. For a better quality, the mathematical measures and the conditions are classified by face and by mesh[fig. 2 and fig. 3}. In addition to that, some conditions are translated to *custom goals* that improve the quality of the mesh.

The measurements and conditions applied to the mesh itself are:

Figure 2: Table showing the main measures of PQ meshes see (Gokhale (2008), guide (2018) and Geometry and Generator (2016))

Figure 2: Table showing the main measures of PQ meshes see (Gokhale (2008), guide (2018) and Geometry and Generator (2016))

The measurements and conditions applied to the elements of the mesh are:

Figure 3: Table showing the main measures of PQ meshes see (Gokhale (2008), guide (2018) and Geometry and Generator (2016))

Figure 3: Table showing the main measures of PQ meshes see (Gokhale (2008), guide (2018) and Geometry and Generator (2016))

# Algorithmic Strategies

Figure 4: Algorithmic strategies diagrams

Figure 4: Algorithmic strategies diagrams

## Several Pre-Processing Techniques

Given four different meshes as inputs, several pre-processing techniques will be adapted in order to generate a *PQ mesh* with *planar faces*. The used techniques will depend on the surface type. Translation surfaces is an easy and fast algorithm to generate specific surfaces. However architecture free-form surfaces with high curvature require more complex algorithms to generate *PQ meshes* see fig. 4.

### Translation Surfaces

Translation surfaces are limited and easy to generate. The quads generated are the proof that it is generated through a set of parallel vectors that result in a planar face. In addition to that they are homogeneous because adding the same length vector as a constraint leads to have evenly spaces faces and reduce the variance. If the sectional curves are plane and the vectors are parallel having the same length the result will respond to the design principle of a translation surface. Assuming that one direction of the quad mesh net to be the sectional curve, two design principles can appear:

* The row of longitudinal sectional curves form parallel vectors.
* The row of lateral sectional curves form parallel vectors.

#### Row of sectional curves translated over a set of parallel vectors

The family of sectional curves translated over a set of parallel vectors is generated as follows: A random spatial curve called generatrix is translated against another random spatial curve called directrix as seen in fig. 5. A translation by equal length gives homogeneous results of the planar quads.

Figure 5: Geometric principle for translation surfaces and planarity measure fulfilled.

Figure 5: Geometric principle for translation surfaces and planarity measure fulfilled.

Several geometrical shapes have been developed in architecture during the history using translation surfaces. The elliptical paraboloid is the most familiar shape found in architecture. It is generated using the same principle, translating one parabolic curve against another.

Figure 6: Elliptical paraboloid

Figure 6: Elliptical paraboloid

In transition surfaces, some geometrical shapes admit boolean and joining operations, for example, the hyperbolic paraboloid is a type of translation surface that acknowledge such operations. By translating a parabolic curve over a hyperbolic the result is as seen in fig. 7

Figure 7: Translated hyperbolic paraboloid and joining possibilities

Figure 7: Translated hyperbolic paraboloid and joining possibilities

#### Scale-Translation Surfaces

Figure 8: Centric scale-translation expansion

Figure 8: Centric scale-translation expansion

Scale Translation surfaces are generated by adding a scale parameter to the output curves . After translating the sectional curve on each point equally distant from the directrix curve , the output curves can be scaled uniformly or non-uniformly controlled by the user. The central expansion of any curve gives a new curve having parallel edges. The center of expansion can be chosen randomly (Glymph et al. 2004). In this technique the centric expansion has been chosen. The resulting algorithm give planar quad meshes see fig. 8.

### *Conjugate networks*

Some curve networks are a robust and efficient method to extract *PQ meshes* (Liu et al. 2006). Such method admits a huge variety of free-form surfaces. The advantage of designing a conjugate direction field is that the user possesses total freedom in controlling the PQ mesh layout (Zadravec, Schiftner, and Wallner 2010). Thus, the panels are flat and discretize the principal curvature lines see (Liu et al. 2006).

Figure 9: Left: High twisting moment. Middle: Stiffening by triangulation. Right: Torsion free alignment.(Zadravec, Schiftner, and Wallner 2010)

Figure 9: Left: High twisting moment. Middle: Stiffening by triangulation. Right: Torsion free alignment.(Zadravec, Schiftner, and Wallner 2010)

In addition to that, it can admit free torsion node while aligning the curve networks with the stress and curvature field see fig. 9 for more information on statics sensitive layout (Schiftner and Balzer 2010).

#### The relation between PQ meshes and *conjugate networks*.

Conjugate curve networks are families of curves : For each unique curves of both family should appear. *Since are conjugate then they pre-define and get by integrating the vector field directions conjugate of family* .(Liu et al. 2006)

*Examples of Conjugate Curve Networks on Surfaces*:

* Suited for PQ meshes: (Liu et al. 2006)
  + *The network of principle curvature lines* see (fig. 10 left).
  + In a translation surface of the form a sectional curve is translated along another curve generatrix and vice versa see fig. 5.
* Less suited for PQ meshes:
  + *Epipolar curves*: The translation of a point along a line and the intersection of the planes through the points with that surface generates asymptotic curves that are not suited for such meshing see (fig. 10 center).
  + *Isophotic curves are conjugate to the system of the steepest descent curves respecting the z-axis* see (fig. 10 right).

Figure 10: Various conjugate networks

Figure 10: Various conjugate networks

#### What is a CDF on a *triangular mesh*.

On a smooth surface the tangent vectors are conjugate if and only if they are treated as two vectors in (Liu et al. 2011). The CDF is a tool for non-photorealistic rendering in order to visualize the surface topology. Therefore it is useful for surface re-meshing and alignment control. On a triangular face as seen in fig. 11 of a triangular mesh a CDF is:

Figure 11: A CDF on a Triangular face based on (Liu et al. 2011).

Figure 11: A CDF on a Triangular face based on (Liu et al. 2011).

* Four vectors {}
* Two scalar parameters {}:
  + oriented angle between and
  + oriented angle between and
  + They define the following: and

#### Generating quad-dominant meshes via conjugate direction field

When the input is a mesh and not a surface, it is preferable to imply isotropic re-meshing. In this case, the re-meshing tool mesh machine is used (Piker, n.d.). After re-meshing the given input meshes , the conjugate direction field is particularly generated using a custom plugin called (Greco 2018) developed by (Mueller and Adriaenssens 2018). The candidate *PQ mesh* is generated after applying the global parametrization using frame fields and tracing the streamlines.

##### Alignment with the curvature (Mueller and Adriaenssens 2018).

Figure 12: Surface tangency extracted on each of the meshes by computing the minimum and maximum *principle directions* in red and blue.

Figure 12: Surface tangency extracted on each of the meshes by computing the minimum and maximum *principle directions* in red and blue.

The quality of the mesh is always better when the panels are aligned with the curvature or the stress lines. Given four different meshes , the orthogonality is introduced for each of the meshes by computing the *principle directions* , and storing them in see fig. 12. This method has been used by (Liu et al. 2011).

##### Interpolating vector field with *N-PolyVector Field* (Mueller and Adriaenssens 2018).

Figure 13: Smoothed vector field using n-polyVector field algorithm

Figure 13: Smoothed vector field using n-polyVector field algorithm

In fig. 13, it is clear that the smoothed vector field and the parallel transport have been well generated. In order to find a smooth and aligned vector field on each of the four meshes , the algorithm is based on finding the trade-off between neighboring faces so that the parallel transport succeeds. It uses the novel method proposed by (Diamanti et al. 2014) called *N-Poly Vector Field*. While selecting a subset of points [P], the vector field is able to be generated smoothly and continuously. It finds the smoothest field by interpolating the two vectors parallelly. This method is different from the one used in (Liu et al. 2011) where a signed permutation method is used in order to find the correct relation between neighboring vectors.

##### *Conjugate direction field*

Figure 14: Conjugate field after smoothing previously the vector field .

Figure 14: Conjugate field after smoothing previously the vector field .

After smoothing the vector field in the previous step, a quad mesh can be computed after generating the conjugate networks (Liu et al. 2011). From the previous step a conjugate vector field is computed using an algorithm provided in (LibDirectional 2018) see fig. 14.

##### Global parametrization using frame fields

Figure 15: a) The first mesh , b) The second mesh . In green the boundary of the cutting path , the isolines, and the frame fields chosen at index .

Figure 15: a) The first mesh , b) The second mesh . In green the boundary of the cutting path , the isolines, and the frame fields chosen at index .

If the mesh possess negative curvature, the parametrization has to be done by patches, see fig. 15 otherwise the parametrization can be done on a single patch see fig. 16. The algorithm succeeds on all the meshes except for the last one where collisions appear.

Figure 16: c) The third mesh , d) The fourth mesh which is still in continuous research. In green the boundary of the cutting path , the isolines, and the frame fields chosen at index .

Figure 16: c) The third mesh , d) The fourth mesh which is still in continuous research. In green the boundary of the cutting path , the isolines, and the frame fields chosen at index .

The global parameterization using frame fields fig. 15 is computed at the index *i* to shape the new mesh in a 2D topology. For each of the given 3D meshes, align the topology with the given vector fields at index *i*. Therefore, such field can be easily manipulated by the user.

##### Tracing streamlines

The streamlines [] are traced on the 2D maps after integrating the Vector field then they are remapped on the 3D meshes . This method is based on the 4th order Runge-Kutta (Mueller and Adriaenssens 2018), see fig. 15 and fig. 16.

##### Extracting the candidate PQ Meshes

Figure 17: The resulting candidate PQ meshes have a semi regular valence. a) with one singularity . b) with one singularity . c) with one singularity .

Figure 17: The resulting candidate PQ meshes have a semi regular valence. a) with one singularity . b) with one singularity . c) with one singularity .

Given the conjugate filed and the streamlines , the meshes are generated by retrieving the faces with the same vertex valence fig. 17. Nevertheless, the resulting meshes are not totally planar and require a further optimization.

#### Generating quad-dominant meshes via principle curvature networks

This method is different from the previous one. The network of curves will be generated on each of the four meshes using a plugin called (Michalatos 2017), however the output is not sorted. Although, without a special library like (Jacobson, Panozzo, and others 2017) and (Ebke et al. 2013b), automatically extracting a robust-quad mesh (Ebke et al. 2013a) is very hard to achieve. This method is based on the on the mixed-integer quadrangulation by (Bommes, Zimmer, and Kobbelt 2009). Therefore, in this research an algorithm had to be developed in order to extract that candidate PQ mesh using conformal mapping.

##### Computing curvature networks

Figure 18: Curve network computed using (Michalatos 2017) on each of the input meshes .

Figure 18: Curve network computed using (Michalatos 2017) on each of the input meshes .

The *principle curvature networks* are generated automatically by reparametrizing the input meshes see fig. 18.

##### Global parametrization using conformal mapping

Figure 19: Conformal mapping parametrization on a unit plane and curvature color gradient. remapped and rebuilt on the 2D map.

Figure 19: Conformal mapping parametrization on a unit plane and curvature color gradient. remapped and rebuilt on the 2D map.

*The curve networks* previously computed, are reparametrized using conformal mapping. Then they are analyzed and rebuilt in order to close naked nodes and form meshes with a semi-regular valence.

##### Extracting the candidate PQ meshes

Figure 20: The resulting candidate PQ meshes have a semi regular valence. a) with four singularities . b) with two singularities . c) with one singularity and one singularity .

Figure 20: The resulting candidate PQ meshes have a semi regular valence. a) with four singularities . b) with two singularities . c) with one singularity and one singularity .

After mapping the *curve networks* and rebuilding the quad mesh on the unit plane, it is now possible to remap the meshes on input meshes see fig. 19 and fig. 20}.

### Conical meshes

Figure 21: Left: Offset property of a conical mesh. Right: *Railway station by B.Schneider* (Liu et al. 2006) a conical mesh as glass structure that *discretizes the principle curvature*.

Figure 21: Left: Offset property of a conical mesh. Right: *Railway station by B.Schneider* (Liu et al. 2006) a conical mesh as glass structure that *discretizes the principle curvature*.

Conical meshes are planar quad meshes which *discretize principle curvature lines*, possess an offset at a constant distance as well as planar connecting elements (Liu et al. 2006) see fig. 23. A conical mesh is conical if and only if all of its vertices are conical which means that the four faces meeting at the vertex are tangent to a common sphere (Liu et al. 2006) see fig. 22.

#### The angle criterion of a conical mesh

Figure 22: Faces configuration of a conical mesh (Liu et al. 2006).

Figure 22: Faces configuration of a conical mesh (Liu et al. 2006).

Assuming that the sum of the opposite angles on a vertex should always be equal to zero, see fig. 22, is a conical vertex if and only if the characterization of a conical mesh interior angles respond to this function:

#### The Offset Properties

Figure 23: Constant offset of a Conical Mesh see (Pottmann and Wallner 2008).

Figure 23: Constant offset of a Conical Mesh see (Pottmann and Wallner 2008).

Triangular meshes are missing the offset property at a constant distance, while conical meshes answer to this property (Liu et al. 2006). The faces of a conical mesh are incident to a common vertex and tangent to a cone with an axis . After offsetting, the axis remains the same and the faces are still tangent to the cone (Liu et al. 2006). The Languerre transformation (Liu et al. 2006) contains one of the instances for offsetting planes by a fixed distance along their normal vector. The Languerre transformation preserves the conical meshes at the offset.

#### The Normals

*The spherical image* is a fact where the vertices of a PQ mesh built on a unit sphere are converted to the normal vectors of . As the four faces incident to a common vertex tangent to the same cone, the normal vectors on each of the four faces share the same angle with the cone’s axis see fig. 22. Consequently, the spherical image of the principle curvature network returns an orthogonal curve network on a sphere (Liu et al. 2006).

#### Conical optimization

PQ meshes generated after computing the principle curve networks are well suited to be optimized using conical meshes conditions. In order to do that, the angles and normals are measured and visualized with a gradient color that varies in a range between the common meshes see fig. 24, fig. 25, fig. 26 and fig. 27.

##### Angles analysis

Figure 24: Before optimization: The sum of the opposite angles measured in radians for each vertex of the meshes generated via CDF.

Figure 24: Before optimization: The sum of the opposite angles measured in radians for each vertex of the meshes generated via CDF.

Figure 25: Before optimization: The sum of the opposite angles measured in radians for each vertex of the meshes generated via *principle curvature networks*.

Figure 25: Before optimization: The sum of the opposite angles measured in radians for each vertex of the meshes generated via *principle curvature networks*.

##### Normals analysis

Figure 26: Before optimization: The angles difference between the normals and the cones normal are measured in radians for each vertex of the meshes generated via *conformal mapping*.

Figure 26: Before optimization: The angles difference between the normals and the cones normal are measured in radians for each vertex of the meshes generated via *conformal mapping*.

The first mesh

Figure 27: Before optimization: The angles difference between the normals and the cones normal are measured in radians for each vertex of the meshes generated via *frame field*.

Figure 27: Before optimization: The angles difference between the normals and the cones normal are measured in radians for each vertex of the meshes generated via *frame field*.

##### Angles optimization

For each vertex on the mesh minimize the sum of the opposite angles equals to zero using (Piker 2010) solver.

Figure 28: After optimization: The sum of the opposite angles in radians of the meshes generated via *conformal mapping*.

Figure 28: After optimization: The sum of the opposite angles in radians of the meshes generated via *conformal mapping*.

Figure 29: After optimization: The sum of the opposite angles in radians of the meshes generated via *conformal mapping*.

Figure 29: After optimization: The sum of the opposite angles in radians of the meshes generated via *conformal mapping*.

##### Normals optimization

For each vertex on the mesh minimize the angles difference between the four adjacent faces normals and the cones normal should be equal to zero .

Figure 30: After optimization: The angles difference in radians of the meshes generated via frame field

Figure 30: After optimization: The angles difference in radians of the meshes generated via frame field

Figure 31: After optimization:The angles difference of the meshes generated via frame field.

Figure 31: After optimization:The angles difference of the meshes generated via frame field.

## Optimization (K2)goals

### Planarity

#### Analysis

Figure 32: Before optimization: Planarity measured in radians of the meshes generated via *frame field*.

Figure 32: Before optimization: Planarity measured in radians of the meshes generated via *frame field*.

Figure 33: Before optimization: Planarity measured in radians of the meshes generated via *conformal mapping*.

Figure 33: Before optimization: Planarity measured in radians of the meshes generated via *conformal mapping*.

#### Optimization

Figure 34: After optimization: Planarity measured in radians of the meshes generated via *frame field*.

Figure 34: After optimization: Planarity measured in radians of the meshes generated via *frame field*.

Figure 35: After optimization: Planarity measured in radians of the meshes generated via *frame field*.

Figure 35: After optimization: Planarity measured in radians of the meshes generated via *frame field*.

### Diagonals Aspect Ratio

#### Analysis

Figure 36: Before optimization: Diagonals aspect ratio measured in cm of the meshes generated via *frame field*.

Figure 36: Before optimization: Diagonals aspect ratio measured in cm of the meshes generated via *frame field*.

Figure 37: Before optimization: Diagonals aspect ratio measured in cm of the meshes generated via *conformal mapping*.

Figure 37: Before optimization: Diagonals aspect ratio measured in cm of the meshes generated via *conformal mapping*.

#### Optimization

Figure 38: After optimization: Diagonals aspect ratio measured in cm of the meshes generated via *frame field*.

Figure 38: After optimization: Diagonals aspect ratio measured in cm of the meshes generated via *frame field*.

Figure 39: After optimization: Diagonals aspect ratio measured in cm of the meshes generated via *conformal mapping*.

Figure 39: After optimization: Diagonals aspect ratio measured in cm of the meshes generated via *conformal mapping*.

### Element areas

#### Analysis

Figure 40: Before optimization: Area and the variance measured in sqcm of the meshes generated via *frame field*.

Figure 40: Before optimization: Area and the variance measured in sqcm of the meshes generated via *frame field*.

Figure 41: Before optimization: Area and the variance measured in sqcm of the meshes generated via *conformal mapping*.

Figure 41: Before optimization: Area and the variance measured in sqcm of the meshes generated via *conformal mapping*.

#### Optimization

Figure 42: Before optimization: Area and the variance measured in sqcm of the meshes generated via frame field\*.

Figure 42: Before optimization: Area and the variance measured in sqcm of the meshes generated via frame field\*.

Figure 43: Before optimization: Area and the variance measured in sqcm of the meshes generated via *conformal mapping*.

Figure 43: Before optimization: Area and the variance measured in sqcm of the meshes generated via *conformal mapping*.

### Warping height

#### Analysis

Figure 44: Before optimization: The height and the variance measured in cm of the meshes generated via *frame field*.

Figure 44: Before optimization: The height and the variance measured in cm of the meshes generated via *frame field*.

Figure 45: Before optimization: The height and the variance measured in cm of the meshes generated via *conformal mapping*.

Figure 45: Before optimization: The height and the variance measured in cm of the meshes generated via *conformal mapping*.

#### Optimization

Figure 46: After optimization: The height and the variance measured in cm of the meshes generated via *frame field*.

Figure 46: After optimization: The height and the variance measured in cm of the meshes generated via *frame field*.

Figure 47: After optimization: The height and the variance measured in cm of the meshes generated via *conformal mapping*.

Figure 47: After optimization: The height and the variance measured in cm of the meshes generated via *conformal mapping*.

## Subdivision Strategy (Starting with a Coarse Quad-Dominant mesh)

### Subdivision Strategy principles

Figure 48: Left: Singularities with negative indices. Right: Singularities with positive indices.

Figure 48: Left: Singularities with negative indices. Right: Singularities with positive indices.

A coarse mesh that approximates the topology of a input surface can be subdivided using the catmull-clark algorithm (Piacentino 2010). For PQ meshes, the valence of the each vertex should be four, vertices with a valence more then four are considered as singularities. After applying the subdivision on the coarse mesh, singularities with negative indices take a negative curvature and singularities with a positive indices take a positive curvature see fig. 48.

### Curvature and singularities analysis

Figure 49: Singularities with indices - and are placed accordingly to the curvature.

Figure 49: Singularities with indices - and are placed accordingly to the curvature.

On the given input meshes, the curvature is analyzed and the singularities are placed by index see fig. 49.

### Generating the coarse mesh

Figure 50: The coarse meshes are generated thru isocurves

Figure 50: The coarse meshes are generated thru isocurves

Subsequently to the previous step, a 2D map by patches is generated. Such a method can help out predicting the pre-networking between singularities and avoiding unexpected ones. Therefore it is now possible to generate the coarse mesh following the 2D map see fig. 50.

### Catmull-clark subdivision and pull to mesh

Figure 51: Left: Subdivided mesh using catmull-clark algorithm and singularities in color. Right: Pulling the subdivided mesh to the input surface.

Figure 51: Left: Subdivided mesh using catmull-clark algorithm and singularities in color. Right: Pulling the subdivided mesh to the input surface.

The catmull-clark algorithm is applied to the coarse meshes. Using kangaroo2 (Piker 2010) the coarse mesh is pulled by constraining the latter’s points on the input meshes. The returning outputs are the candidate *PQ meshes* that need iterative optimization.fig. 51.

### Optimization

The optimizzation happens by constraining the faces under the planarity goal using (Piker 2010) solver.

#### Planarity

##### Analysis

Figure 52: Before optimization: Planarity measured in radians of the meshes generated via subdivision technique.

Figure 52: Before optimization: Planarity measured in radians of the meshes generated via subdivision technique.

##### Optimization

Figure 53: After optimization: Planarity measured in radians of the meshes generated via subdivision technique.

Figure 53: After optimization: Planarity measured in radians of the meshes generated via subdivision technique.

#### Diagonals aspect ratio

#### Analysis

Figure 54: Before optimization: Diagonals aspect ratio measured in cm of the meshes generated via subdivision technique.

Figure 54: Before optimization: Diagonals aspect ratio measured in cm of the meshes generated via subdivision technique.

#### Optimization

Figure 55: After optimization: Diagonals aspect ratio measured in cm of the meshes generated via subdivision technique.

Figure 55: After optimization: Diagonals aspect ratio measured in cm of the meshes generated via subdivision technique.

#### Element areas

#### Analysis

Figure 56: Before optimization: Area and the variance measured in sqcm of the meshes generated via subdivision technique.

Figure 56: Before optimization: Area and the variance measured in sqcm of the meshes generated via subdivision technique.

#### Optimization

Figure 57: After optimization: Area and the variance measured in sqcm of the meshes generated via subdivision technique.

Figure 57: After optimization: Area and the variance measured in sqcm of the meshes generated via subdivision technique.

#### Warping height

#### Analysis

Figure 58: Before optimization: The height and the variance measured in cm of the meshes generated via subdivision technique.

Figure 58: Before optimization: The height and the variance measured in cm of the meshes generated via subdivision technique.

#### Optimization

Figure 59: After optimization: The height and the variance measured in cm of the meshes generated via subdivision technique.

Figure 59: After optimization: The height and the variance measured in cm of the meshes generated via subdivision technique.

## Comparison & Synthesis

Figure 60: Graph showing the difference between the elements areas and the number of panels for each of the first mesh generated by the three different techniques

Figure 60: Graph showing the difference between the elements areas and the number of panels for each of the first mesh generated by the three different techniques

Figure 61: Graph showing the difference between the elements areas and the number of panels for each of the first mesh generated by the three different techniques

Figure 61: Graph showing the difference between the elements areas and the number of panels for each of the first mesh generated by the three different techniques

Figure 62: Graph showing the difference between the elements areas and the number of panels for each of the first mesh generated by the three different techniques

Figure 62: Graph showing the difference between the elements areas and the number of panels for each of the first mesh generated by the three different techniques

Figure 63: Graph showing the difference between the elements areas and the number of panels for each of the first mesh generated by the three different techniques

Figure 63: Graph showing the difference between the elements areas and the number of panels for each of the first mesh generated by the three different techniques

# Conclusion

*PQ meshes* must show different results from the mere geometry since the planarity of faces should obey the goals in order to fulfil the basis properties of the *planar quad meshes* (Zadravec, Schiftner, and Wallner 2010). They are very hard to deal with when the input surface is a free-form. However some algorithms have shown the differences between them and there results. Having a conjugate direction field as a tool to control the mesh layout is very useful. Thus generating PQ meshes from curve network is robust as well. The two different methods are almost planar after generation since they are extracted from the principal directions. The conical optimization has proved its robustness over planar quad meshes. By optimizing and combining the methods the last one was to generate planar quads by subdividing a coarse mesh and then optimize it. The boundary condition has been neglected.

# Further work

For further research the boundary will be taken in consideration while generating the PQ meshes. The fourth mesh that failed in the frame field algorithm has to be developped accordingly

Bommes, David, Henrik Zimmer, and Leif Kobbelt. 2009. “Mixed-Integer Quadrangulation.” In *ACM Transactions on Graphics (Tog)*, 28:77. 3. ACM.

Diamanti, Olga, Amir Vaxman, Daniele Panozzo, and Olga Sorkine-Hornung. 2014. “Designing N-Polyvector Fields with Complex Polynomials.” In *Computer Graphics Forum*, 33:1–11. 5. Wiley Online Library.

Ebke, Hans-Christian, David Bommes, Marcel Campen, and Leif Kobbelt. 2013a. “QEx: Robust Quad Mesh Extraction.” *ACM Transactions on Graphics (TOG)* 32 (6). ACM:168.

———. 2013b. “QEx: Robust Quad Mesh Extraction.” *ACM Trans. Graph.* 32 (6). New York, NY, USA: ACM:168:1–168:10. <https://doi.org/10.1145/2508363.2508372>.

Geometry, CUBIT Sandia, and Mesh Generator. 2016. “Metrics for Quadrilateral Elements.” 2016. <https://cubit.sandia.gov/public/15.2/help_manual/WebHelp/mesh_generation/mesh_quality_assessment/quadrilateral_metrics.htm>.

Glymph, James, Dennis Shelden, Cristiano Ceccato, Judith Mussel, and Hans Schober. 2004. “A Parametric Strategy for Free-Form Glass Structures Using Quadrilateral Planar Facets.” *Automation in Construction* 13 (2). Elsevier:187–202.

Gokhale, Nitin S. 2008. *Practical Finite Element Analysis*. Finite to infinite.

Greco, Lorenzo. 2018. “Plugin for Advanced Mesh Editing, Remeshing, Quads from Fields, Mesh Analysis.” 2018. <https://www.food4rhino.com/app/capybara>.

guide, Autodesk Nastran Reference. 2018. “Geometry Check Options.” 2018. <https://knowledge.autodesk.com/support/nastran/learn-explore/caas/CloudHelp/cloudhelp/2019/ENU/NSTRN-Reference/files/GUID-69125711-AB89-4A46-8FA9-B7DB2C856A13-htm.html>.

Jacobson, Alec, Daniele Panozzo, and others. 2017. “libigl: A Simple C++ Geometry Processing Library.”

Leon, Alexander Peña de, and Dennis Shelden. 2011. “A Technique for the Conditional Detailing of Grid-Shell Structures: Using Cellular Automata’s as Decision Making Engines in Large Parametric Model Assemblies.” In *Computational Design Modelling*, 267–73. Springer.

LibDirectional. 2018. “Matlab Library for Directional Statistics and Directional Estimation.” 2018. <https://github.com/libDirectional>.

Liu, Yang, Helmut Pottmann, Johannes Wallner, Yong-Liang Yang, and Wenping Wang. 2006. “Geometric Modeling with Conical Meshes and Developable Surfaces.” In *ACM Transactions on Graphics (Tog)*, 25:681–89. 3. ACM.

Liu, Yang, Weiwei Xu, Jun Wang, Lifeng Zhu, Baining Guo, Falai Chen, and Guoping Wang. 2011. “General Planar Quadrilateral Mesh Design Using Conjugate Direction Field.” In *ACM Transactions on Graphics (Tog)*, 30:140. 6. ACM.

Michalatos, Panagiotis. 2017. “Millipede for Structural Analysis and Optimization.” 2017. <http://www.sawapan.eu/>.

Mueller, Caitlin, and Sigrid Adriaenssens. 2018. “Optimized Quad Gridshell from Stress Field and Curvature Field.”

Piacentino, Giulio. 2010. “Weaverbird – Topological Mesh Editor.” 2010. <http://www.giuliopiacentino.com/weaverbird/>.

Piker, Daniel. 2010. “Kangaroo Is an Interactive Physics/Constraint Solver and Grasshopper Plugin for Designers.” 2010. <http://kangaroo3d.com/>.

———. n.d. “MeshMachine for Remeshing.” <https://www.grasshopper3d.com/profiles/blogs/meshmachine-update?id=2985220%3ABlogPost%3A1085830&page=2>.

Pottmann, Helmut, and Johannes Wallner. 2008. “The Focal Geometry of Circular and Conical Meshes.” *Adv. Comp. Math* 29:249–68.

Schiftner, Alexander, and Jonathan Balzer. 2010. “Statics-Sensitive Layout of Planar Quadrilateral Meshes.” *Advances in Architectural Geometry 2010*. Springer, 221–36.

Zadravec, Mirko, Alexander Schiftner, and Johannes Wallner. 2010. “Designing Quad-Dominant Meshes with Planar Faces.” In *Computer Graphics Forum*, 29:1671–9. 5. Wiley Online Library.