DESIGN OF MESO_SCALE CELLULAR STRUCTURE FOR RAPID MANUFACTURING

A Thesis
Presented to
The Academic Faculty

by

Sarah Engelbrecht

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in Mechanical Engineering

Georgia Institute of Technology

May, 2009

DESIGN OF MESO_SCALE CELLULAR STRUCTURE FOR RAPID MANUFACTURING

Approved by:

Dr. David Rosen, Advisor
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Suresh Sitaraman
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Seung-Kyum Choi
School of Mechanical Engineering
Georgia Institute of Technology

Date Approved: March 20, 2009

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. David Rosen, and the members of my committee, Dr. Seung-Kyum Choi and Dr. Suresh Sitaraman, for taking the time to read and advise on my thesis and for offering their insight and expertise.

I would also like to thank the faculty, staff, and students of the Systems Realization Laboratory at the Georgia Institute of Technology, especially Greg Graf and Jane Chu, for their support, collaboration, and friendship.

Finally, I would like to thank my family for their support.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS……...……iii

LIST OF TABLES……...……..vi

LIST OF FIGURES……...……...vii

LIST OF SYMBOLS AND ABBREVIATIONS…….……....x

SUMMARY……xi

1 Introduction 1
1.1 Background 1
1.2 Motivation 3
1.3 Goals 5
1.4 Organization of Thesis 7

2 Literature Review and Research Gap 9
2.1 Literature Review 9
2.1.1 Flattening 9
2.1.2 Automated Mesh Creation 11
2.1.3 Offsetting 17
2.1.4 Cellular Structure Design 19
2.2 Research Gap 20
2.3 Summary 21

3 Algorithm Description 23
3.1 STL Processing 26
3.2 Sampling 28
3.2.1 Line Sampling 28
3.2.2 Face Sampling 29
3.3 Partitioning 32
3.4 Flattening 33
3.5 Mesh Generation 40
3.5.1 Inside Points 42
3.5.2 Boundary Points 46
3.5.3 Corners 47
3.6 Boundary Matching 49
3.7 Offsetting 54
3.8 Hexahedral Element Formation 61
3.9 Truss Definition 64
3.10 Detailed Example 66
3.11 Summary 87

4 Examples 89
4.1 Evaluation Criteria 89
4.2 Example Problems 94
4.2.1 Two planar surfaces meeting at an angle 94
4.2.2 Cylindrical surface 101
4.2.3 Flat plane meeting a concave surface 107
4.2.4 Flat plane meeting a convex surface 114
4.2.5 Spherical surface 121
4.2.6 Conical surface with varying curvature 127
4.2.7 Surface with a saddle point with one partition 133
4.2.8 Surface with a saddle point with two partitions 137
4.3 Sensitivity 141
4.3.1 Mesh size sensitivity 141
4.3.2 Mesh with varying sampling rate 144
4.4 Summary 148

5 Conclusion 151
5.1 Conclusions 152
5.1.1 Conclusion One: Surface Mesh 152
5.1.2 Conclusion Two: Offsetting 154
5.1.3 Conclusion 3: Cellular material 154
5.2 Contributions 155
5.2.1 System of operations 155
5.2.2 Efficient offsetting 156
5.3 Future work 156
5.3.1 Element quality 156
5.3.2 Partition orientation 157
5.3.3 Cellular material library 158
5.3.4 FEM interface 159
5.4 Closure 159

REFERENCES 160

LIST OF TABLES

Table 4-1: Example descriptions 94

Table 4-2: Surface data for example problem with two flat surfaces 95

Table 4-3: Algorithm inputs for example with two flat surfaces 96

Table 4-4: Mesh quality for input of two flat surfaces 99

Table 4-5: Offset quality for two flat surfaces 100

Table 4-6: Algorithm data for cylindrical surface example 102

Table 4-7: Cylindrical surface mesh quality 105

Table 4-8: Cylindrical surface offset quality 106

Table 4-9: Surface data for flat/concave surface example 108

Table 4-10: Algorithm input data for flat/concave surface example 108

Table 4-11: Flat/concave surface mesh quality 112

Table 4-12: Flat/concave surface offset quality 113

Table 4-13: Surface data for flat/convex surface example 115

Table 4-14: Algorithm input data for flat/convex surface example 115

Table 4-15: Flat/convex surface quality mesh 119

Table 4-16: Flat/convex surface offset quality 120

Table 4-17: Algorithm data for the hemispherical surface example 121

Table 4-18: Hemispherical surface mesh quality 124

Table 4-19: Hemispherical surface offset quality 127

Table 4-20: Algorithm data for conical surface example 128

Table 4-21: Conical surface mesh quality 130

Table 4-22: Conical surface offset quality 133

Table 4-23: Algorithm data for saddle surface example 134

Table 4-24: Saddle surface mesh quality 135

Table 4-25: Saddle surface offset quality 137

Table 4-26: Algorithm data for saddle surface example, two partitions 137

Table 4-27: Comparison of mesh quality data for a saddle surface 139

Table 4-28: Offset quality for saddle surface with two partitions 141

Table 4-29: Mesh quality for varying mesh size 144

Table 4-30: Quality for mesh with varying sampling criteria 147

Table 4-31: Effect of sampling size on quality 148

Table 5-1: Surface mesh quality summary 153

Table 5-2: Quality summary for high sampling rate 153

Table 5-3: Offset data summary 154

LIST OF FIGURES

Figure 1.1: Uniform and custom lattice examples 2

Figure 1.2: Cellular structure process 6

Figure 1.3: Thesis organization 7

Figure 2.1: Automated quadrilateral mesh using a Voronoi tesselation 12

Figure 2.2: Automated quadrilateral mesh in polygons 13

Figure 2.3: Automated quadrilateral mesh in loops 13

Figure 2.4: Automated hexahedral mesh 17

Figure 2.5: Offset example 18

Figure 3.1: Algorithm map 25

Figure 3.2: STL file format 26

Figure 3.3: Line sampling 29

Figure 3.4: Face sampling 31

Figure 3.5: Partition in original orientation (left) and reoriented for flattening (right) 34

Figure 3.6: Flattening the first side on the first face 36

Figure 3.7: Flattening the third vertex of a face 38

Figure 3.8: Averaging two locations for a flattened point 39

Figure 3.9: Flattened partition 40

Figure 3.10: Grid superimposed on a flattened partition 41

Figure 3.11: Using cross products to determine if a point is within a triangle 43

Figure 3.12: Locating a point within a flattened triangular face 44

Figure 3.13: Matching a point to an existing sampled point 45

Figure 3.14: Inside Point matrix example 46
Figure 3.15: Matching a boundary point to an existing sampled point 47
Figure 3.16: Adding corner points 48
Figure 3.17: Matching boundaries with equal numbers of elements 50
Figure 3.18: Matching boundaries with unequal numbers of elements 51
Figure 3.19: Element quality checks 53
Figure 3.20: Inside and outside offset of a square 54
Figure 3.21: Offsetting face sampled points 55
Figure 3.22: Concave and convex lines 56
Figure 3.23: Offsetting line sampled points 57
Figure 3.24: Offsetting vertex sampled points 58
Figure 3.25: More efficient offsetting 60
Figure 3.26: Quadrilateral element projected onto the offset surface 61
Figure 3.27: Matching a quadrilateral base to an offset point cloud 63
Figure 3.28: Cellular structure definition 65
Figure 3.29: Truss structure formulation 65
Figure 3.30: Detailed example - cylindrical input surface 66
Figure 3.31: Detailed example - sampling example 67
Figure 3.32: Detailed example - cylindrical input surface in three partitions 68
Figure 3.33: Detailed example - first partition 69
Figure 3.34: Detailed example - second partition 70
Figure 3.35: Detailed example - third partition 71
Figure 3.36: Detailed example - partition three flattened with grid overlaid 73

Figure 3.37: Detailed example - Inside Point Matrix 73
Figure 3.38: Detailed example - Boundary Point Matrix 74
Figure 3.39: Detailed example - combined boundary and inside points..... 75
Figure 3.40: Detailed example - element formation examples 76
Figure 3.41: Detailed example - initial mesh on the third partition..... 77
Figure 3.42: Detailed example - initial mesh..... 78
Figure 3.43: Detailed example - mismatched partition boundary..... 79
Figure 3.44: Detailed example - boundary match and quality check for partition 1/2 80
Figure 3.45: Detailed example - boundary match and quality check for partition 1/3 82
Figure 3.46: Detailed example - secondary quality check..... 84
Figure 3.47: Detailed example - calculated offset mesh..... 85
Figure 3.48: Detailed example - surface and offset meshes plotted together 86
Figure 3.49: Detailed example - cellular material primitive..... 86
Figure 3.50: Detailed example - cellular material 87
Figure 4.1: Element quality measurements..... 90
Figure 4.2: Aspect ratio measurements..... 91
Figure 4.3: Offset distance measurement..... 93
Figure 4.4: Two flat surfaces meeting at an angle..... 95
Figure 4.5: Input of two flat surfaces divided into two partitions 96
Figure 4.6: Input of two flat surfaces with mesh 98
Figure 4.7: Input of two flat surfaces with truss cellular structure 100
Figure 4.8: Half regular cylinder 101
Figure 4.9: Cylindrical input surface divided into three partitions..... 102

Figure 4.10: Partition division examples 103
Figure 4.11: Cylindrical input surface with mesh..... 104
Figure 4.12: Cylindrical input surface with cellular structure 106
Figure 4.13: Flat surface meeting a concave surface at an angle..... 107
Figure 4.14: Flat/concave input surface divided into four partitions..... 109
Figure 4.15: Flat/concave input surface with mesh 111
Figure 4.16: Flat/concave input surface with cellular structure..... 113
Figure 4.17: Flat surface meeting a convex surface at an angle 114
Figure 4.18: Flat/convex input surface divided into four partitions 116
Figure 4.19: Flat/convex input surface with mesh..... 118
Figure 4.20: Flat/convex input surface with cellular structure 120
Figure 4.21: Spherical hemisphere 121
Figure 4.22: Hemispherical input surface divided into five partitions 122
Figure 4.23: Hemispherical input surface with mesh 124
Figure 4.24: Hemispherical input surface with cellular structure..... 126
Figure 4.25: Conical surface with varying curvature..... 128
Figure 4.26: Conical input surface divided into four partitions..... 129
Figure 4.27: Conical input surface with mesh 130
Figure 4.28: Conical input surface with cellular structure..... 132
Figure 4.29: Surface with a saddle point 133
Figure 4.30: Input saddle surface with mesh 135
Figure 4.31: Input saddle surface with cellular material..... 136
Figure 4.32: Input saddle surface divided into two partitions 138

Figure 4.33: Input saddle surface with two partitions with mesh 139

Figure 4.34: Input saddle surface with two partitions with cellular material 140

Figure 4.35: Surface mesh with increasing size..... 143

Figure 4.36: Surface mesh with decreasing sampling rate 146

Figure 4.37: Summary list of example problems..... 149

Figure 5.1: Possible mesh correction..... 157

Figure 5.2: Rotated partition example 158

LIST OF SYMBOLS AND ABBREVIATIONS

FEM…… Finite element model

α……...skew angle

AR……....aspect ratio

h……..…quadrilateral mid-side lengths

Summary

Customized cellular material is a relatively new area made possible by advancements in rapid manufacturing technologies. Rapid manufacturing is ideal for the production of customized cellular structure, especially on the meso scale, due to the size and complexity of the design. The means to produce this type of structure now exist, but the processes to design the structure are not well developed. The manual design of customized cellular material is not realistic due to the large number of features. Currently there are few tools available that aid in the design of this type of material. In this thesis, an automated tool to design customized cellular structure is presented.

The tool developed to design cellular material uses a surface as input. The surface is broken into two-dimensional finite elements, with the goal of creating square-shaped elements. The input surface and finite element mesh are then offset a uniform distance from the input surface to create a volume. The matching two-dimensional meshes on the input and offset surfaces form three-dimensional elements that are filled with cellular material primitives. The result is a continuous cellular material design that covers the input surface.

The tool is tested on a number of example surfaces. The results show that cellular material can effectively be produced for a variety of different surface types. Mesh on the input surface is generally good quality, especially in areas away from boundaries. The offset function is very accurate, and work done to make offsetting more efficient was successful. Finally, a uniform layer of cellular structure is created with no continuity or orientation errors, which is the desired result.

1 Introduction

Additive manufacturing, often referred to as rapid prototyping, is capable of producing a vast array of structures, some of which would be impossible or infeasible using more traditional manufacturing methods. One application is the use of integral meso-scale cellular structures, such as truss elements, to modify the performance of structural elements. Meso-scale cellular structures offer lightweight stiffness solutions, as with traditional materials such as “honeycomb” core. The first step in producing this type of structure is to design an initial configuration of cells and populate them with a specific type of structure.

The focus of this research is to develop a method to generate one or more layers of meso-scale cellular structure for a given surface. The goal is to input an STL representation of the surface and to produce an array of cellular structures to cover the area, which can further be used in the design of integral structural elements. From the input surface, an offset surface will be calculated, and both will be parameterized by a grid. Hexahedral mesh will be generated between the surfaces, which can be populated with a variety of cellular structures.

1.1 Background

Additive manufacturing is a relatively new technology which is changing the boundaries of design. Improvements to the speed and accuracy of additive techniques have opened an almost limitless window of design opportunities. On paper, almost any

shape or configuration is possible, and the only constraint is the imagination of the designer. In reality, however, there is a barrier between the designer's mind and the finished product. Transferring an idea from paper to the computer interface for production can create considerable obstacles.

This thesis focuses on the area of customized meso-scale cellular structure, in particular trusses. Figure 1.1 shows an example of the difference between a uniform truss, which contains struts that are of constant size and regular repetition, and a custom mesh, which fits to a specified surface and has struts of varying size and spacing. The uniform truss can be easily designed and mass produced. The custom truss is more challenging.

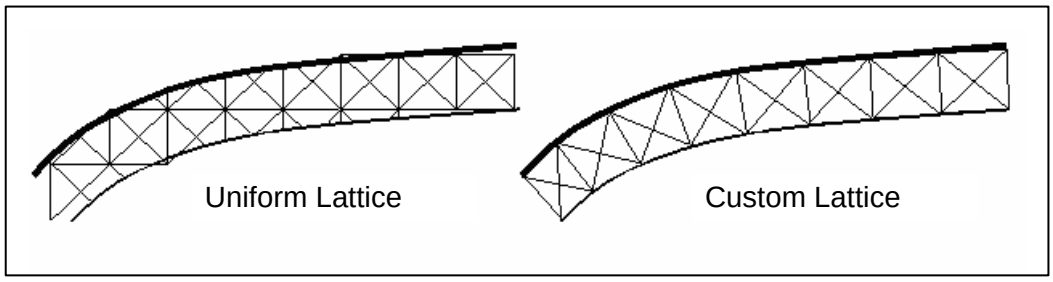


Figure 1.1: Uniform and custom lattice examples

For most applications, the design of custom cellular structure requires a large number of small unit cells to cover a surface or fill a volume. To design this type of structure manually would take considerable time and skill, even for relatively simple surfaces. The meso-structure could comprise thousands of individual elements that would each have to be placed independently, making even modest applications infeasible. The purpose of this thesis is to develop a tool to aid in the design of meso-scale cellular structure.

1.2 Motivation

Additive manufacturing allows the creation of a wide range of structure that cannot be produced with traditional manufacturing techniques. One example is customized meso-scale cellular structure. The size, complexity, and non-uniformity possible with this type of structure make producing it extremely complicated. Additive manufacturing offers a unique solution to the problem of customized structure because it does not rely on hard patterns, machinery configurations, casts, etc. The remaining challenge is to design the structure itself.

Often, a good application of meso-scale cellular structure is as a stiffening agent for a surface. If one were to design cellular structure to stiffen a particular surface, the only known information would be the surface itself. To create cellular structure, a volume is needed. It may be feasible to calculate an acceptable volume for the given input surface, though it may be hard to evaluate the quality of the volume. For some input surfaces, it would be very difficult to create a proper volume to fit with cellular structure.

Once the volume is known, it must be divided into cells. This could not be done manually in most situations. If the cellular material can be fit to tetrahedral-shaped elements and if uniform sizing is not crucial and if the orientation of the cellular structure is not important, many finite element modeling tools are available that could divide the volume effectively. If cubic elements and uniform element size are desired, it is much more difficult to use available finite element modelers to divide the volume. If the surface is very simplified or very regular, commercial finite element modeling software

could possibly be used. The types of surfaces that could be used would be severely limited, however.

Once the surface is divided into cells, or elements, the cellular structure must be inserted. Again, this is not something that could realistically be done manually for most cases. If the orientation of the cellular material is not important and all features of the cellular structure are equal in size, it may be feasible to partially automate the process of inserting cellular material into defined cells created by a finite element modeling tool. If a specific orientation is desired, the finite element output may have to be sorted before cellular structure could be inserted, which may take considerable time and produces additional book-keeping requirements. Cellular structure created this way also limits opportunities for customization because the input is not standardized or inherently organized.

So, for some types of surfaces, volumes that can be filled with cellular structure are relatively easy to calculate. For some of those volumes, it may be possible to get a set of cells/elements that can be fit with cellular structure. For some of those sets of cells, a cellular structure definition can effectively be inserted. Between each of those steps, information may have to be transferred from one software package to another, which is time consuming and may lead to errors. There are no software packages for inserting cellular structure into finite element mesh, so this would have to be done in some other manner and would likely have to be highly tailored to the mesh for specialized orientation arrangements. All in all, the current process to produce cellular structure for a given surface is extremely limited, complicated, and time consuming. There is a large window of opportunity for improvement.

1.3 Goals

The general goal of this thesis is to develop a tool to automate the design process for meso-scale cellular structure for an input surface. Currently, design of cellular structure is very limiting and time consuming. It requires a number of different tools and specialized skills. A tool that combines the necessary processes, is efficient, and expands the range of surfaces that can be fit is needed to make it feasible to implement cellular structure more often.

There are three specific goals for this tool. The first goal is to parameterize the input surface into square elements of a given size. Square elements are chosen because they are more difficult to calculate using commercial finite element modeling tools, and they can be broken into tetrahedral elements or other shapes if desired. Many cellular material configurations are designed to fit cubic elements, as well, because of the symmetry. A uniform element size is desired in order to create cellular materials with good stiffness, strength, and weight properties. Elements that are smaller than requested will produce cellular structure that is more dense, which negates the benefits of cellular structure. Elements that are larger than requested will produce cellular structure that is less dense, but that may not be strong or stiff enough for the intended application. The square elements on the input surface form the bases of the cubic elements to be filled with cellular structure.

The second goal is to create the volume to fill with cellular structure. This is achieved by offsetting the original surface a given distance. The mesh on the input surface is projected onto the offset surface to form the opposite face of the cubic element to be filled with cellular structure.

The final goal is to insert cellular material into the cubic cells formed between the mesh on the original surface and the offset surface. Because the cubic elements should be defined in a very organized way, all that should be needed is a pattern for the cellular structure that can be duplicated for each existing cube.

Figure 1.2 shows the planned progression from input surface to surface mesh to offset mesh to full cellular structure.

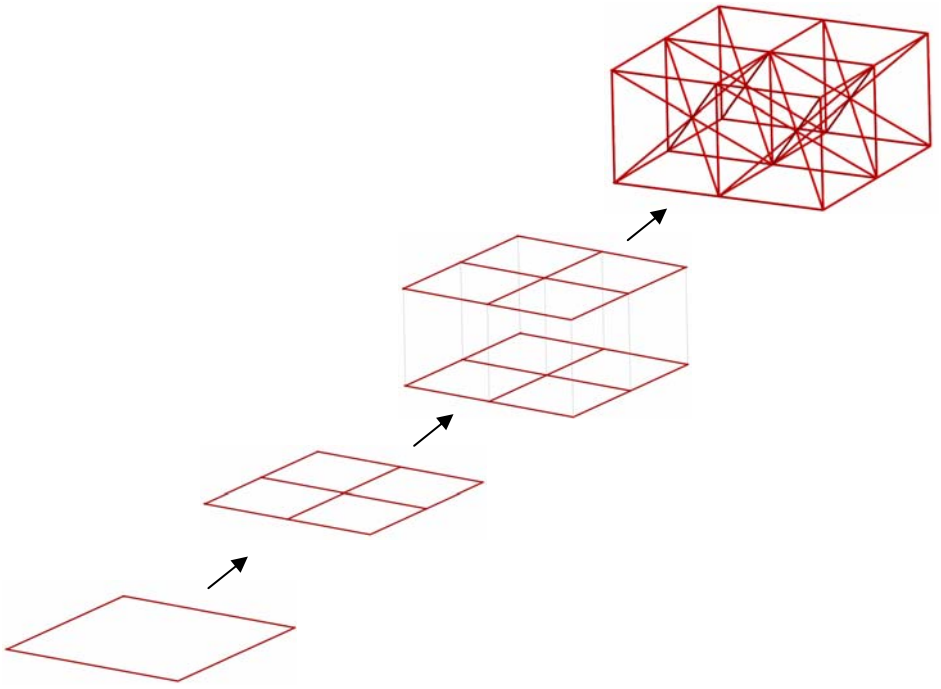


Figure 1.2: Cellular structure process

1.4 Organization of Thesis

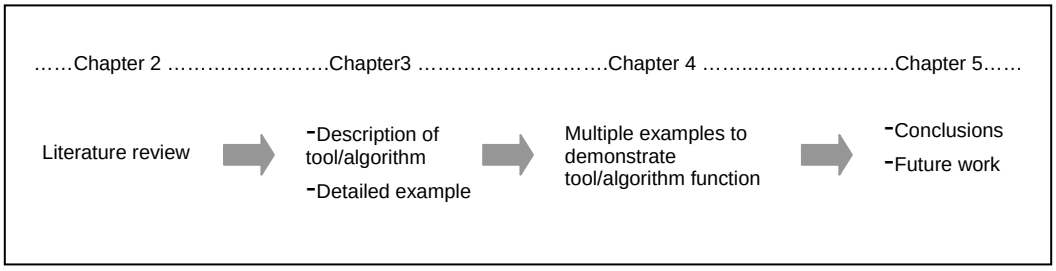


Figure 1.3: Thesis organization

The second chapter in this thesis is a literature review. A number of areas that relate to the processes needed to design cellular structure are investigated. These areas include two-dimensional and three dimensional finite element meshing techniques and surface offsetting methods. Many of these areas are well developed but do not fit well with the needs of this research, and current work that combines these areas of research is limited. A review of current cellular materials and cellular material design is also included in the second chapter.

The third chapter is a detailed presentation of the algorithm that has been developed to automate the creation of cellular structure. The description is followed by a detailed example that highlights individual processes in the algorithm.

The fourth chapter presents several example problems. Examples are chosen to illustrate the algorithm's performance with different input geometry. Performance is evaluated in two major categories: Speed and accuracy. Accuracy is evaluated based on standard finite element categories, such as skew, aspect ratio, and size. Speed is also an important criterion. The speed performance is related specifically to the offsetting

procedure. Two methods for offsetting are investigated and compared with accuracy and speed as the primary criteria.

The final chapter is a conclusion to the work and a look at limitations and future possibilities.

2 Literature Review and Research Gap

The following is a review of topics related to the research reported in this thesis. After completing the review, the research gap is identified.

2.1 Literature Review

The following research areas were reviewed in conjunction with the development of the research presented in this thesis. Two methods were initially explored to create cubic primitives in which to insert cellular structure: 1) Generate two-dimensional finite element mesh on a surface and project it onto an offset surface, 2) generate a volume between the input surface and a calculated offset surface and generate three-dimensional finite element mesh for the volume. Therefore, finite element meshing techniques for two-dimensional and three-dimensional elements and methods for offsetting are studied. A method utilizing two-dimensional mesh was ultimately chosen. To create the two-dimensional mesh with the method presented in this thesis, the input surface is flattened, so flattening methods are also examined.

2.1.1 Flattening

A significant amount of research has been conducted on the flattening of three-dimensional surfaces onto two-dimensional planes. Applications are often in the areas of textile production, such as pattern making for shoes and clothing.

Most research in this area focuses on minimizing an energy function. Many start with a triangulated surface, then systematically project the individual triangular elements onto a plane, while maintaining dimensions as close to the original dimensions as possible. McCartney, et al [1] use a very simple, geometric technique to flatten the surface, and any change from the original dimensions results in a stored energy, assuming every triangle edge is, in effect, a spring. Tam, et al [2] use strain energy calculated for each triangular surface to measure distortion. In both cases, an iterative process is used to reposition triangle vertices to minimize the stored energy.

Zhong, et al [3] use a more complicated method, which applies a force to unfold pairs of triangles. A driving force is also applied which results in a velocity toward a plane. The triangles then collide with the plane to form a flattened surface. This method requires both a velocity redistribution and strain minimization.

In [1] and [3] and in many other instances, darts or breaks in the surface are allowed when strain in an area is particularly high. This is particularly important when making patterns for clothing because distortion would yield highly unfavorable results. Therefore, it is important to these flattening applications.

In the research presented in this paper, the need for strain reduction is eliminated by breaking the input surface into smaller areas which minimize curvature. There is also no need for darts because of minimized curvature and because the resulting flattened surface is used for reference. Therefore, the distortion will not lead to a similar undesirable result in a produced part.

2.1.2 Automated Mesh Creation

Automated mesh creation is studied for methods of meshing surfaces with shell elements and volumes with solid elements. Both methods are relevant to cellular structure design.

2.1.2.1 Quadrilaterals

The primary application for research into automated quadrilateral mesh of a surface is finite element analysis. Triangular mesh can be created and optimized using a number of methods, but quadrilateral mesh produces less error in the results of a finite element analysis. The increased use and complexity of finite element models has lead to a large amount of research in the area.

Meshing techniques fall into two general categories: Mapped mesh and free mesh. Mapped meshing techniques can produce very good quadrilateral mesh results, but they often require extensive user interface and can be limiting in regard to what types of surfaces can be meshed. Mapped mesh can only be applied to three- or four-sided parametric surfaces or surface patches. Arbitrary three-dimensional surfaces, which are the focus of the work presented in this thesis, would represent significant challenges for most mapped meshing procedures.

The most basic free meshing techniques involve triangular mesh because any surface can be meshed with triangular elements. Some techniques start with a triangular mesh then combine adjacent elements to form quadrilaterals [26]. These methods typically produce highly skewed elements and have regions where size varies widely.

Because a Voronoi tessellation is the dual of a Delauney triangulation, Voronoi tessellations are also often used for the basis of creating mesh due to the ability to mesh arbitrary surfaces. Methods incorporating Voronoi tessellations typically focus on triangular mesh, but methods have also been developed to create quadrilateral mesh this way [27]. As seen in Figure 2.1a, the Voronoi tessellation divides a surface into a number of polygons, which are then individually meshed. It can be seen in Figure 2.1b that the mesh is not square and that the size of elements in each polygon of the tessellation is not regular.

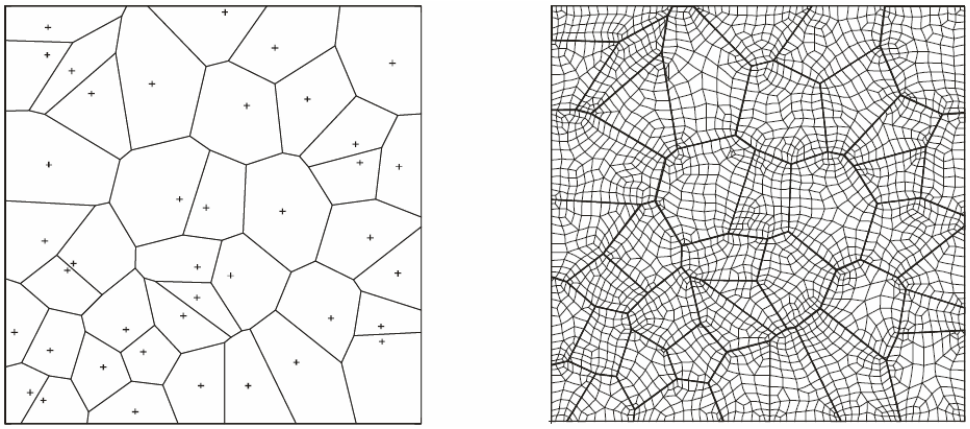


Figure 2.1: Automated quadrilateral mesh using a Voronoi tesselation

Joe [4] and Park, et al [5] have each developed other effective methods for filling a polygon with quadrilateral mesh. In [4] a desired number of quadrilateral elements is specified as a means to guide size. This method first fills a region with polygons, and then breaks the polygons into quadrilateral elements based on a geometric algorithm. The large number of added polygonal boundaries within the input surface complicates the

interfaces and leads to many elements of irregular shape and different element sizes. Figure 2.2 shows the result of a test case.

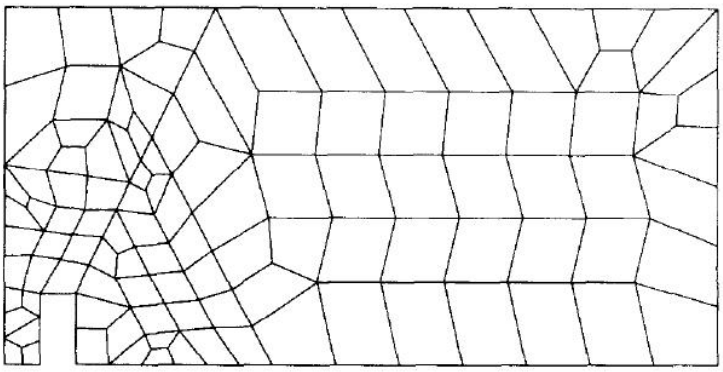


Figure 2.2: Automated quadrilateral mesh in polygons

In [5], a looping scheme is used for different areas of the polygon that is being meshed. The different areas are then meshed from the boundaries of the loop inward. This method is concerned less with size regularity than with element quality. Figure 2.3 show an example result.

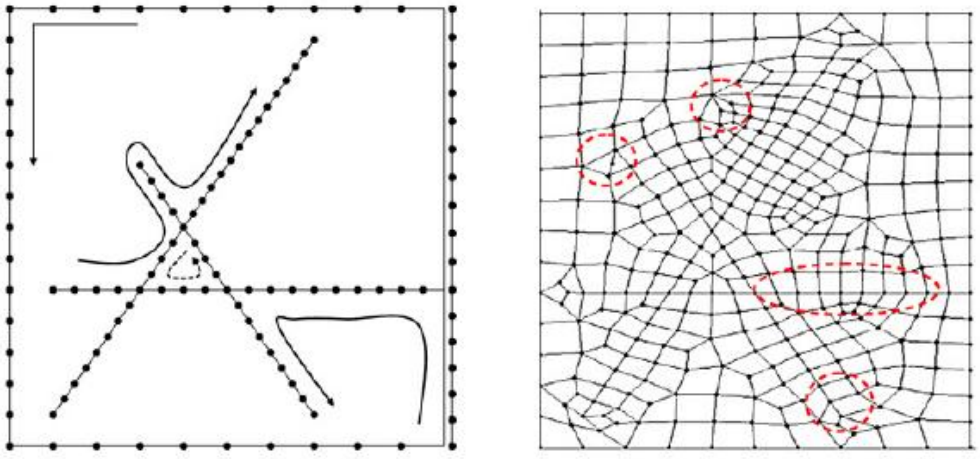


Figure 2.3: Automated quadrilateral mesh in loops

Both methods produce good results for finite element analysis. However, because both are focused on producing quadrilateral-only mesh, some elements are significantly smaller than others, and many, if not most, angles are not approximately square. These results share the same characteristics of all automated mesh generators and are not ideal for the preliminary steps of laying out cubic truss primitives. The desired base for three-dimensional truss primitives is a square grid.

2.1.2.2 Hexahedral Mesh

As with the quadrilateral methods above, hexahedral mesh generation for a volume is primarily used for finite element methods where the key goal is to produce higher-order elements for analytical accuracy. Many methods have been developed for this purpose [6, 7]. Solid meshing an arbitrary surface is often difficult or impossible with some methods, and when mesh can be produced, elements are often different sizes and stray from cubic form, especially in transitional regions.

A simple method for creating hexahedral mesh is sweeping. Swept mesh is produced by generating mesh on a planar cross-section surface which is then translated along a guiding curve. Nodes are produced at regular intervals along the translation. Results are usually very good, but the types of volumes that can be produced with this method are very limited. Only volumes that can be extruded along a smooth curve and that have a uniform cross-section are viable. Lai, et al [8] have developed a method to separate volumes into multiple, smaller volumes that can be swept. There are still many limits to the types of surfaces that can be meshed this way. Many arbitrary surfaces

would not be accommodated. Sweeping methods also require the input of guiding curves, which would not necessarily be straightforward to calculate or integrate into an automated process. Another complication is that for the original surface to be swept it must be meshed with only quadrilateral elements.

There are a number of other hexahedral mesh generation techniques, including mapping, plastering, whisker-weaving, tetrahedral-based, and grid-based methods.

Mapping methods can produce high quality results, but often require significant user input. Mapping methods also require that the meshed surface be broken down into a group of regular regions. With an arbitrary surface, this can lead to small areas filled with poor mesh and often elements that are much smaller than desired. It may also be impossible to map mesh some arbitrary surfaces.

Whisker-weaving [9] and plastering [10] are both "advancing front" methods. They begin with mesh on the surface of a solid and work in toward the center. Plastering often encounters voids in the center of a solid that either produce bad element quality or cannot be meshed with hexahedral elements at all. Whisker-weaving avoids this by creating internal loops that dictate the meshing within the solid. Kawamura, et al [9] have developed a method to improve the loops and eliminate self intersections and other problems that create bad elements. The improvements often lead to a refinement of the mesh in some areas, and the resulting meshes can still suffer from irregularities at the center of the solid. This method is not practical for creating uniform layers of cubic elements.

Tetrahedral-based methods produce all tetrahedral mesh first and then break each tetrahedral into four hexahedral elements. The advantage of this method is that creating a

tetrahedral mesh is generally easier than producing a hexahedral mesh. The result, however, is generally poor quality mesh. Elements are not approximately square, and orientation is not regulated. Owen [11] developed a method to improve the quality of the hexahedral elements produced from tetrahedral elements. While more elements are a good quality, problems with orientation and shape are still present. This type of method also eliminates the possibility of creating uniform layers of elements/truss.

Grid-based methods begin by superimposing a grid of cubic elements with nodes over a volume. Then nodes and elements outside the volume are eliminated, and inside nodes are projected onto the surface. Zhang, et al [12] have developed a grid-based method to mesh a solid. Their method makes advancements in dealing with surface boundaries and surface mesh to improve element quality. Figure 2.4 shows a result from their research. Mesh is generally good for FEA applications. However, in the corner the transition mesh is much finer than the rest of the body. Elements are also distorted from cubic. Lee and Yang [13] have developed a method that uses a custom grid instead of a cubic grid to improve shape results. The mesh shape improvements refer to analytical results, however.

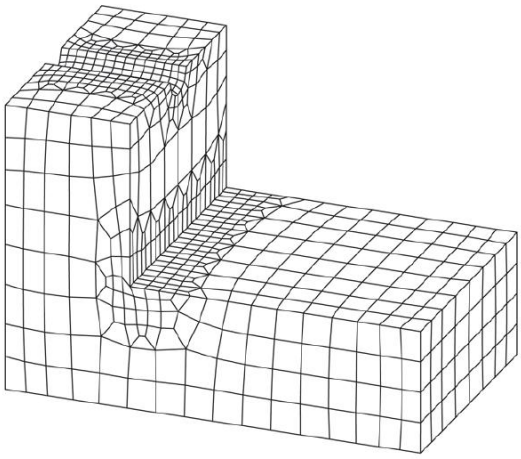


Figure 2.4: Automated hexahedral mesh

Transitional mesh areas are present in many methods of automated hexahedral mesh generation for finite element analysis, which is not acceptable when the desired outcome is a uniform layer of cubic elements.

The automated mesh generation techniques presented above all suffer from one or more of the following problems: Local mesh refinement, unregulated orientation, skewed element shapes, impractical user interface, or inability to deal with arbitrary surfaces/volumes.

2.1.3 Offsetting

Calculating the offset of a surface is useful for many applications. For this application, a uniform layer of cellular structure is desired, which makes offsetting necessary. Figure 2.5 shows an example of inside and outside offset surface in two dimensions. The offset surface in two or three dimensions is at least a given distance from all points on the original surface.

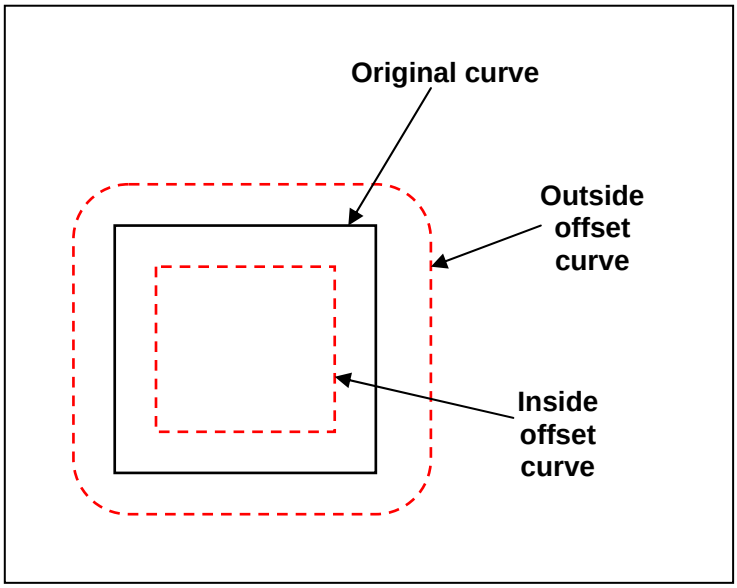


Figure 2.5: Offset example

A point-based method for calculating the offset surface was developed by Chen, et al [14]. In this method, the input surface is sampled to created many points that lie on the surface for reference. Then, a set of offset points are established by projecting the sampled points normal to the surface. The offset points are then compared to sample points and each other to establish the point cloud that represents the offset surface. The point cloud can then be triangulated to establish a new surface.

Other methods developed for offsetting involve NURBS surfaces or require "water tight" volume inputs. These methods are impractical for the purpose of this research. A point-based method makes sense for this application because the surface will already be sampled for other applications, and a reconstituted surface is not necessary.

A method similar to [14] was used in this research, along with a simplified method based on a similar point-based method. The method based on [14] produces good results but can be time consuming.

2.1.4 Cellular Structure Design

Some work has already been completed in the area of cellular structure design. It is studied here to understand what is currently available and opportunities for improvement

2.1.4.1 Regular Periodic Cellular Structures

A number of methods for creating regular periodic cellular materials have been developed. Honeycomb core has long been a popular lightweight cellular material for aerospace applications, etc. Many companies mass produce honeycomb core. Truss core has some structural advantages to the honeycomb configuration, which has led to the development of new types of cellular material. In [15], Queheillalt, et al discuss a fabrication method to create periodic aluminum truss structure by combining an extrusion with electrodischarge machining. The result is a uniform layer of regular truss between two plates.

Sypeck, et al [16] have also designed a method to create periodic metallic truss between to plates. Their method involves deforming hexagonally perforated metal sheets to create the truss elements. The truss is then bonded to face sheets using a transient liquid phase approach. The result is tetrahedral truss core between two face sheets.

Approaches using rapid manufacturing of periodic truss elements have also been developed [17-19]. These methods generally focus on stereolithography as the rapid manufacture type, but some are applicable to many kinds of rapid manufacturing

techniques. The results are a variety of truss types. This shows that many configurations are possible and that there is room to expand similar technologies to conformal truss.

2.1.4.2 Conformal Cellular Structures

Some past work has been completed in the area of conformal truss design. Wang and Rosen [20] have developed a parametric modeling method to produce custom truss structures. This method uses Bezier surfaces to decompose input geometry into truss elements. The result is conformal truss for a given surface or volume. This method often requires surfaces to be broken down into multiple Bezier approximations and related back to each other through continuity requirements. Because the surface is only approximated by the Bezier curves, some truss vertices may not lie on the original surface. Also, for three-dimensional truss, the bounding surfaces must be known. If a uniform layer of truss is desired for a surface, another means of calculating the volume must be used first.

2.2 Research Gap

Currently, there are many methods for meshing a surface. Any commercial finite element modeling package can create almost exclusively quadrilateral mesh on a given surface. The mesh produced may be sufficient, but it would not be optimal for cellular design because size and shape regularity are sacrificed.

If a designer using a typical CAD package could calculate the offset surface for his original surface and create a solid, a finite element package could possibly be used to create a hexahedral mesh, which could then be fit with cellular material. With current

methods, it would be extremely difficult to interface a calculated offset surface with an existing surface, and it would require considerable user interface. It is also difficult to create hexahedral mesh for many solids to get a single, or multiple, uniform layers of hexahedral elements with nodes on the offset surface directly across from nodes on the original surface. Many finite element tools allow mesh to be altered, which would allow the user to create appropriate mesh, but this would require a large effort on the part of the designer. Therefore, solid meshing with traditional techniques does not work well for the application of cellular structure design.

Currently there are no applications that meet the specific needs of cellular structure design.

2.3 Summary

Chapter 2 presents a literature review of fields related to the design of cellular structure. The method for designing cellular structure developed in this thesis creates finite element mesh to fill with cellular primitives. Therefore, many two- and three-dimensional finite element meshing techniques are explored. All of the methods investigated are designed specifically to create elements with good qualities for analytical solutions. This is often not ideal for cellular material design. Many of the techniques studied are also only applicable to limited types on input surfaces. Offsetting and flattening operations and current methods of designing and manufacturing cellular materials are also discussed.

The research gap addressed by this thesis is to combine and adapt modeling tools that are already developed to focus on the specific task of cellular structure design with the hope that it will facilitate the use of cellular structure in a wide range of applications.

3 Algorithm Description

Cellular structures such as honeycomb core, lattices, and foams are useful in many applications, especially when minimizing weight is crucial. They offer a high stiffness to weight ratio and often improved thermal and acoustic properties. Some cellular structures are already mass produced, like honeycomb and various lattice configurations. These are regular structures, meaning that the exact same unit cell is replicated many times to form a system of structures. Foams are stochastic rather than regular, with a random arrangement of voids that have a general bulk property.

Regular or random cellular materials have many of the beneficial qualities of cellular structures, but they are necessarily limited in performance. Only a certain variety of regular cellular structures are produced, so a designer must pick one that has the best shape from the choice available. This might result in having cellular structure that is heavier or more complex than needed for the application.

There are two primary reasons that a customized cellular structure is desired. The first is that with a customized structure material is placed only where it is needed for a specific application. The second is that a customized cellular structure is conformal with the surface or volume that it interfaces with. Currently, honeycomb core, for example, must be cut to size. Partial cells at the boundaries could have weaknesses or altered stiffness properties.

The purpose of this chapter is to describe the components of an algorithm that creates a custom cellular structure layout for a given surface. The desired output is a

structure design that takes advantage of the unique features of cellular materials and eliminates the weaknesses of the regular or random materials currently available.

Figure 3.1 shows a map of the process to create three-dimensional cellular structure from an input surface. Each step in the process will be described in detail below. The input for the algorithm is an STL file, which is a convenient universal file for this application. First, the STL file is read in and sampled to define the input surface. Next, the input surface is separated into smaller areas in the partitioning step. The partitions are separately run through the flattening step and grid application (indicated by the dashed lines in Figure 3.1). The grid applied is basically a finite element mesh. The meshes on all partitions are then joined back together in the boundary matching step. A quality check is then performed on the resulting mesh. At this point, the input surface has been covered with a mesh that is mostly quadrilateral elements. Now a volume must be defined to hold a three-dimensional cellular structure. The input surface is offset, and then the mesh on the input surface is translated to the offset surface to form three-dimensional elements. These elements are filled with a cellular structure definition and can be written out in numerous formats.

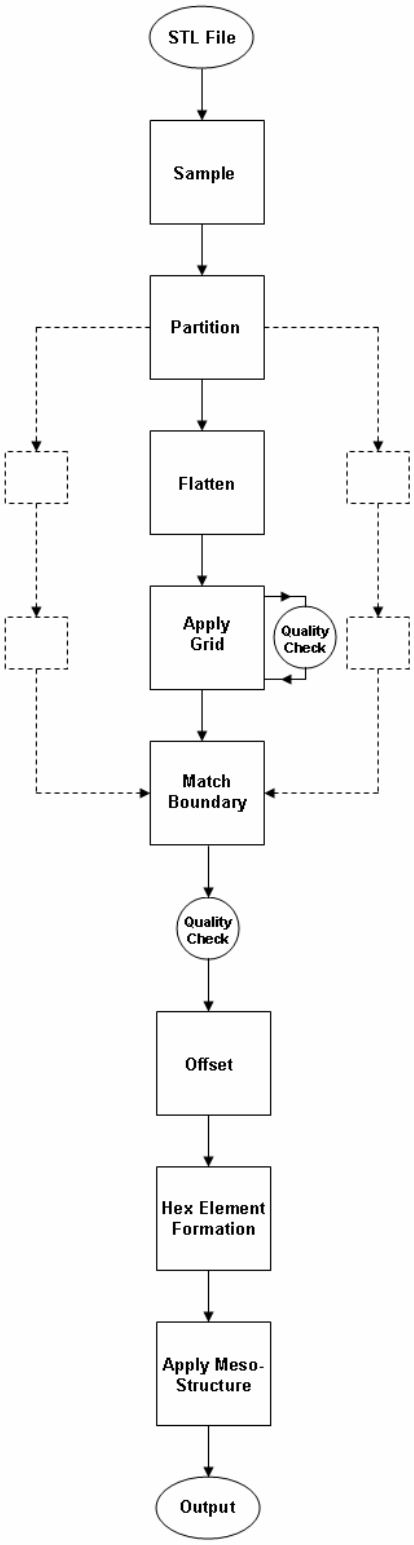


Figure 3.1: Algorithm map

3.1 STL Processing

The input for this algorithm is an STL file, which describes surfaces as a composition of triangular faces [21]. A sample of the file format and resulting triangular face are shown in Figure 3.2. Each triangular face is defined by its vertices and an outward-pointing normal vector. The order of the vertices relates to the normal by right-hand rule.

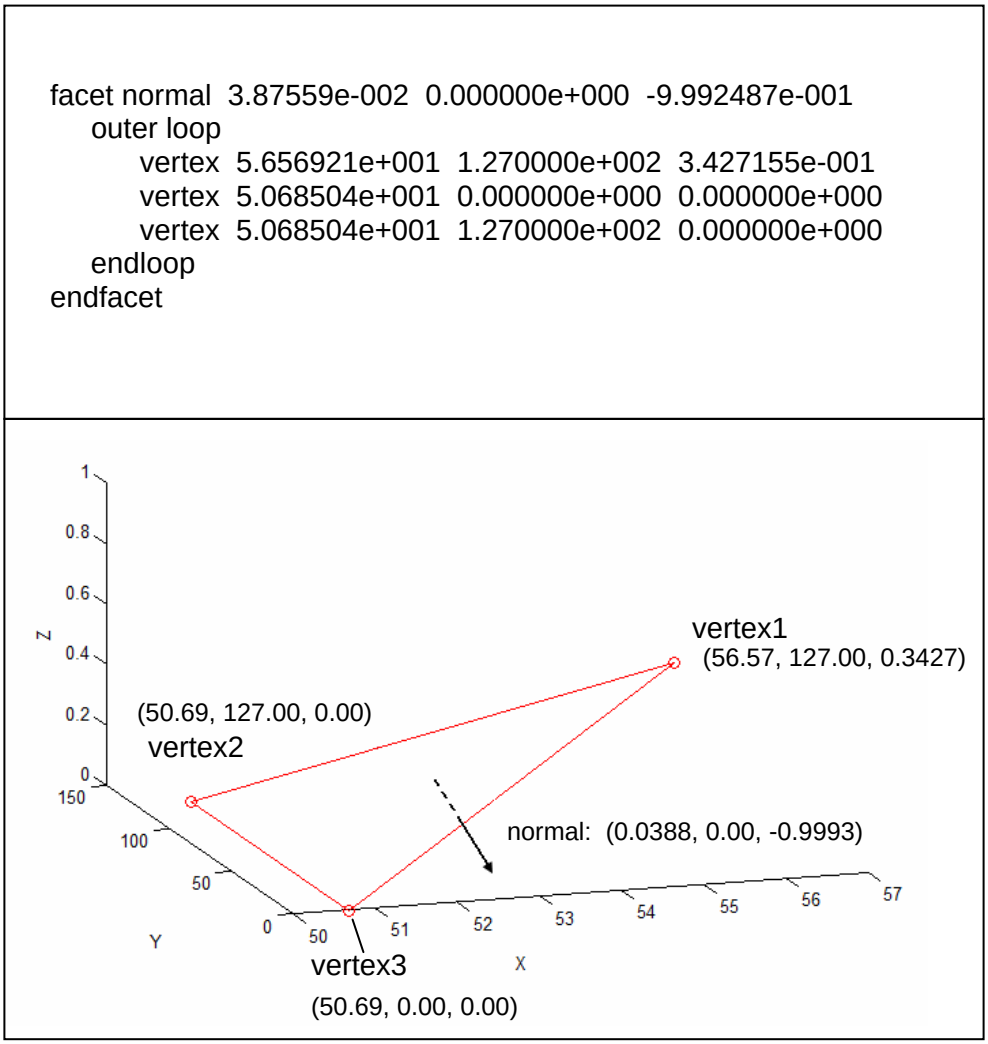


Figure 3.2: STL file format

The STL input must be processed into a more usable format, and because each triangular face is formed independently, connections must be established between faces that share faces and/or vertices.

When the STL file is read, each triangular element is documented with its vertices, lines, and normal. Each triangle has three lines: vertex 1 to vertex 2, vertex 2 to vertex3, and vertex 3 to vertex1. The vertices are documented by their x-, y-, and z-coordinates in a separate array and referenced to their face. Once all points are read in, there will be many duplicate entries because the faces are defined individually, without reference to neighboring faces. Therefore, after all points are established, points sharing the same spatial coordinates are consolidated, and the face variable is updated to reference the unique set of points.

The lines are also documented separately by their end points as well as the normal of the face they are associated with. The normals are included for use later in the offsetting. The points defining the lines are from the unique set of points, so that their point references match the point references in the definitions of the faces. Line definitions also have duplicates. They cannot be consolidated, however, because of their reference to the face normal. Therefore, a new variable is established to match lines to their duplicates, if two faces share an edge. This variable documents the pairs of lines, whether a line is shared by a pair of faces or not, and if the faces that share an edge form a concave or convex angle. This variable is important for offsetting because it determines whether a point on a line should be offset or not, as will be explained further when offsetting is discussed in detail. It is also important for sampling, to avoid creating duplicate sampled points.

3.2 Sampling

Sampling the input geometry serves a number of purposes. The primary purpose is for parameterizing the surface, but it is also needed for the offsetting function and matching of boundaries, which will be discussed more in following sections. Sampling the surface involves creating new points that lie on the surface. This refines the surface definition by increasing the number of data points recognized by the algorithm.

All vertices from the original triangular faces of the input STL file are added immediately to the set of sampled points as they are defined, with no additional processing. They are added separately, instead of as part of the line or face sampling, to avoid duplication or exclusion. They are included to keep the important definition of the faces' corners as part of the sampled set and for offsetting.

Lines and faces are sampled using the same universal size increment. This is a user input which can be increased or decreased to control the number of points being generated. A higher number of points will give a better result, but will increase processing time.

3.2.1 Line Sampling

Line sampling creates new points along the lines that represent each edge of each face from the input STL surface to further define them for later steps. An example of line sampling is shown below in Figure 3.3. The length of each line is divided by the universal sampling increment, shown as L_S in the figure. The ratio of line length to the sampling ratio is then rounded up to the next whole number using a ceiling function. The

calculation is shown in Figure 3.3a. If the ceiling function returns a one, it is automatically increased to two. This is to ensure that all lines have at least one sampled point other than the endpoints. A new local sampling increment is then established for each line, shown as L_{SL} in Figure 3.3b, which is the original line length divided by the number returned by the ceiling function. A sampled point is then created along the line every L_{SL} units, excluding the beginning and end points. The sampled points are shown in Figure 3.3b.

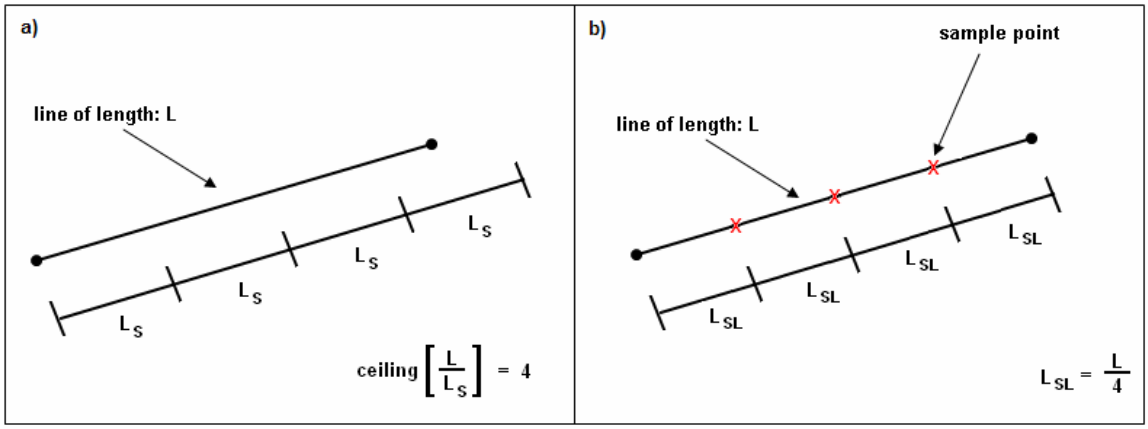


Figure 3.3: Line sampling

3.2.2 Face Sampling

Faces are sampled in a manner similar to lines. Figure 3.4 shows an example of face sampling. The first two lines of a face description are used. Each line is divided by the universal sampling increment, and the ceiling function value is taken, just as in line sampling. The maximum of the two ceiling function values is used to create local sampling increments for each of the lines. This is shown in Figure 3.4a.

The local sampling increments are used to create temporary sampled points along the lines. Then between corresponding points on opposing lines, a new temporary line is established, as shown in Figure 3.4b. This line is then sampled exactly as other lines.

Figure 3.4c shows the sampling of one of the temporary lines, L_{T4}. The ratio of the length of L_{T4} to the universal sampling increment is between five and six. Therefore, it is rounded up to six by the ceiling function, and five evenly spaced sampled points are created that reference the face they lie on.

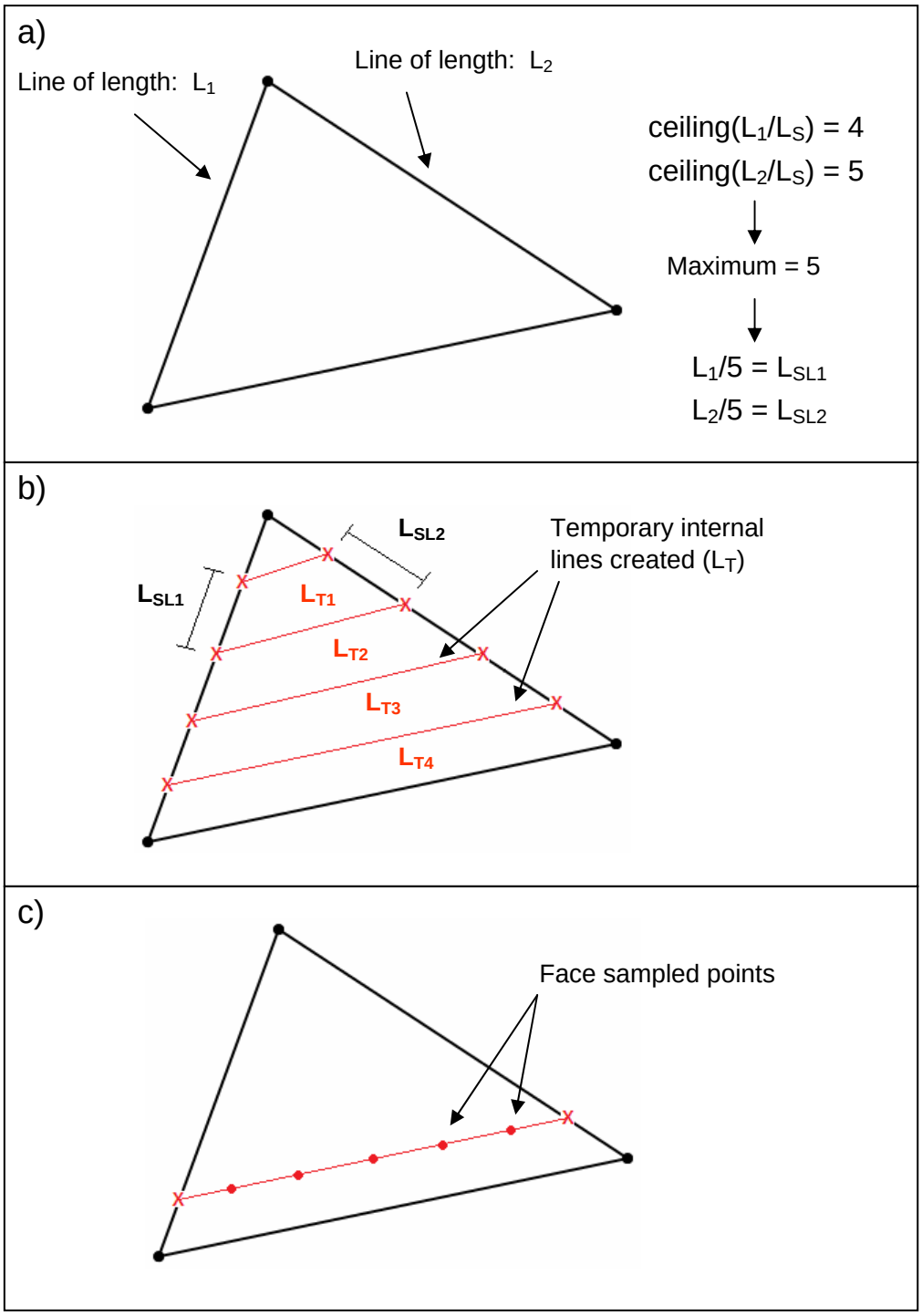


Figure 3.4: Face sampling

The set of sampled points now defines the entire area of the surface. Each sampled point references the vertex, line, or face that it lies on. These points are stored for use in future operations.

3.3 Partitioning

The partitioning operation divides the input surface into multiple smaller surfaces. The partitioning step is necessary for future flattening and meshing steps.

Because many applications of cellular structure can involve closed or semi-closed volumes, such as a hollow sphere or conical surfaces, rather than planes or open faces, it is necessary to divide the surface before flattening. Otherwise, the surface will be flattened partially or completely onto itself. For example, flattening of a spherical surface onto a plane is impossible without either a break in the surface or complete overlap of opposite hemispheres. Even volumes that are not completely closed, such as open-ended cylinders, cannot be flattened without at least one break or else they will also experience overlap.

Partitioning is also important to the quality of the mesh result. Even when working with surfaces that can be flattened without a break, such as a hollow hemisphere, the flattening will cause distortion. This distortion will then be translated to the mesh result. The purpose of partitioning in this case is to diminish the distortion caused by flattening.

The input for this step is the surface read from the STL file. Most partitioning methods use a triangulated surface [22]. The partitioning routine divides the surface into multiple areas using the normal vectors of the triangular faces that make up the input

surface. Curvature is a common way to separate a surface into partitions [23,24]. The partitioning method used in this algorithm is very simplified. An initial face is chosen, and then its normal is compared to those of the adjacent faces. If the angle between face normals differs by more than a given amount, they are put into separate partitions. The angle of difference allowed within a partition is a user input. Less difference can decrease the amount of error in the flattened partition, but will increase the number of partition interfaces, which may increase the number of irregular elements at interfaces.

The result of this step is a set of partition definitions that contain groups of adjacent faces with similar normal vectors, within a given limit. The partition definitions also include information about the boundaries between partitions.

3.4 Flattening

The flattening step takes each partition separately and "unfolds" it onto a plane. The purpose of flattening the surface is to facilitate parameterization. Two-dimensional surfaces can easily be divided into a number of regular, in this case square, regions that will serve as the base for the hexahedral element definitions.

The first step of the flattening procedure is to re-orient the triangular faces of the partition to generally face the XY-plane (normal to Z-axis) and to include a nominal offset in the Z-direction to differentiate re-oriented elements from flattened elements. The primary purpose of re-orientation is to avoid confusion between elements that lie in the XY-plane in their original definition and those that have been flattened onto that plane. Figure 3.5 shows a re-oriented partition ready for flattening. The plot on the left

is the original orientation of the faces in the partition, and the one on the right shows the faces reoriented.

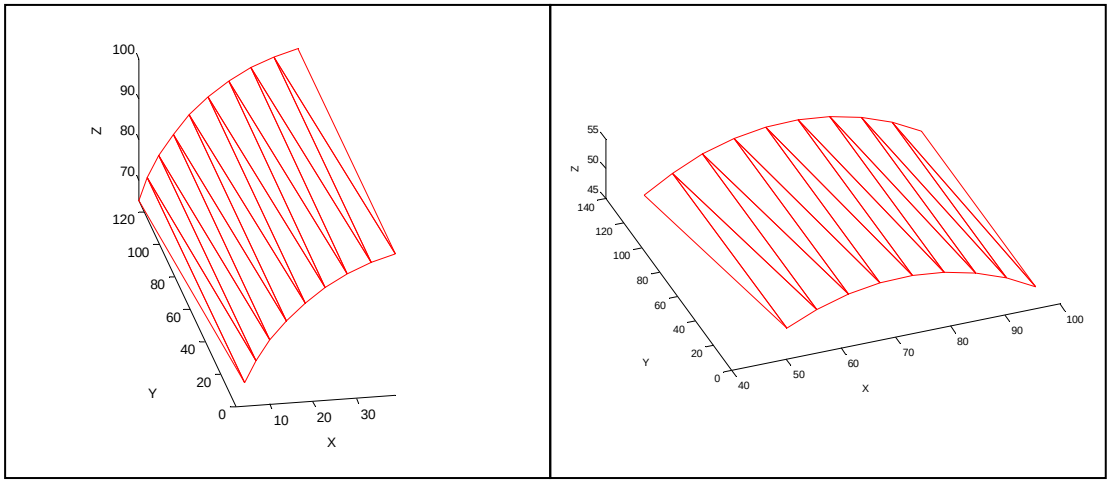


Figure 3.5: Partition in original orientation (left) and reoriented for flattening (right)

Once they are re-oriented, the faces are flattened onto the XY-plane. The flattening technique used here is highly simplified. There are no energy or strain relaxation routines as in other flattening methods [1-3].

The first side of the first face to get flattened is handled with a unique procedure. After that all other sides of all faces are handled the same. First, the initial face is translated down so that at least one vertex is on the XY-plane. Figure 3.6 shows an example of a first face to be flattened. In Figure 3.6a, vertex 1 of the face lies in the XY-plane. The second vertex of the face is then flattened. The location of the second vertex is calculated to maintain the original side length and the original orientation (neglecting the Z-component). The second point is moved to a location along the same XY-vector, which is shown in Figure 3.6a as a dashed line, as its original position from vertex 1 at a distance equal to the original size of that side of the triangle. Figure 3.6b shows the

original length of the side, L1. Figure 3.6c shows vertex 1 and vertex 2 flattened onto the XY-plane.

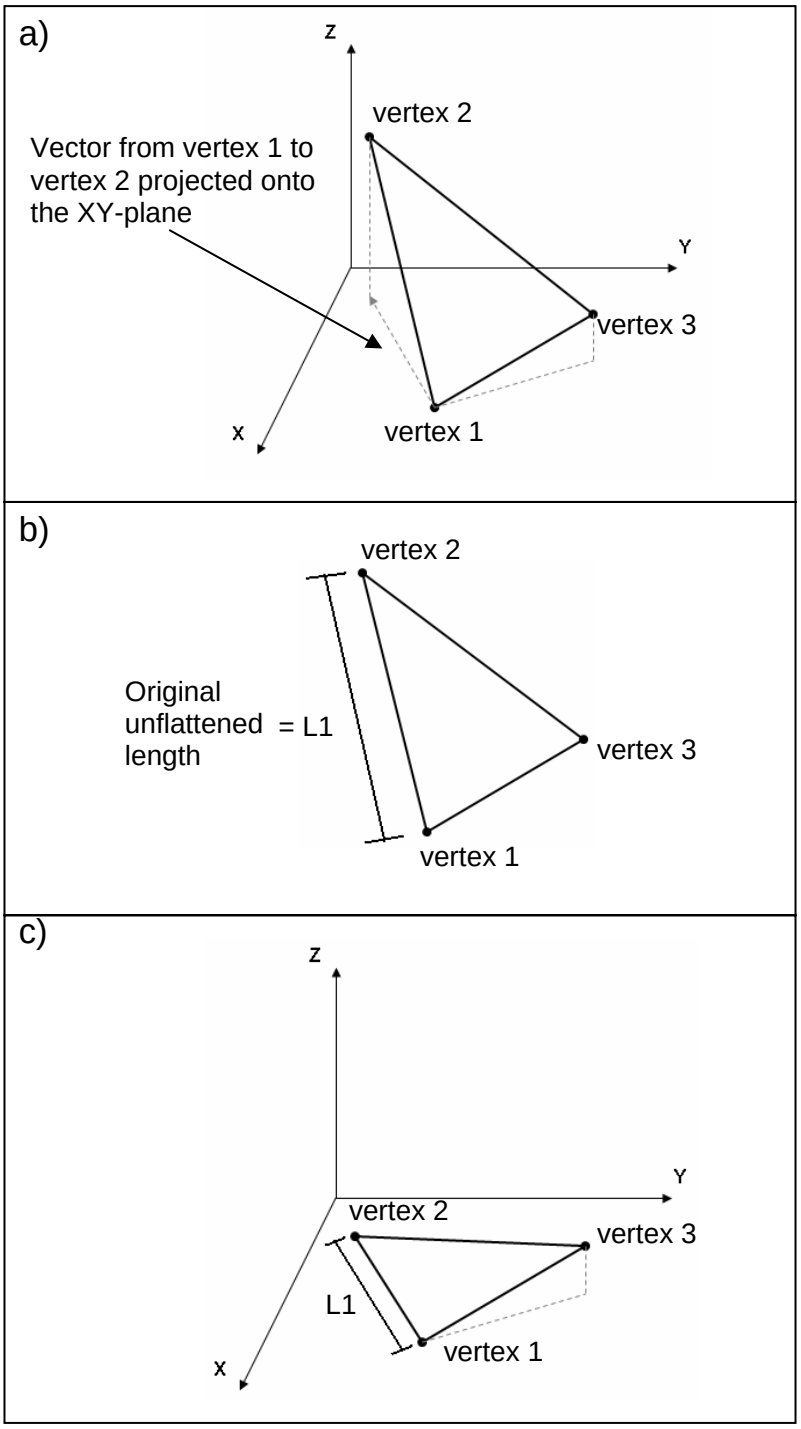


Figure 3.6: Flattening the first side on the first face

All other vertices, after the first two of each partition, are flattened by measuring their distances from the two vertices of the face that have already been flattened. Figure 3.7 continues the first face example from Figure 3.6 above. The only vertex that has not been flattened is vertex 3. Figure 3.7a show the measurements of the sides of the face that will be used as radii to calculate a flattened location for vertex 3. The new location of the third vertex is the intersection of circles of those radii from the flattened locations of the first and second vertices. Figure 3.7b shows the intersection point that is the calculated location of the third vertex for this triangular face.

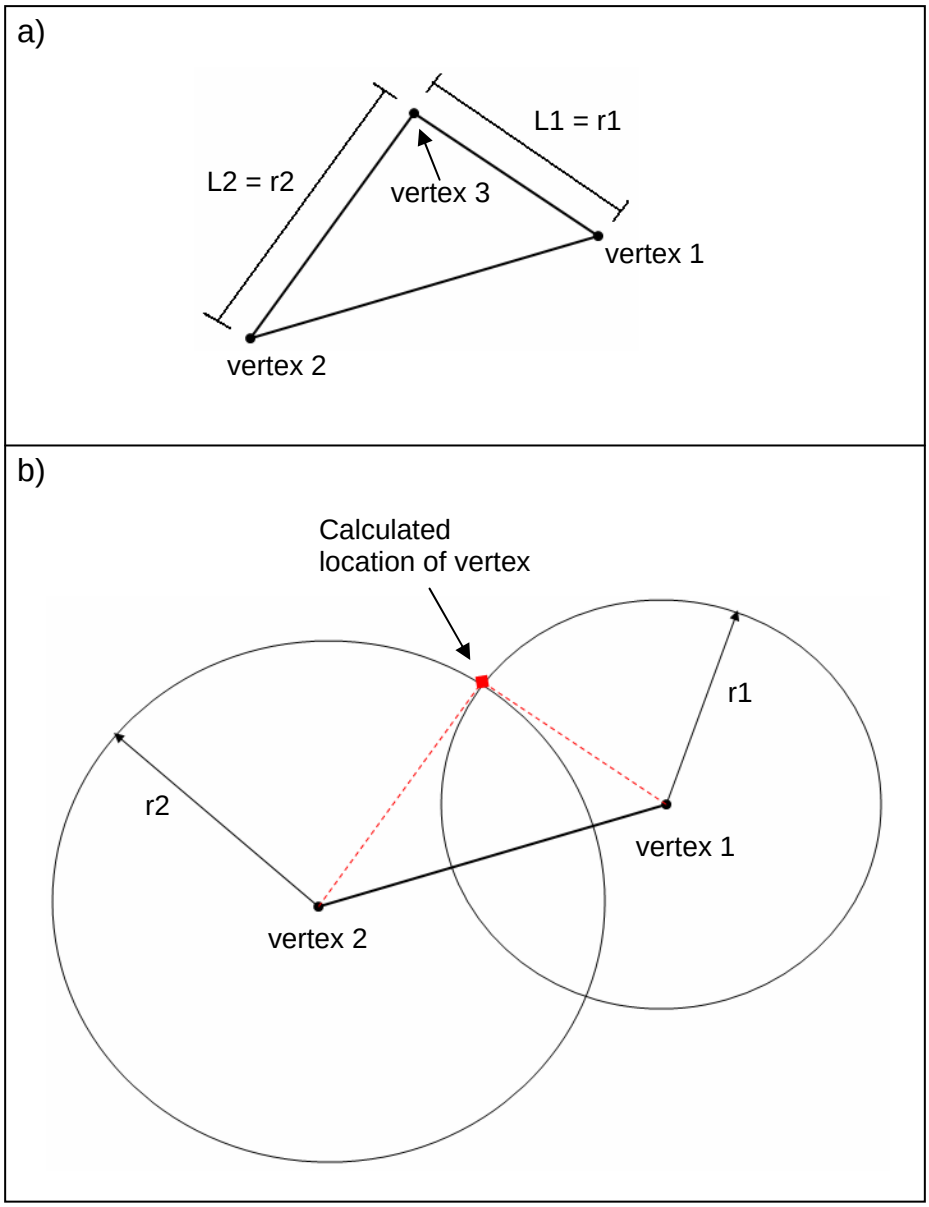


Figure 3.7: Flattening the third vertex of a face

The rest of the faces are flattened in the same way, working out from the flattened points of the first face toward the edges.

If a face is flattened which already had a vertex flattened along with a different face, the location calculated will be averaged with the previous location. Figure 3.8 below shows an example where a point has been flattened with two separate faces, shown as location 3a and 3b on the left. The new location of the vertex is calculated as the average of the coordinates of locations 3a and 3b in the XY-plane. This is also shown in Figure 3.8 on the left. Both faces in the figure then reference the new location, as shown on the right.

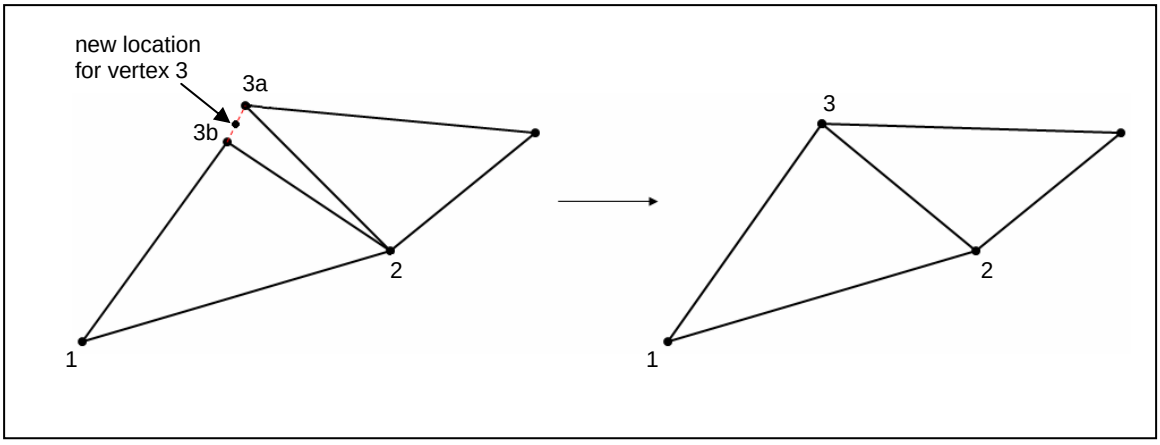


Figure 3.8: Averaging two locations for a flattened point

If no intersection can be found at all for the circles due to distortion from the flattening, the radii are incrementally increased as a percentage of radial length until an intersection is achieved. This will result in an extremely thin face, essentially a line. This occurs rarely, if ever, for a given surface. The STL input files can sometimes have very thin triangular elements that are nearly linear before flattening, which is the primary

cause of this situation in the flattened plane. It is not problematic for further steps. This type of face is simply preserved to maintain boundary definitions and continuity.

Only the input triangular faces and their vertices, not the sampled points, are translated into a flattened plane.

Figure 3.9 shows the partition from Figure 3.5 flattened.

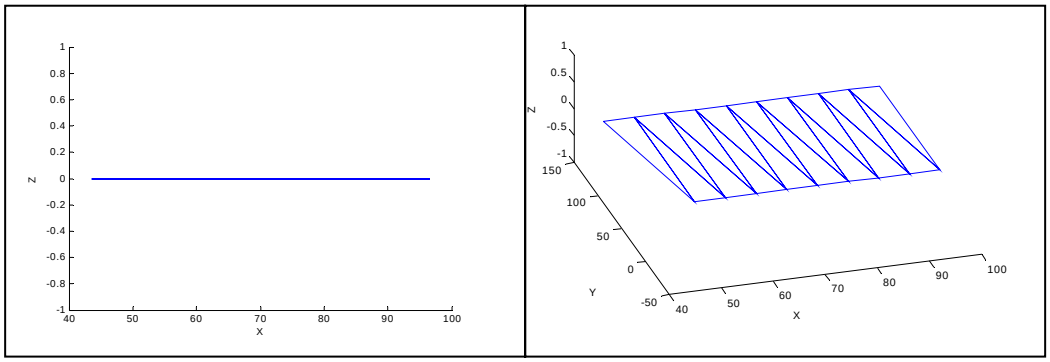


Figure 3.9: Flattened partition

3.5 Mesh Generation

Once a flattened surface is established, it is parameterized using the side length of the base of the desired cubic primitive as the parameter. A grid is used to parameterize the surface, which breaks it down into mostly quadrilateral, approximately square regions. These regions form the bases of the cubic primitives that will later be filled with cellular structure.

Figure 3.10a shows a flattened partition with a grid superimposed. The figure only shows the vertices of the grid, which are represented by small red circles. Each grid vertex that falls within the boundaries of the flattened surface, shown in the figure within the dashed line, is matched to a sampled point by relating its relative position within a

flattened triangle to its corresponding position in the same triangle when un-flattened. Grid vertex locations are related back to the un-flattened triangles because they are to be matched to sampled points, which have not been translated to the flattened faces. This is described in detail in Section 3.5.1 below.

The grid is made to be slightly larger than the flattened partition to ensure that all boundaries are accounted for, as shown in Figure 3.10. Vertex locations that are just outside the boundary will be matched to sampled points on the boundary. The boundary points are identified by going across grid rows of constant y-value and down columns of constant x-value. The first grid location that is not within the boundary in either direction is counted as a boundary point. Figure 3.10b shows the points that are considered for the boundary, as indicated by the arrows in the figure. They are immediately adjacent to a point within the boundary of the surface when moving across or up and down lines of grid vertices.

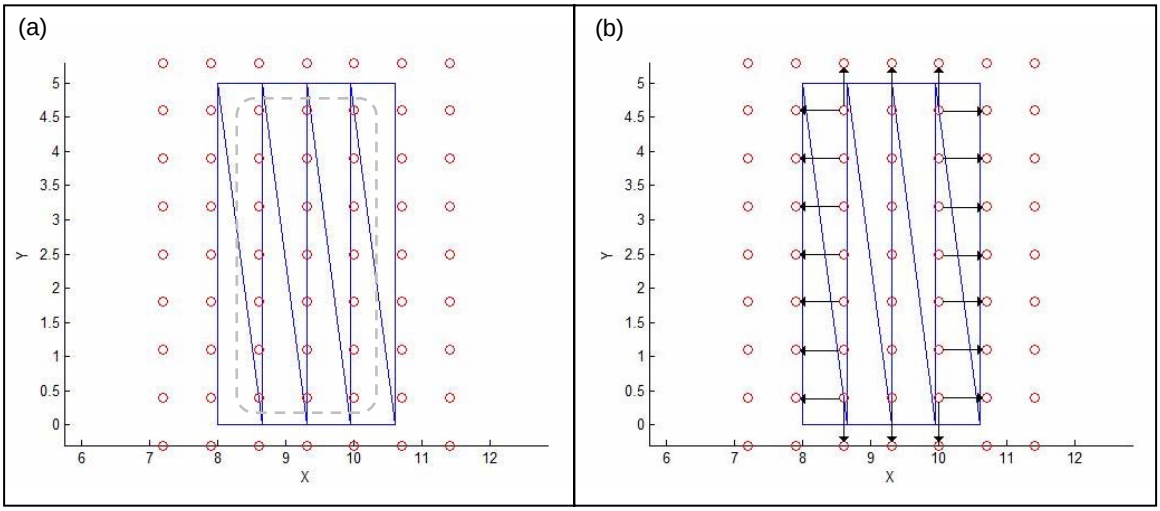


Figure 3.10: Grid superimposed on a flattened partition

Establishing boundary points that are distinct from points in the middle of the structure is important for reassembling the separate partitions after meshing.

All grid vertices that are determined to be either within the boundary of the flattened surface or boundary points are matched to sampled points and stored in separate matrices, one for inside points and one for boundary points.

3.5.1 Inside Points

The first step in matching the grid vertices inside the surface to a sampled point is to find which flattened triangular face it falls within. This is done with a series of cross products. See Figure 3.11. In the figure, the grid vertex of interest is labeled GP. The vectors from P1 to P2 (V1) and from P1 to P3 (V2) are crossed. The direction of this cross product is compared to the direction of the cross product of vectors V1 and V3. The same procedure is repeated for all three vertices. If the directions are the same in all three cases, then the grid point is in the triangle.

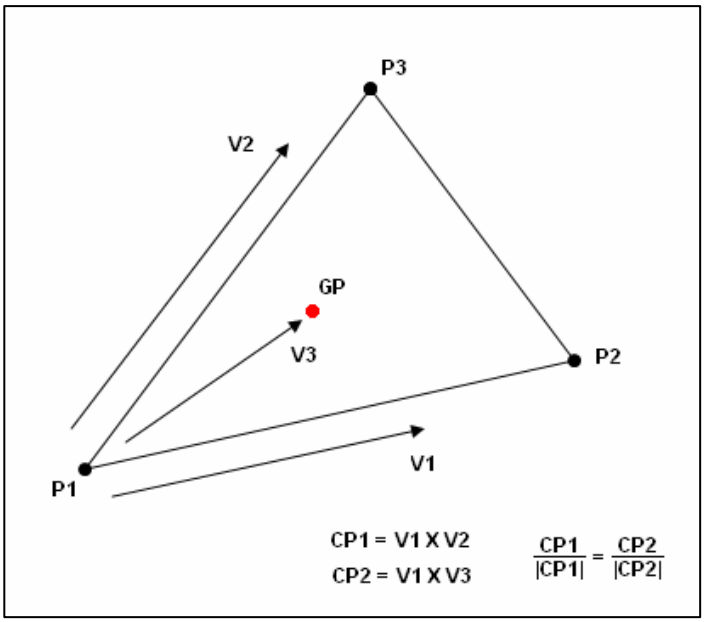


Figure 3.11: Using cross products to determine if a point is within a triangle

Once the grid point is matched to a triangle, the relative position of that grid point within the triangle is captured by three ratios. These ratios are taken from most acute angle to avoid problems with negative dot products, etc. Figure 3.12 shows the ratios and how they are calculated. The line from PT1 to PT2 is perpendicular to the line from P1 to GP.

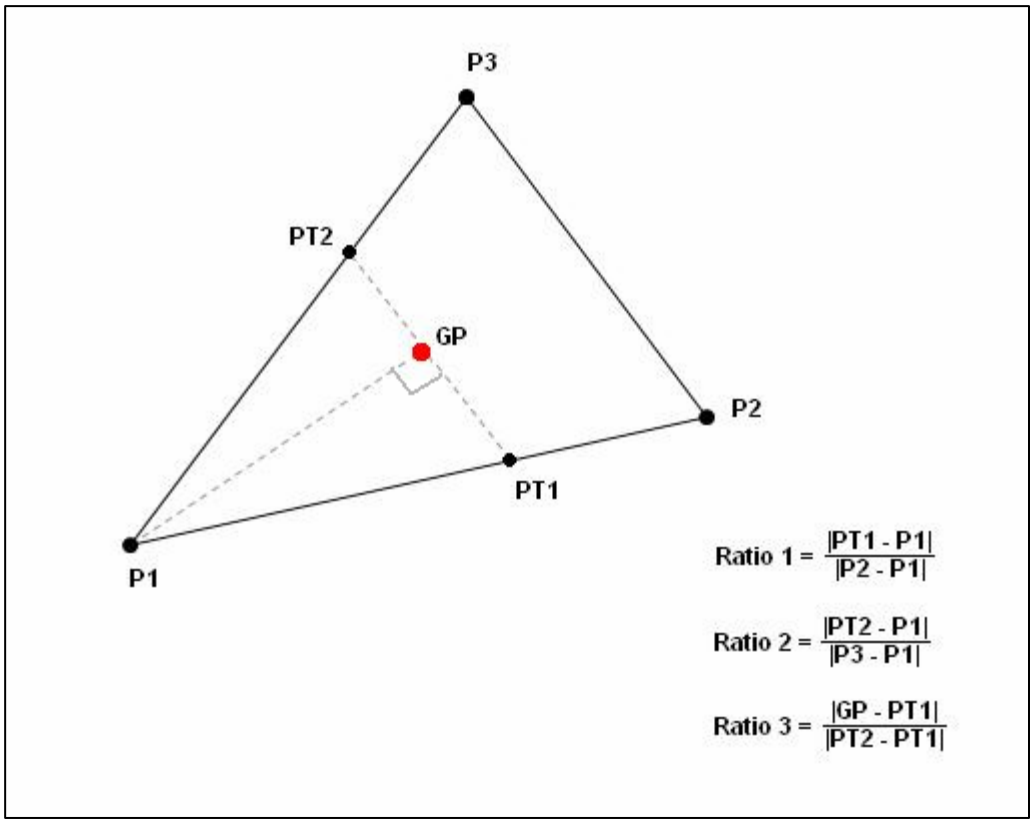


Figure 3.12: Locating a point within a flattened triangular face

The same triangle in its un-flattened configuration is covered in sample points, as shown in Figure 3.13 below. Using the ratios calculated as shown in Figure 3.12, an equivalent point to the grid vertex is found on the un-flattened triangle , labeled as 'C' in Figure 3.13, and matched to the nearest sample point.

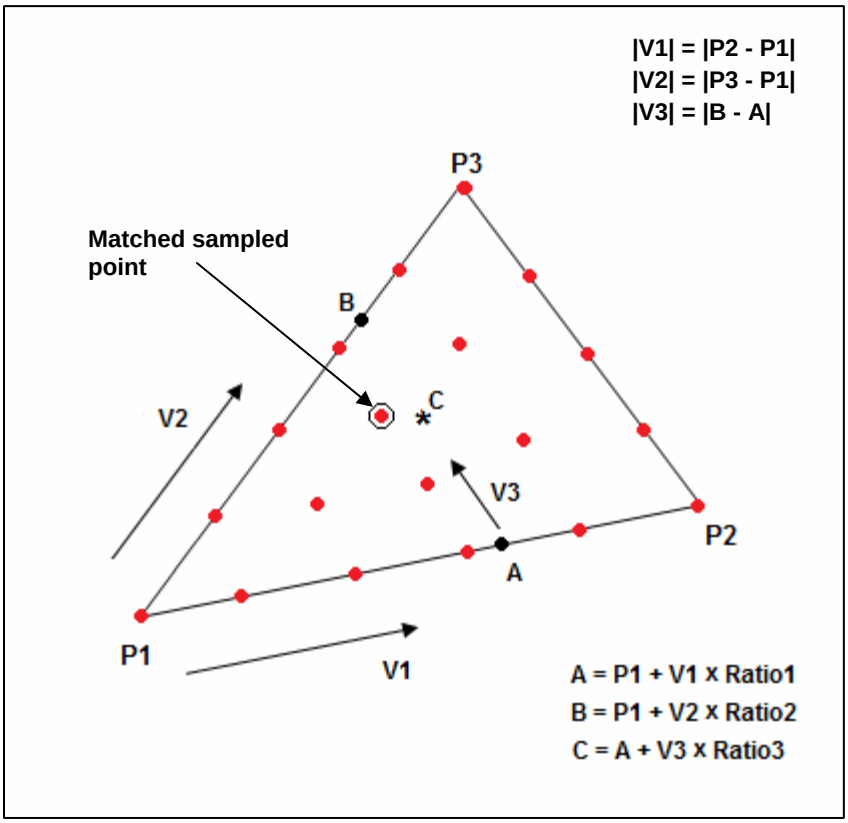
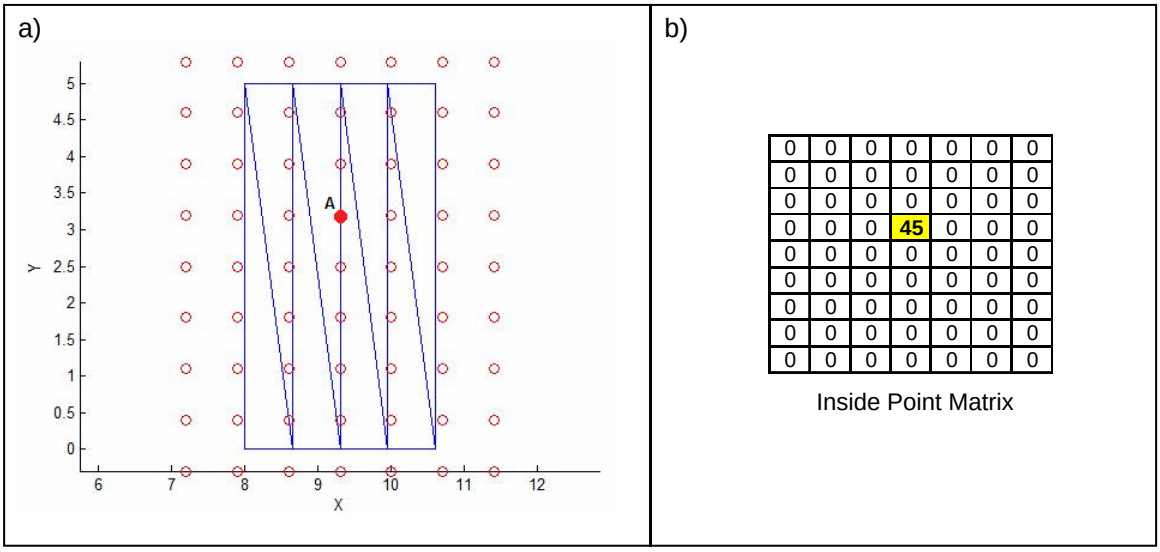


Figure 3.13: Matching a point to an existing sampled point

After an inside point has been matched to a sampled point, it is stored in the inside point matrix. The inside and boundary point matrices both have cells for each vertex of the overlaid grid. Many of the locations will not be matched to a sample point, and are filled with a value of zero for a place-holder. A cell in the matrix that represent a grid vertex that has been matched to sampled point get filled with the number of the sampled point to which it was matched. The matrix keeps the matched points in order spatially and allows boundary points to be identified.

Figure 3.14 shows an example of the inside point matrix. The highlighted point in Figure 3.14a labeled 'A' is matched to sample point number 45. In the figure, it can be seen that there are seven grid vertices across and nine down. Therefore, in the matrix shown in Figure 3.14b there are seven columns and nine rows. Point A is the fourth grid

from the left in the fourth row down, so the matched sampled point, number 45, is placed in row four, column 4 in the point matrix. This is done for all inside points.



0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	45	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Figure 3.14: Inside Point matrix example

3.5.2 Boundary Points

Since boundary points are not within any triangle that is part of the surface, they must be related to the edge of the flattened partition. The intersection of the side of the triangle that forms the partition boundary and the line between a boundary point and its closest inside point is found. Then a ratio is again used to document the location. This is shown in Figure 3.15a. The ratio is then used to find the equivalent point for the un-flattened configuration. Figure 3.15b shows the same triangle un-flattened with its related sampled points. The point labeled ‘A’ is found using the ratio from Figure 3.15a and is matched to the nearest sampled point. Just as shown above, the matched sampled

points are stored in a matrix. The inside and boundary point matrices are kept separate because it is important to know which points are on the boundary for matching later.

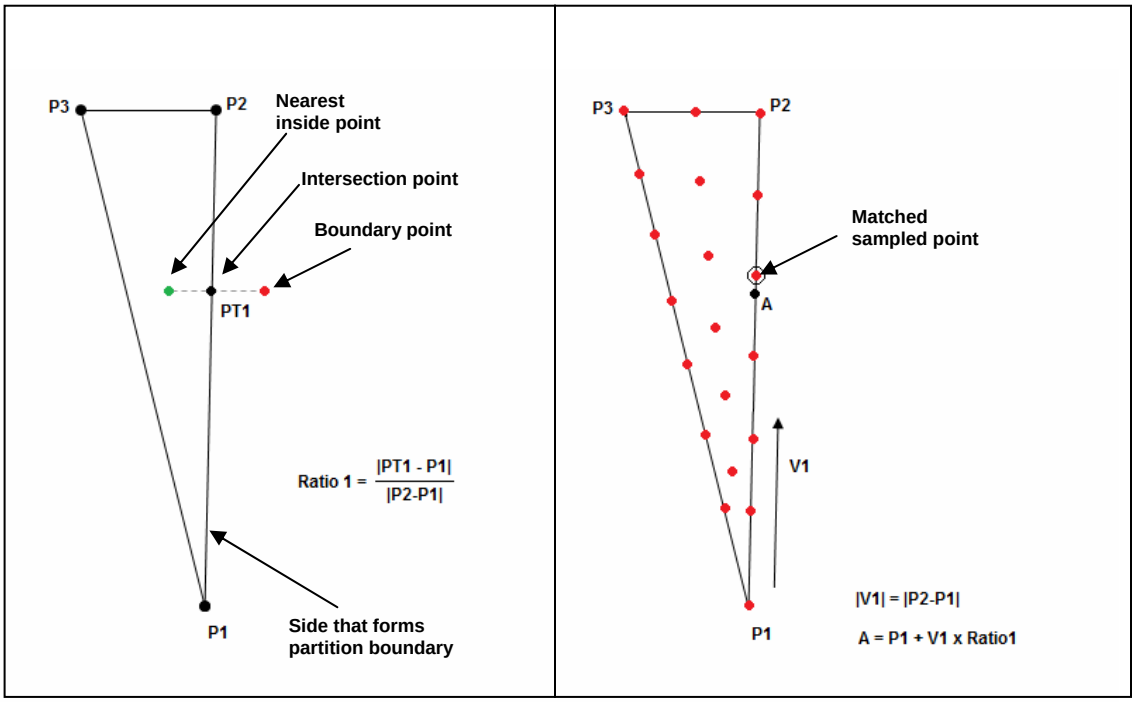


Figure 3.15: Matching a boundary point to an existing sampled point

3.5.3 Corners

From Figure 3.10a and b above, it is clear that the corners of the partition are excluded. The corner points for each partition are already known. They are identified in the partitioning step because they are defined as the start and end points of a partition boundary line. After the inside and boundary points have been matched to sampled points, the corner points are added. The corner point is compared to all grid locations to determine which is closest. The corner point is then associated to that position in the grid. If the position is already matched to a sampled point, the corner point takes

precedence and replaces the existing matched point. Corner points are part of the boundary point set. Figure 3.16 shows an example where the corner point matches to a grid location that was not matched to an inside or boundary point, following the example from Figure 3.10 above.

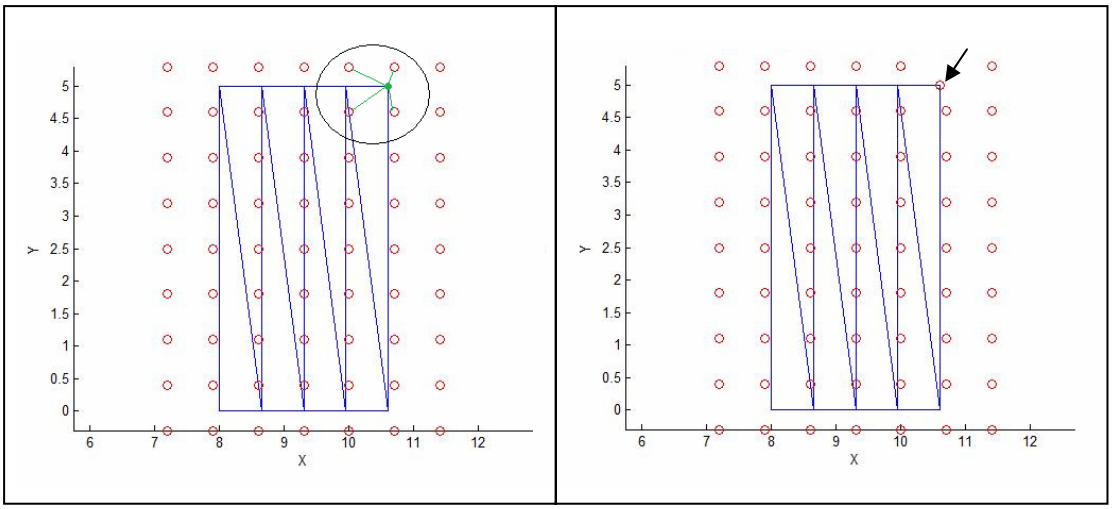


Figure 3.16: Adding corner points

After the corner points have been added, an initial quality check is run on the grid. This quality check is designed to eliminate very thin elements which can cause errors in future steps. The quality check simply measures the distances between points and merges any that would result in an element with a side length less than one quarter of the desired grid size. Corner points are preserved over other boundary points, and boundary points are preserved over inside points. If an inside point is merged with a boundary point, the value at the location in the inside point matrix is put to zero, and the corresponding location in the boundary point matrix takes the number of the matched point on the boundary.

Before matching boundaries, the grids are written to an "element" format, which means that a group of points is specified as a four-point quadrilateral or three-point triangular face. By defining elements this way, changes in geometry can more easily be propagated, and continuity is defined for boundary matching and quality checks later. The individual partitions are then reconstructed in the boundary matching step to form a continuous mesh for the entire surface.

3.6 Boundary Matching

The mesh patterns on adjacent boundaries will likely not match perfectly due to different orientations, distortion from flattening, etc. Therefore, meshes on the partitions must be connected together by matching up points on the partitions' shared boundaries.

Boundaries that have the same number of points from each partition, as in Figure 3.17a, are matched directly by combining corresponding points. The coordinates of corresponding points are averaged, and the averaged location is matched to a new sampled point on the boundary. The resulting boundary is shown in Figure 3.17b. The procedure is the same for all boundary points except for the first and last points. The first and last points on a boundary are "corner" points. They will either be at the intersection of multiple partitions or will define the boundary of the original surface, so they are important to preserve.

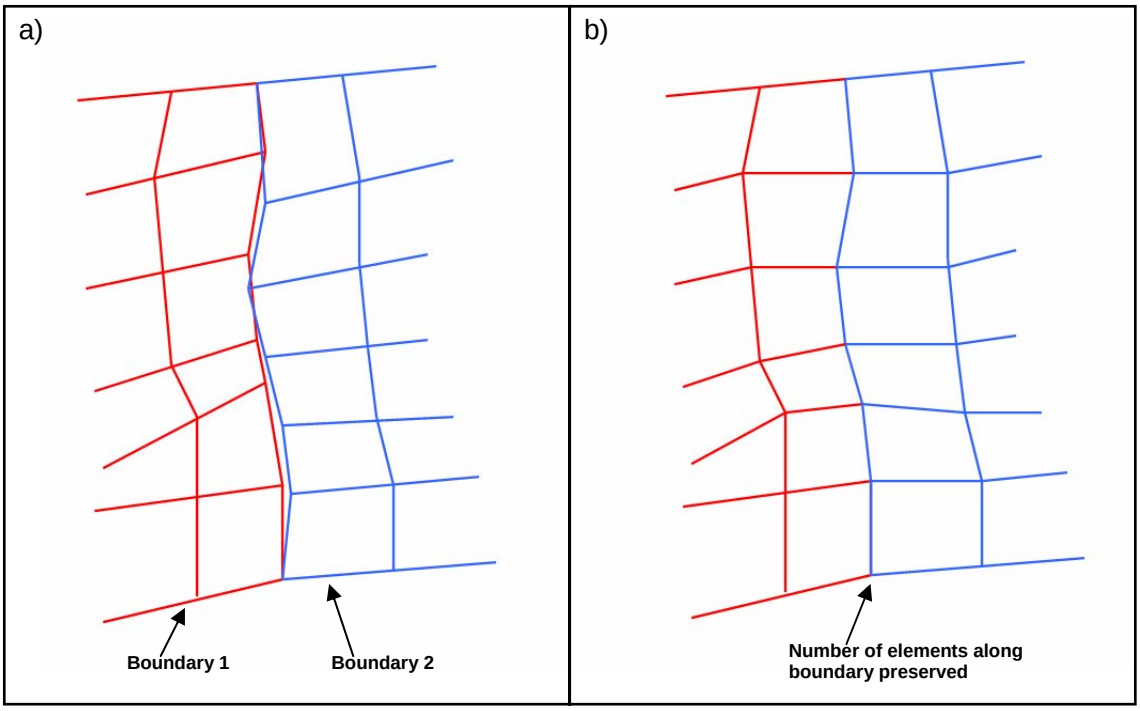


Figure 3.17: Matching boundaries with equal numbers of elements

If boundaries have unequal numbers of points from each partition, as in Figure 3.18a, some points will have to be combined so that the boundaries can be completely connected. This will result in triangular elements being formed at some locations at the boundary. The example in Figure 3.18b shows the triangles that have been created in order to match the boundaries. The points that are combined together, which form the triangular elements, are selected in order from the beginning of the boundary. So, if Boundary 2 has six points and Boundary 1 has seven points, the first and second points from Boundary 1 will be combined with the first point of Boundary 2. If Boundary 2 has six points and Boundary 1 has eight points, as in the example in Figure 3.18, then the first and second points of Boundary 1 will be combined with the first point of Boundary 2, and the third and fourth points of Boundary 2 will be combined with the second point in Boundary 2. The remaining points will be matched one-to-one. The points are selected

this way to keep triangular elements near boundary corners, leaving large areas in the middle with only quadrilateral elements. For real applications, having triangular elements near edges of boundaries makes them more accessible for removal, if desired.

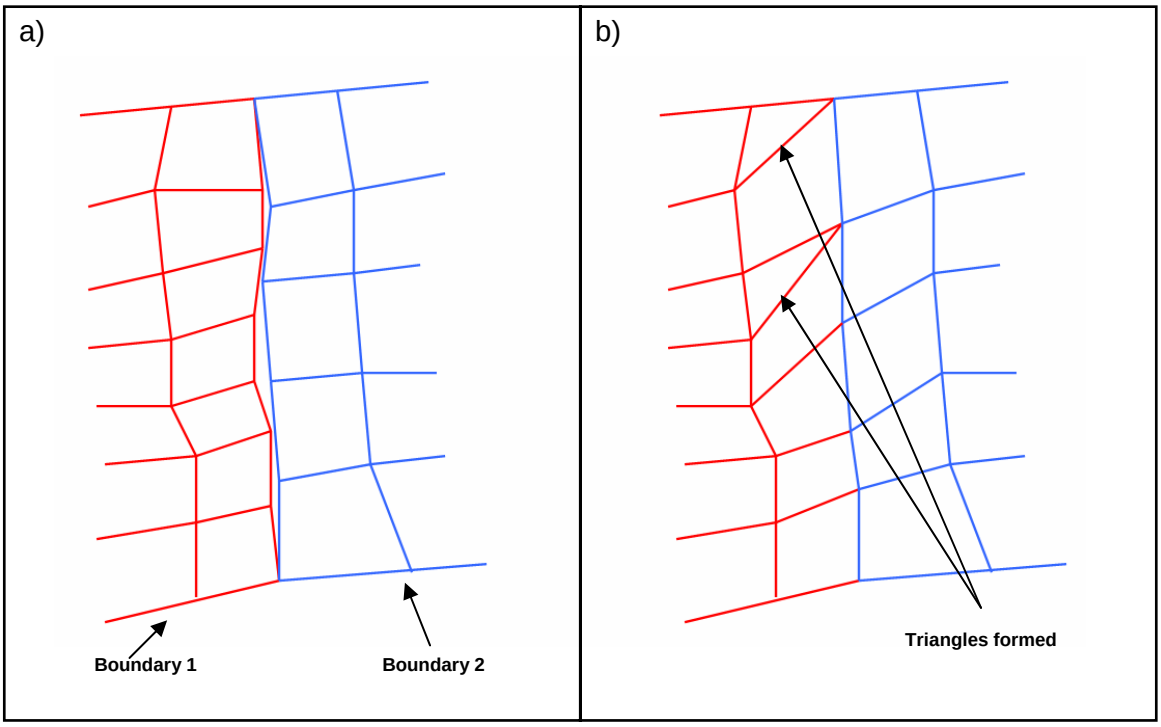


Figure 3.18: Matching boundaries with unequal numbers of elements

After the boundaries have been matched, a quality check is performed to improve element quality. This quality check provides fixes for triangular and quadrilateral elements with three vertices that are nearly co-linear and for triangular and quadrilateral elements with side lengths less than a quarter of the desired length. For both quadrilateral and triangular elements, vertices are considered nearly co-linear if they lie on edges that are 135 degrees or more apart. An angle of 135 degrees is used because it is a general rule of thumb for finite element quality, being halfway between the ideal of 90 degrees

(for a quadrilateral) and the unacceptable value of 180 degrees. Figure 3.19 shows the mesh operations.

Figure 3.19a shows an example of a triangular element with nearly co-linear vertices. The fix for this type of element is to break the adjacent element in two, as shown in Figure 3.19b, and to eliminate the original bad element. The resulting mesh is shown in Figure 3.19c.

Figure 3.19d shows an example of elements that have an edge length less than one quarter of the desired edge length. This is not a standard finite element quality criterion. It was determined for the application of cellular materials that elements with short edges should be eliminated. As shown in Figure 3.19e, a triangular element with a short edge will be eliminated, and a quadrilateral with a short edge will be converted to a triangular element. The corrected mesh is shown in Figure 3.19f.

Quadrilaterals with three of its four vertices nearly co-linear are also corrected as shown in Figure 3.19g. The bad element and adjacent element are both broken. The lines of the break and the resulting corrected mesh are shown in Figure 3.19h and Figure 3.19i, respectively. This fix results in the creation of two new triangular elements.

Once a continuous grid is calculated for the input surface, a corresponding offset mesh is needed to create three-dimensional elements.

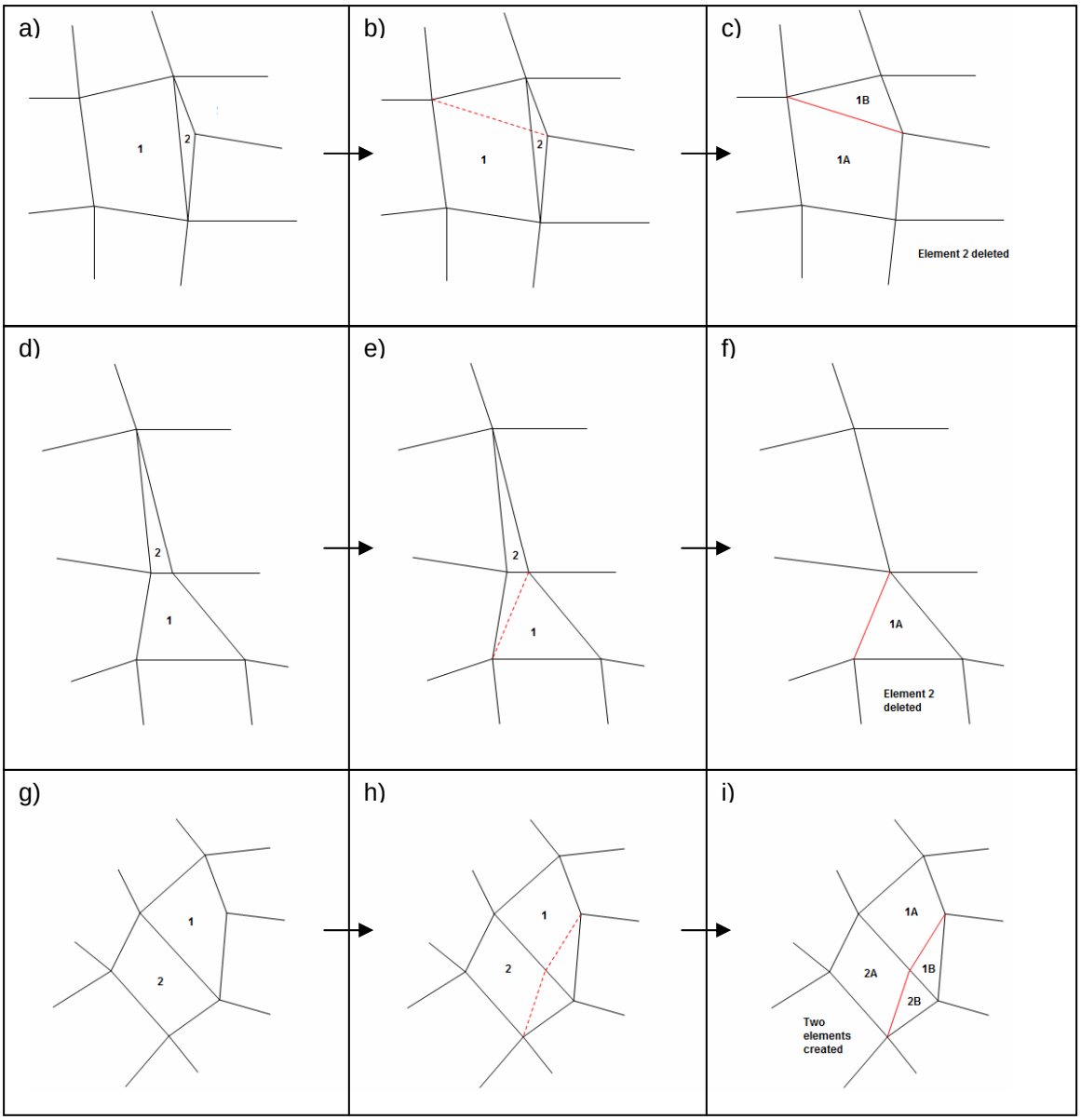


Figure 3.19: Element quality checks

3.7 Offsetting

The offsetting function creates a new surface that is a minimum uniform distance from the original surface at all points. Figure 3.20 shows an example of a two dimensional case of offsetting.

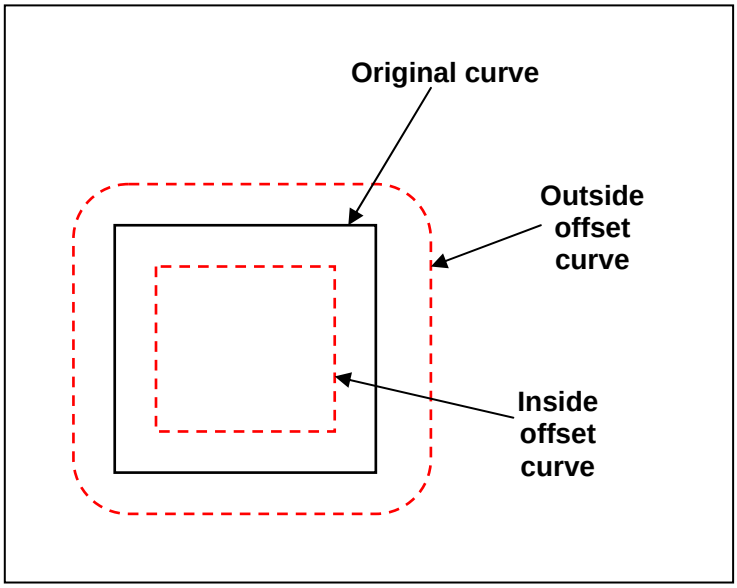


Figure 3.20: Inside and outside offset of a square

Because the input for the algorithm is a surface, an additional surface is required to specify the volume that will eventually be filled with cellular material. A continuous three-dimensional layout of cellular structure cannot be defined without the specifications of the boundary volume. Two different methods are explored for offsetting the surface.

The first method is similar to the method presented in [14]. A cloud of points is created by displacing each sampled point the desired amount in a direction normal to the surface at that point. After all points are offset, some are eliminated because they are not the full offset distance away from all points. The offsetting procedure is efficient, but the

process of eliminating erroneous offset points is very time consuming, even using the technique developed in [14].

For this method, the sampled points on a face are offset in the direction of the face's normal vector, as shown in Figure 3.21. These points are not necessarily on the offset surface. That is determined independently of the point offsetting. The sampled points on a convex line are offset in a cylindrical array. Sampled points on concave lines are not offset at all. Concave and convex lines are shown in Figure 3.22, with α less than 180 degrees. The offset points for a convex line are shown in Figure 3.23 below. The vertices are offset in a hemispherical array for a convex point, as shown in Figure 3.24. The idea of a convex point is a direct translation from the definition of a convex line.

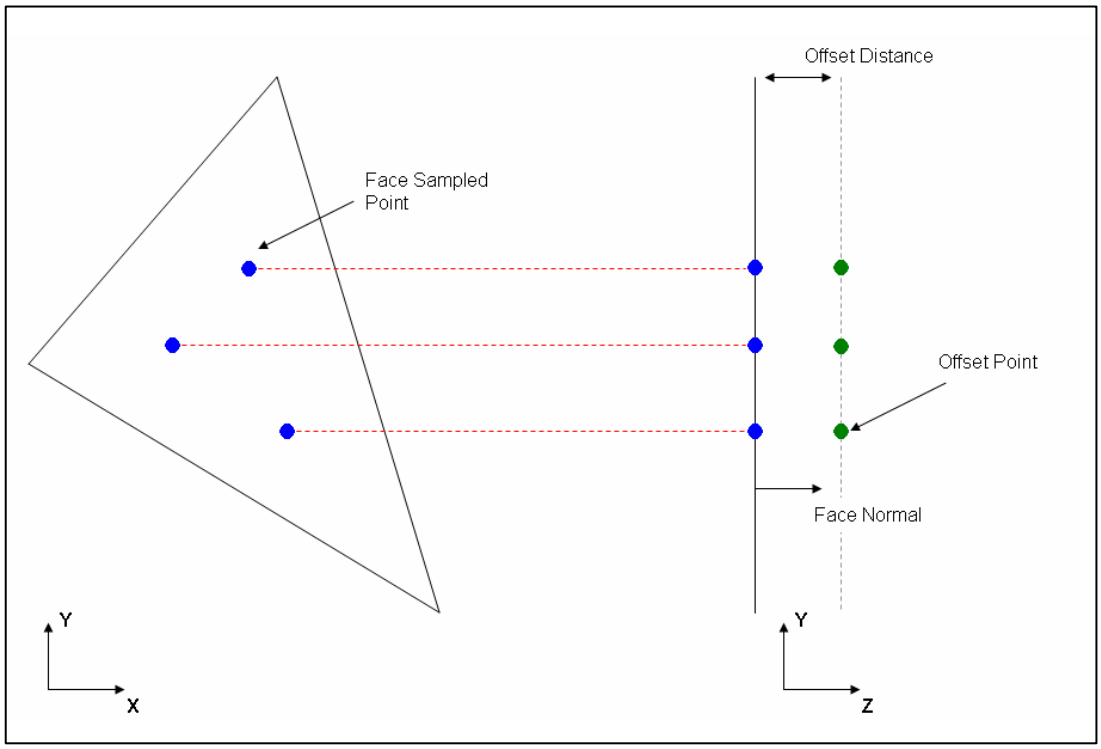


Figure 3.21: Offsetting face sampled points

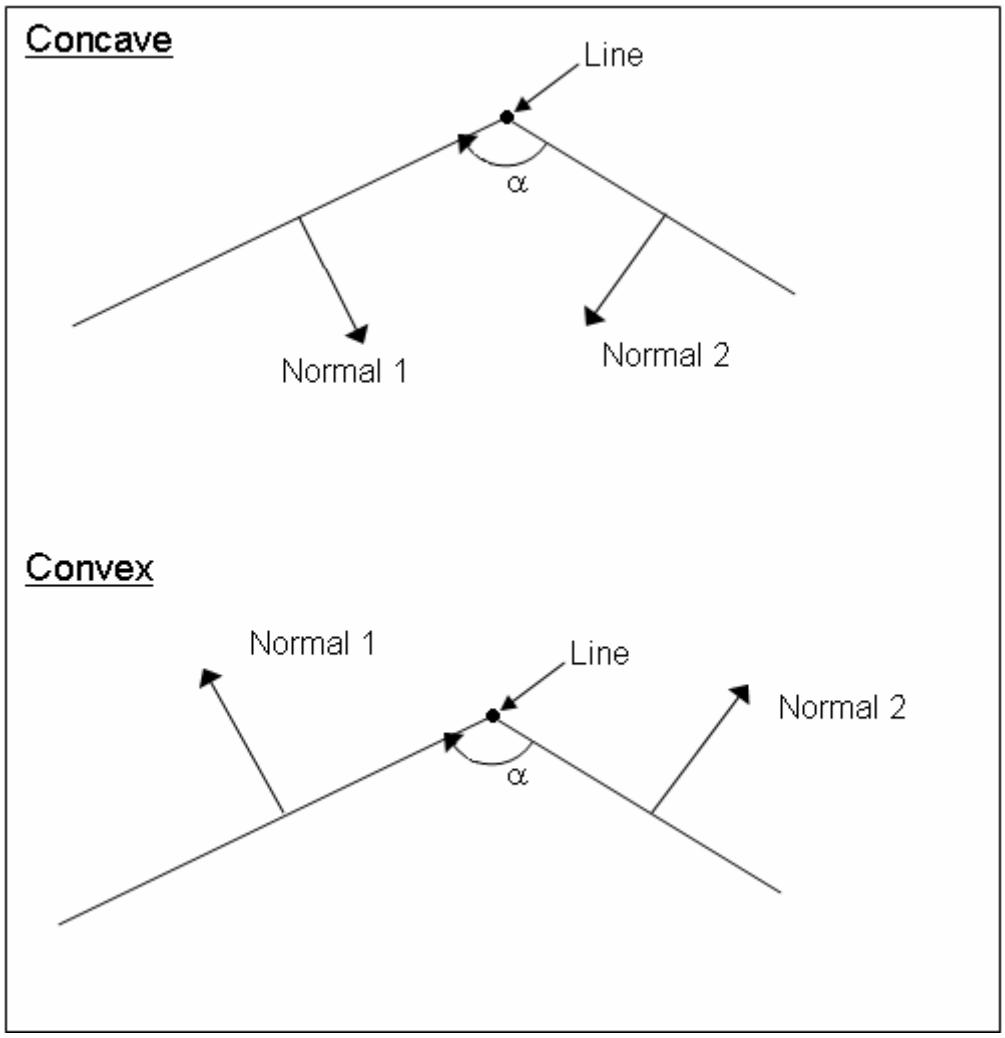


Figure 3.22: Concave and convex lines

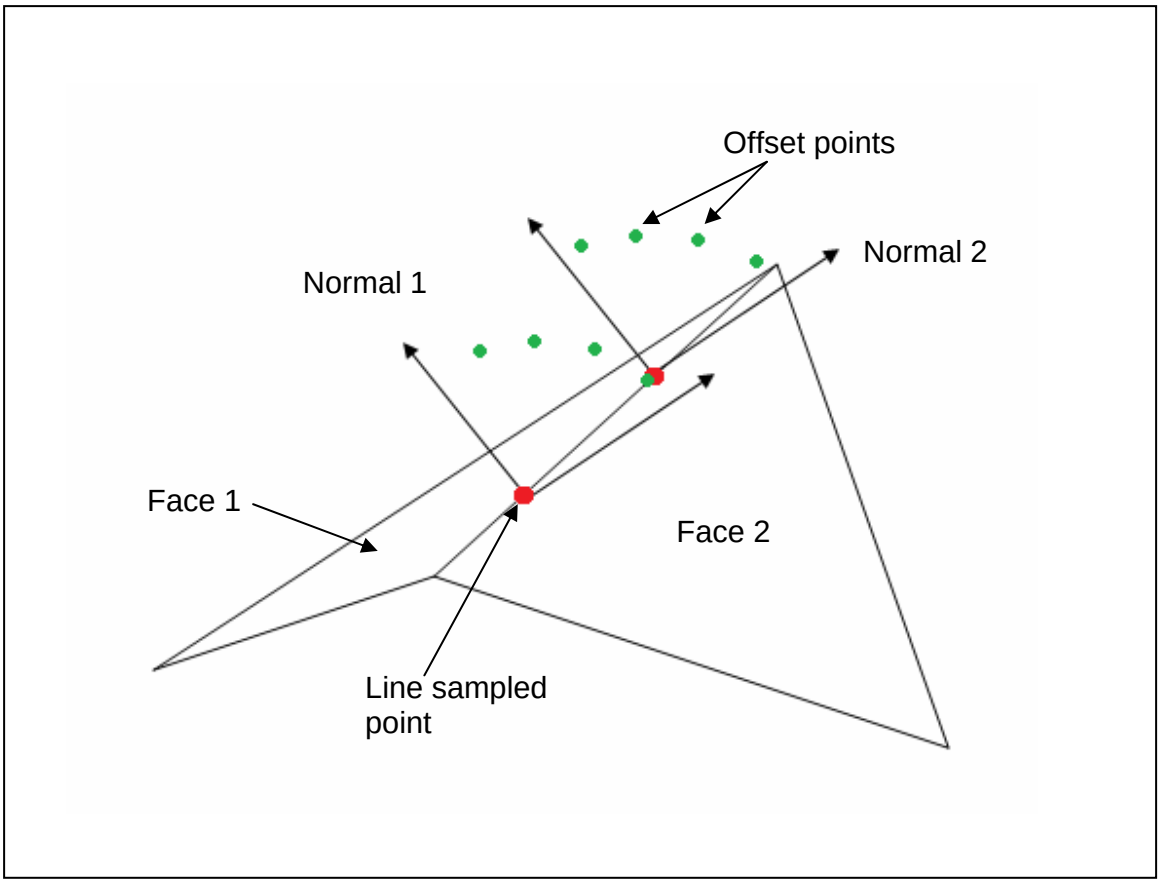


Figure 3.23: Offsetting line sampled points

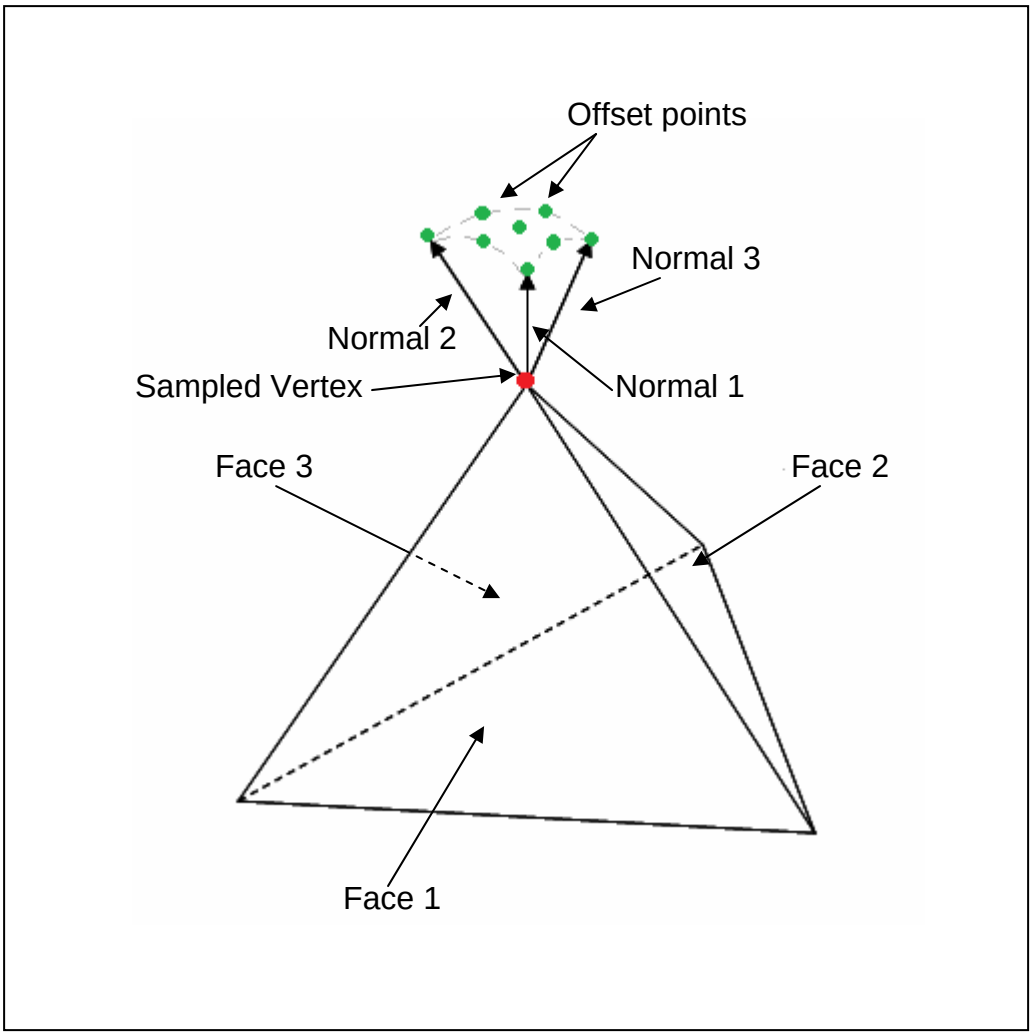


Figure 3.24: Offsetting vertex sampled points

A second method of offsetting the surface was developed to improve efficiency. This process offsets only the vertices of the mesh elements on the input surface. Each vertex is given an initial offset from the input surface at that point. If the vertex is a sampled point from a face, it is offset in the direction normal to the face. If the vertex is a sampled point from a line, it is offset in the direction that is the average of the two faces that intersect at that point. If the vertex is a sampled point from a vertex of the original input surface, then it is offset in a direction that is the average of all faces that intersect at that vertex. The initial offset position is then checked against all sampled points. If it is the full offset distance away from all sampled points, the process is complete. If the point is not at least the full offset distance away from all points, the nearest sampled point is found, and a new location for the offset point is calculated to push the point slightly away from that sampled point. The process is repeated to check this new point, and the point will continue to be moved away from the nearest sampled point until it is at least the full offset distance away from all sampled points. Figure 3.25 illustrates an example of this more efficient method of offsetting with a sampled point from a line between two faces.

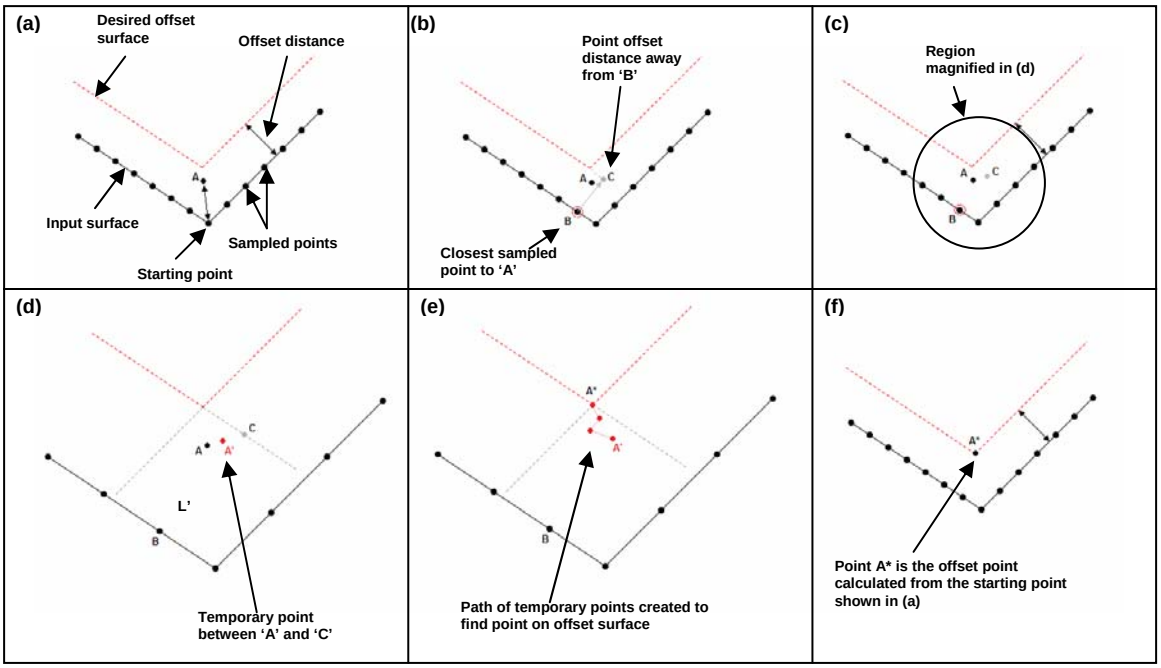


Figure 3.25: More efficient offsetting

Figure 3.25s shows the original input surface with sampled points and the theoretical offset surface (red). Point A is the offset point from the corner, but it is not on the offset surface. Figure 3.25b shows that point B is the closest sampled point. Point C is the location that is the offset distance from Point B normal to the original surface at B. Figure 3.25c shows the region magnified for parts d through f of the figure. Figure 3.25d shows a temporary point that is created that is moved away from the closest sampled point in an attempt to get closer to the offset surface. Figure 3.25e shows the multiple temporary points created working toward the offset surface. This method returns a result much faster than the first method.

The benefit of the first method is that there are more points of reference for determining which points are truly on the offset surface, producing a more precise result. Moving the calculated offset point as in the second method can lead to a slight error in the final position. However, the first method adds another degree of pixelation in the

final grid because each element vertex is matched to an offset point that may not be the result of offsetting the vertex's sampled point location. For the second method, each vertex location has an offset node specifically calculated for it, which generally leads to a better result.

3.8 Hexahedral Element Formation

Once the offset is complete, a three-dimensional volume is defined to hold the cellular structure. The volume must now be divided into small primitive volumes to hold individual cellular structure units.

The quadrilateral elements applied to the original surface form the bases of cubic primitives. A matching mesh must be produced on the offset surface to create the opposing face of the primitive. The definitions of the two opposing faces of a cube contain all eight corners needed to define the volume. Figure 3.26 shows a quadrilateral on the input surface projected onto the offset surface forming two opposing faces of a cube.

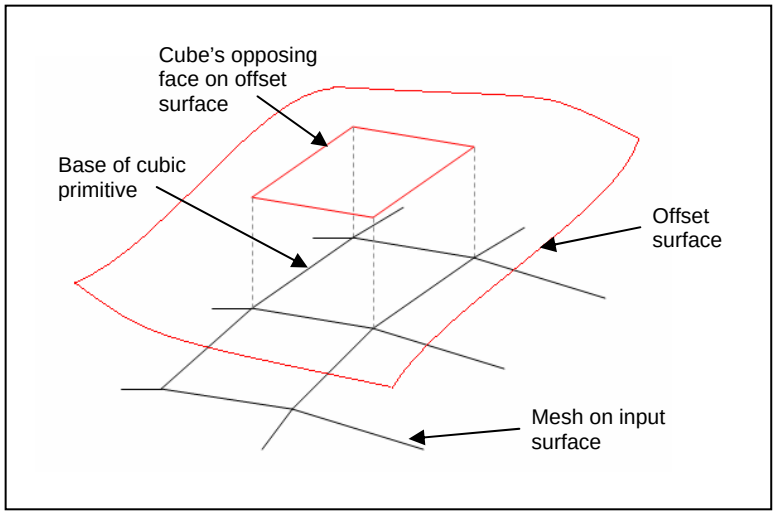


Figure 3.26: Quadrilateral element projected onto the offset surface

If the first method of offsetting is used, a large point cloud that defines the offset surface is created. To define the primitive's opposing face on the offset surface, each vertex of the primitive's base on the input surface must be matched to the nearest point in the offset point cloud. This procedure is another time-consuming element to the first offsetting method because each vertex has to be compared to each point in the offset point cloud to find the closest. Figure 3.27 shows a quadrilateral with a matching face on the offset surface created by the first method of offsetting.

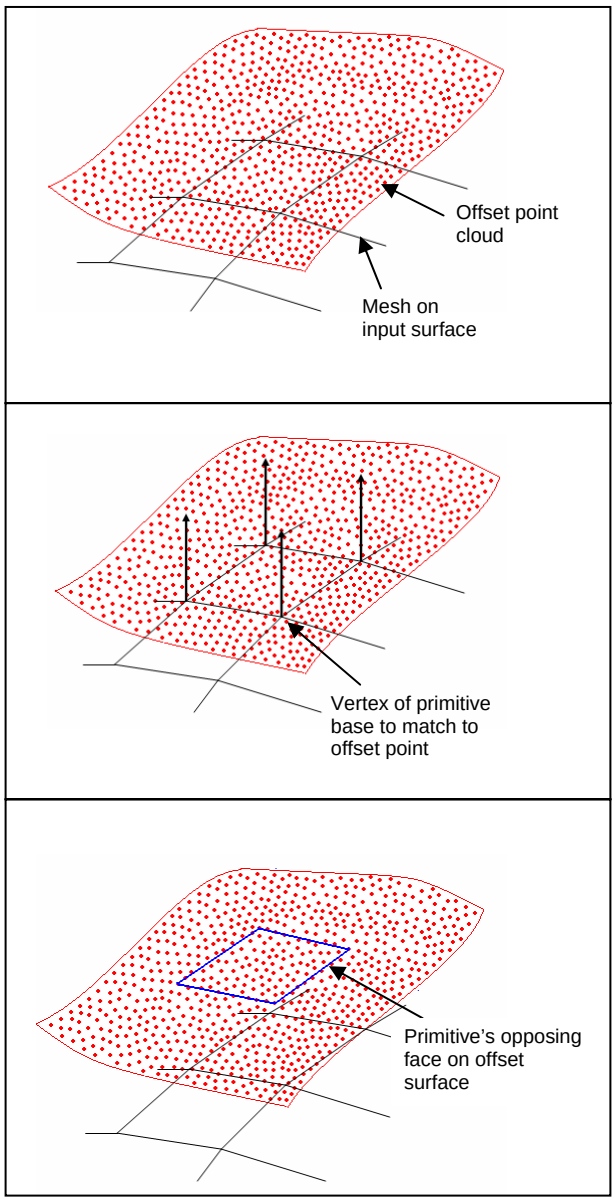


Figure 3.27: Matching a quadrilateral base to an offset point cloud

If the second method of offsetting is used, the mesh vertices do not have to be matched to offset points by searching through a set of points. Each offset point has been specifically created for a vertex of the mesh on the input surface and therefore is automatically matched.

When this step is completed, the vertices of the mesh on the input surface have been matched to points on the offset surface. The eight points for a cubic primitive or six points for a primitive formed by a triangular element (triangular prism) are all that is needed to describe the volume that is to be filled with cellular material.

3.9 Truss Definition

The last step is to insert a cellular structure into each three-dimensional element defined above. For the examples that follow, the cellular structure inserted for quadrilateral elements will be the truss element shown in Figure 3.28a. However, any element that fits into a cubic primitive can be inserted. All that is needed is a different definition for output file. A library of such input elements could easily be developed for reference.

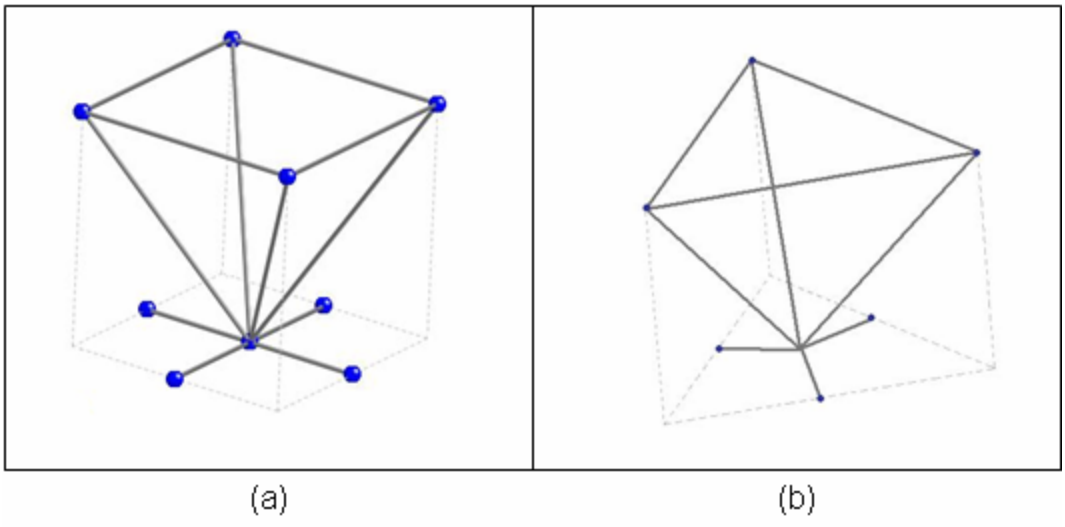


Figure 3.28: Cellular structure definition

Triangular prism elements must have a modified cell definition. One option for triangular prism elements that would accompany the cubic element for the following examples is shown in Figure 3.28b. The examples below use the elements in Figure 3.28.

Figure 3.29 shows an example of a primitive element and how the vertices coordinate to the truss unit cell shown in Figure 3.28.

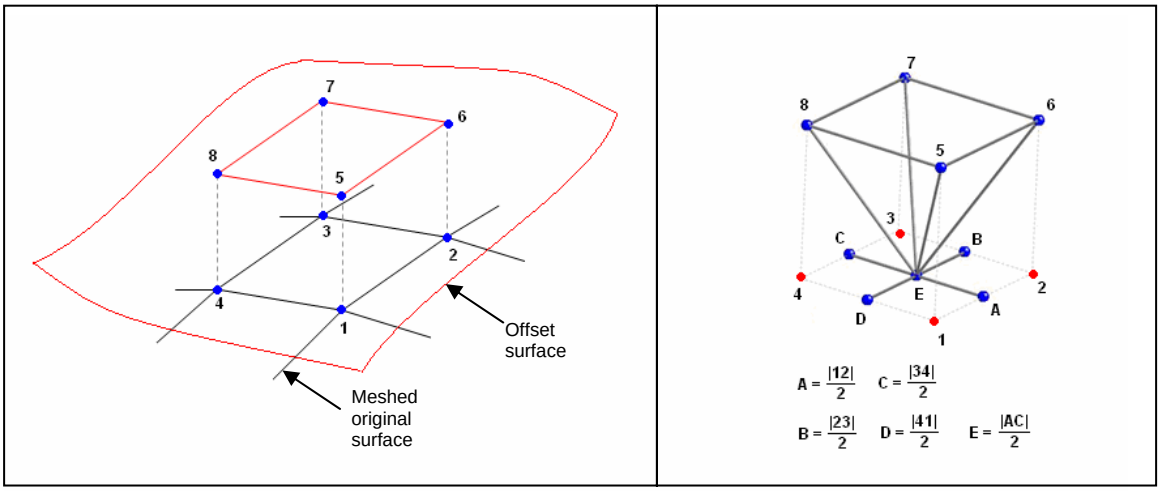


Figure 3.29: Truss structure formulation

3.10 Detailed Example

This detailed example is presented to show some of the individual steps of the algorithm described above in more detail. The performance of the algorithm will be addressed in the following chapters.

The surface geometry chosen for this example is a half cylinder. The input geometry is shown in Figure 3.30.

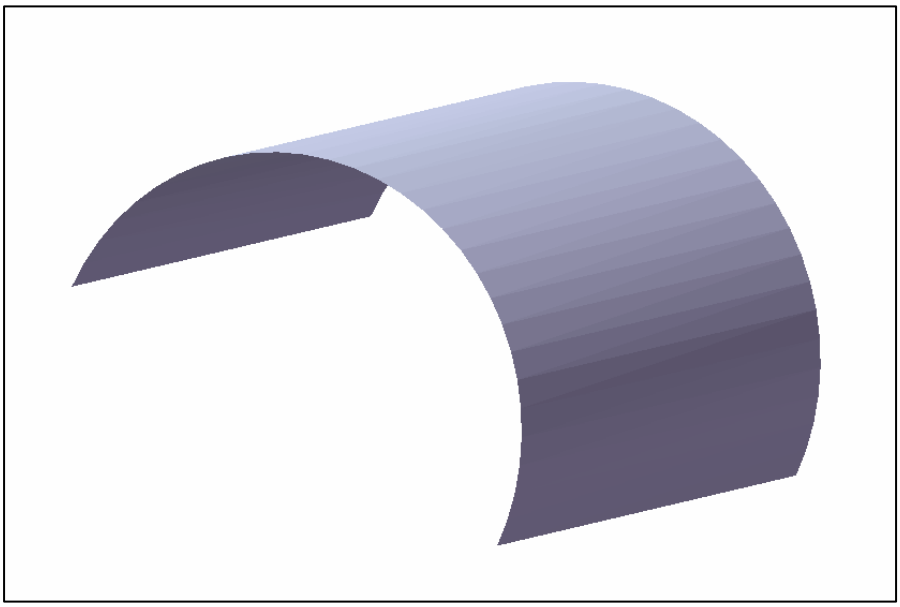


Figure 3.30: Detailed example - cylindrical input surface

The first step in the algorithm is to sample the input surface. The sampling density is a user input. The higher the input value, the greater the sampling density or the smaller the sampling distance. Figure 3.31 (note axis scale) shows one triangular face from the input surface with its sampled points for two different values of sampling density. Figure 3.31a shows the triangular face with a sampling rate of 20. Figure 3.31b

shows the same face with a sampling rate of 60. There are nearly three times as many sampled points on the surface in Figure 3.31b, which is expected because the sampling rate is three times higher.

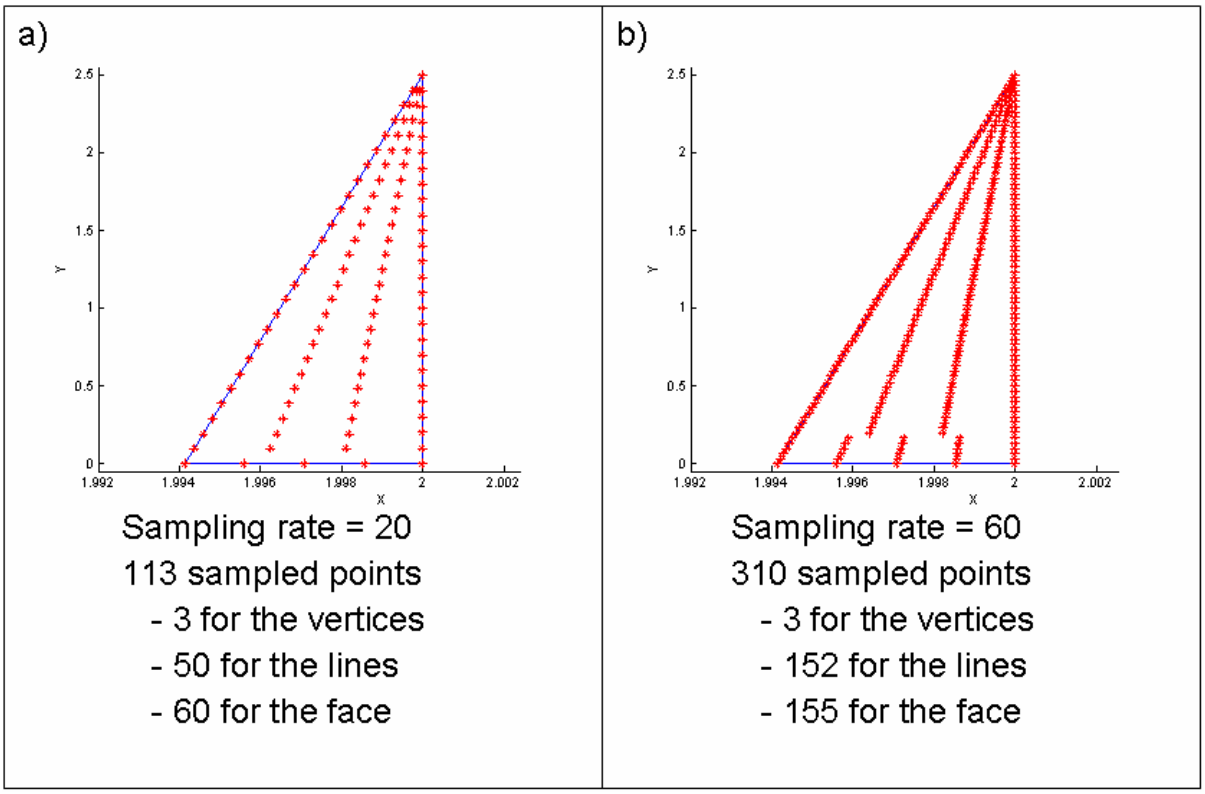


Figure 3.31: Detailed example - sampling example

The next step in the algorithm is to partition the input surface. The surface is partitioned based on an angle input that represents the angle between normal vectors of faces being compared. Figure 3.32 shows one way that the algorithm would calculate three partitions for this surface with an angle input of 0.87 radian. There is a range of values for the angle criterion that would produce three partitions. They would not all look exactly the same. For the rest of the example problem, the three partitions shown will be used.

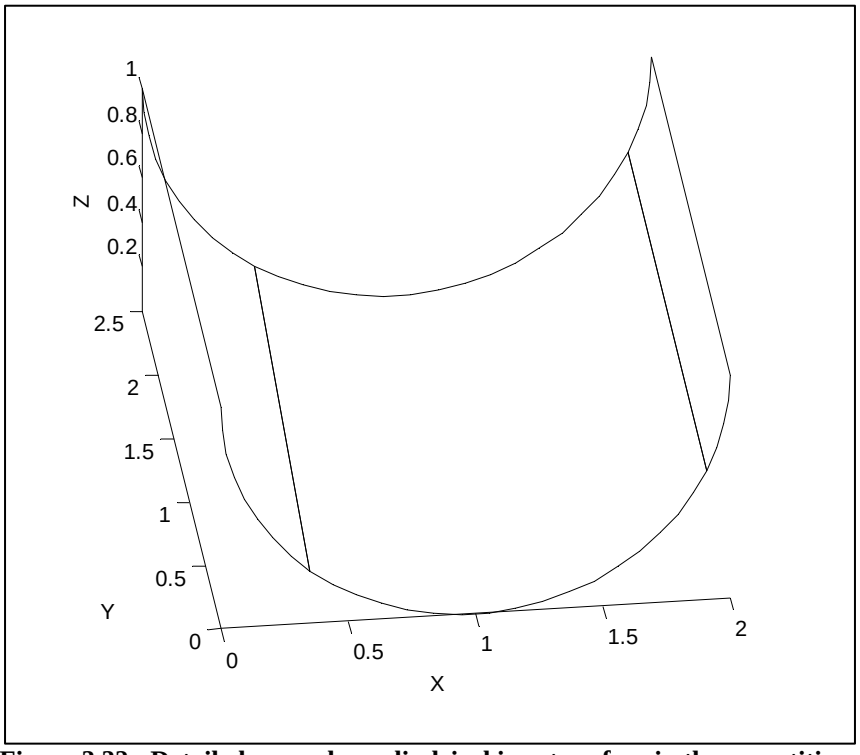


Figure 3.32: Detailed example - cylindrical input surface in three partitions

The next step is to flatten each partition separately. The partitions are reoriented before flattening. Figure 3.33, Figure 3.34, and Figure 3.35 show the partitions. The reoriented partition is shown on the right, and the flattened partition is shown on the left in each of the three figures.

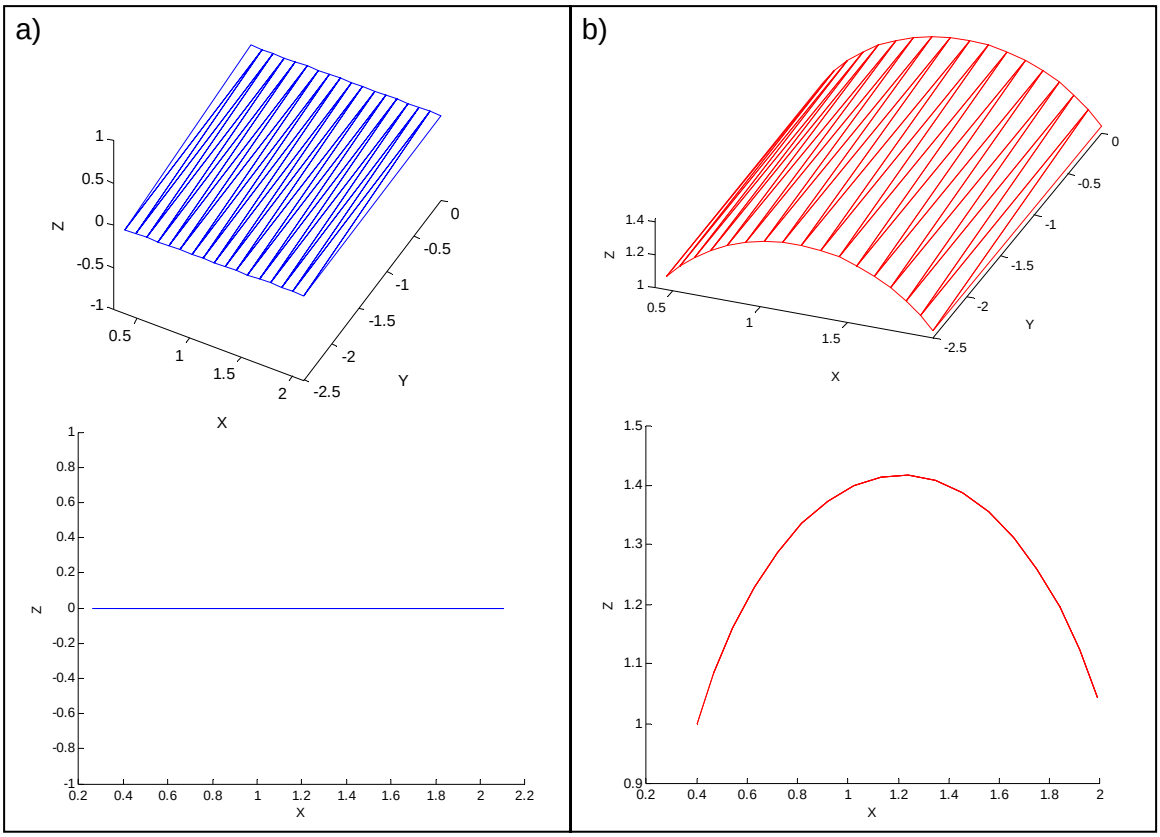


Figure 3.33: Detailed example - first partition

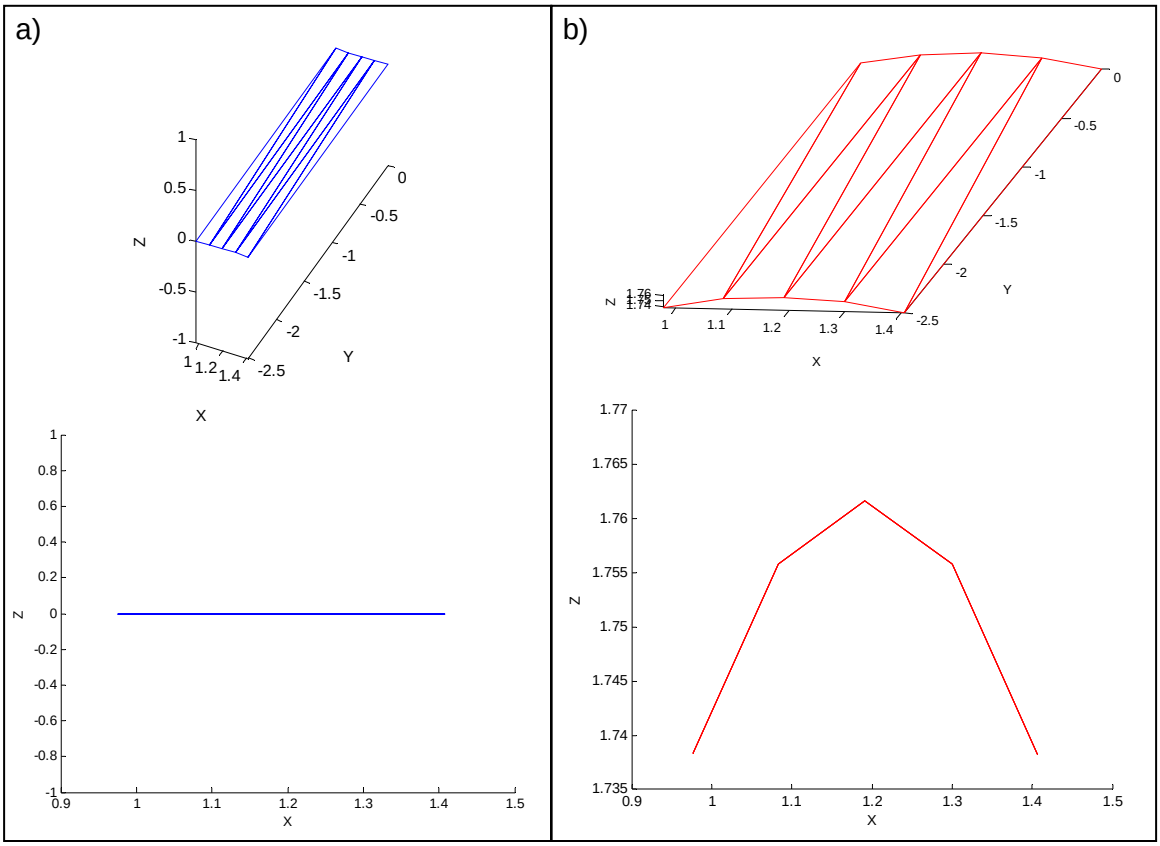


Figure 3.34: Detailed example - second partition

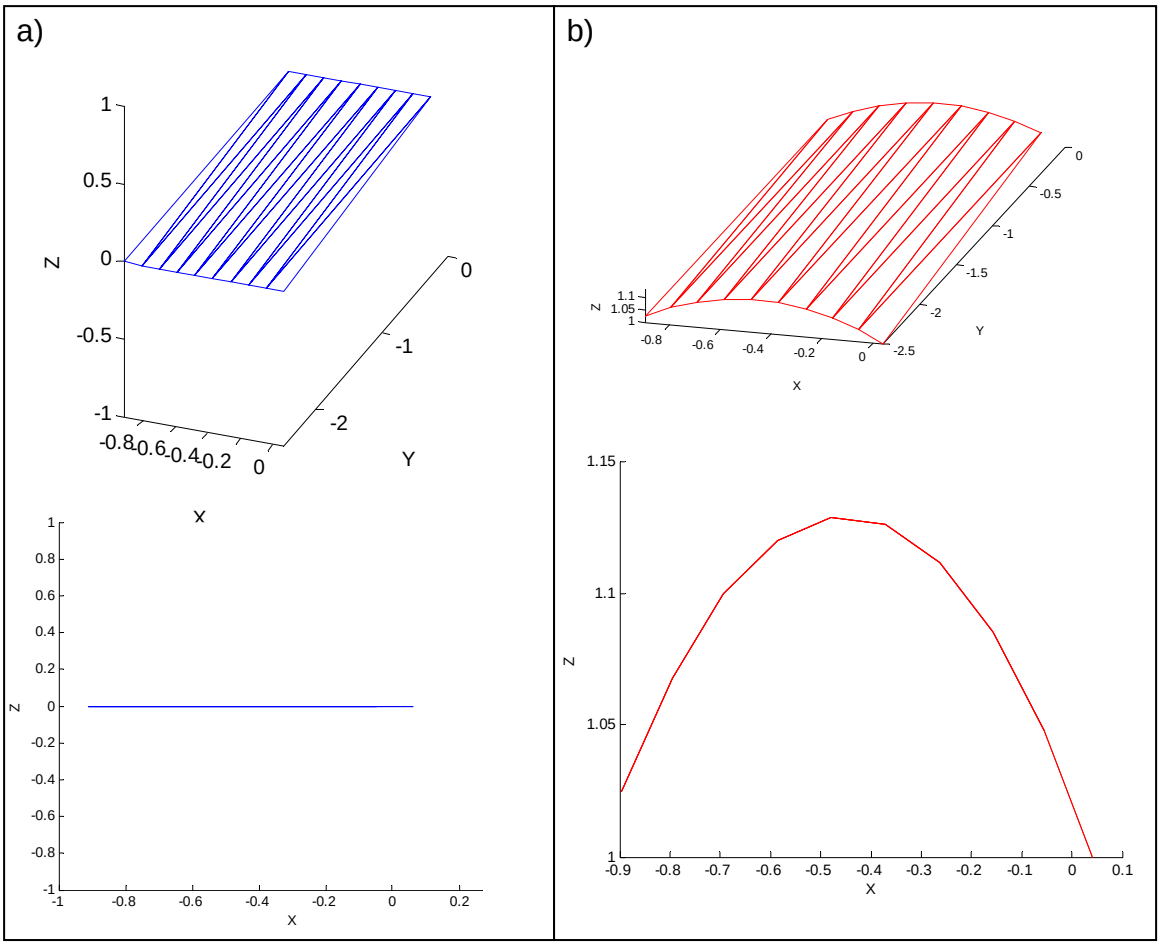


Figure 3.35: Detailed example - third partition

Once a partition is flattened, a grid is overlaid to calculate mesh. The flattened partition shown in Figure 3.35a is shown with a grid overlaid in Figure 3.36. The grid vertices within the partition boundaries are matched to sampled points and stored in the Inside Point Matrix, with each location in the matrix corresponding to a vertex of the grid. The Inside Point Matrix is shown in Figure 3.37. Some of the matrix locations will always be zero because the grid is purposefully made larger than the partition. The vertices of the grid that are matched to boundary points are determined by finding open positions in the Inside Point Matrix adjacent to filled positions. These vertices are matched to points on the partition boundary, and corner points are also added. These points are stored in the Boundary Point Matrix shown in Figure 3.38. In this example problem, some inside vertices are very close to the partition boundary and may be matched to a sampled point on the boundary. This can cause the same sampled point to be matched to two different locations in the point matrices. The way this is handled when forming elements is shown below.

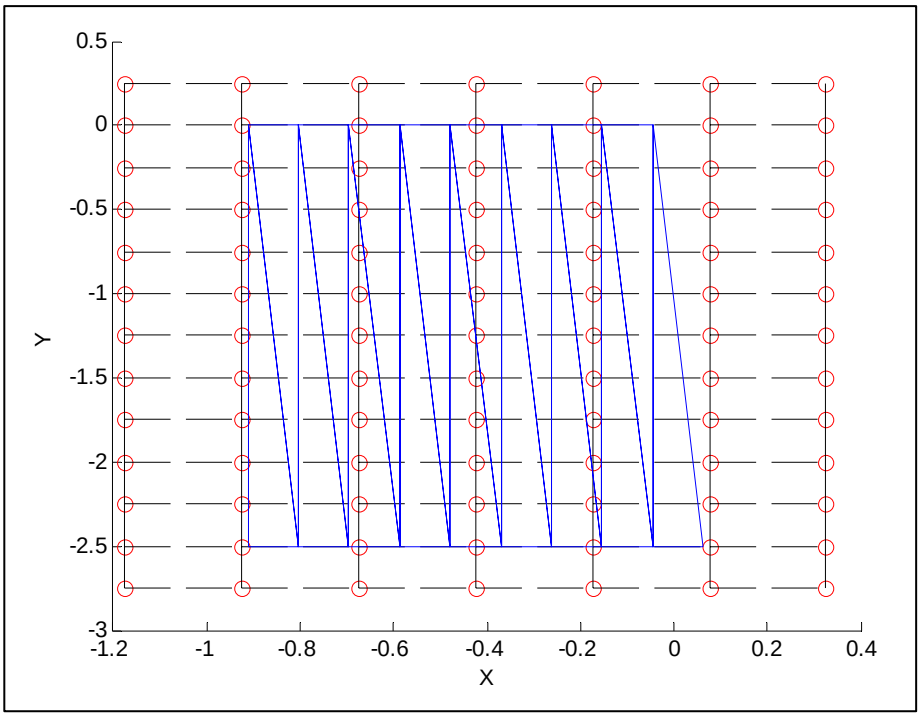


Figure 3.36: Detailed example - partition three flattened with grid overlaid

0	0	0	0	0	0	0
0	0	0	0	3699	0	0
0	0	82823	75853	69412	0	0
0	0	82830	75860	3608	0	0
0	0	82838	75868	69200	0	0
0	0	82845	75875	69215	0	0
0	0	82853	3895	69231	0	0
0	0	82861	75112	69247	0	0
0	0	82868	75126	69261	0	0
0	0	4182	75142	69277	0	0
0	0	4175	75156	69291	0	0
0	0	4167	3856	0	0	0
0	0	0	0	0	0	0

Figure 3.37: Detailed example - Inside Point Matrix

0	0	0	0	3699	0	0
0	0	4317	4008	0	42	0
0	4620	0	0	0	3305	0
0	4612	0	0	0	3298	0
0	4605	0	0	0	3290	0
0	4597	0	0	0	3283	0
0	4590	0	0	0	3275	0
0	4582	0	0	0	3267	0
0	4575	0	0	0	3260	0
0	4567	0	0	0	3252	0
0	4560	0	0	0	3245	0
0	59	0	0	3856	43	0
0	0	4167	3856	0	0	0

Figure 3.38: Detailed example - Boundary Point Matrix

The inside and outside point matrices are combined so that elements can be written for the partition. The combined matrix is shown in Figure 3.39. Starting in the upper left corner and working across and down, elements are written by grouping together four adjacent matrix locations. Figure 3.40 below shows three examples of elements being formed.

0	0	0	0	3699	0	0
0	0	4317	4008	3699	42	0
0	4620	82823	75853	69412	3305	0
0	4612	82830	75860	3608	3298	0
0	4605	82838	75868	69200	3290	0
0	4597	82845	75875	69215	3283	0
0	4590	82853	3895	69231	3275	0
0	4582	82861	75112	69247	3267	0
0	4575	82868	75126	69261	3260	0
0	4567	4182	75142	69277	3252	0
0	4560	4175	75156	69291	3245	0
0	59	4167	3856	3856	43	0
0	0	4167	3856	0	0	0

Figure 3.39: Detailed example - combined boundary and inside points

Figure 3.40a shows an example of four matrix locations that are used to form an element. This particular element is on the partition boundary. Only three of the four locations have matched sampled points. Therefore, only a triangular element could be formed. However, in this case two of the locations have been matched to the same boundary point because of the proximity of the inside point to the boundary. Because only two unique sampled points are present in this group of four, no element is written for this case.

Figure 3.40b shows another example of a group of four matrix locations. In this case, three unique sampled points are identified, which form a triangular element. This element is identified in Figure 3.41 with the letter ‘b’. Using the same technique of grouping four adjacent matrix locations, three quadrilateral elements would be expected after the triangular element in this row. It can be seen in Figure 3.41 that three quadrilateral elements are indeed formed in that row.

Figure 3.40c shows a final example of element formation. In this case, there is again a duplicated point. There are only three unique sampled points, so again a triangular element is produced. The resulting element is identified in Figure 3.41 with the letter ‘c’.

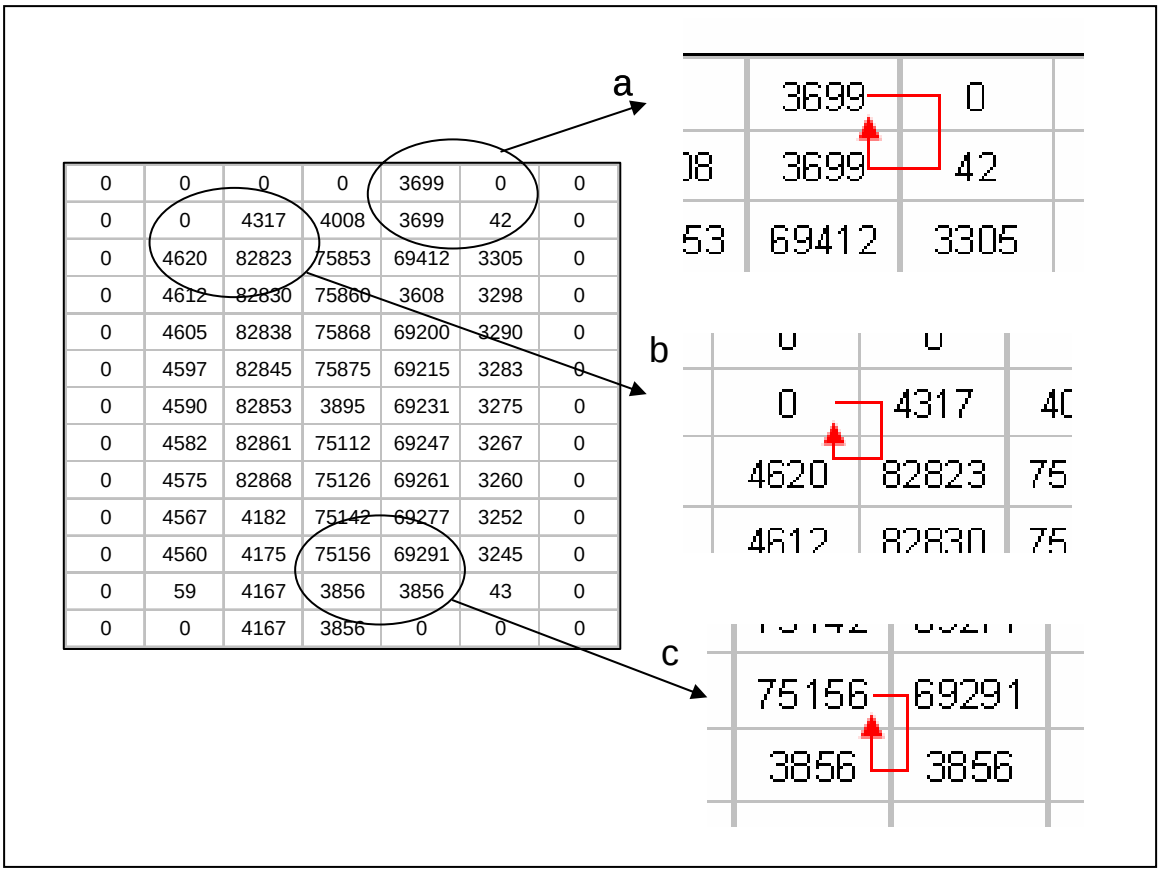


Figure 3.40: Detailed example - element formation examples

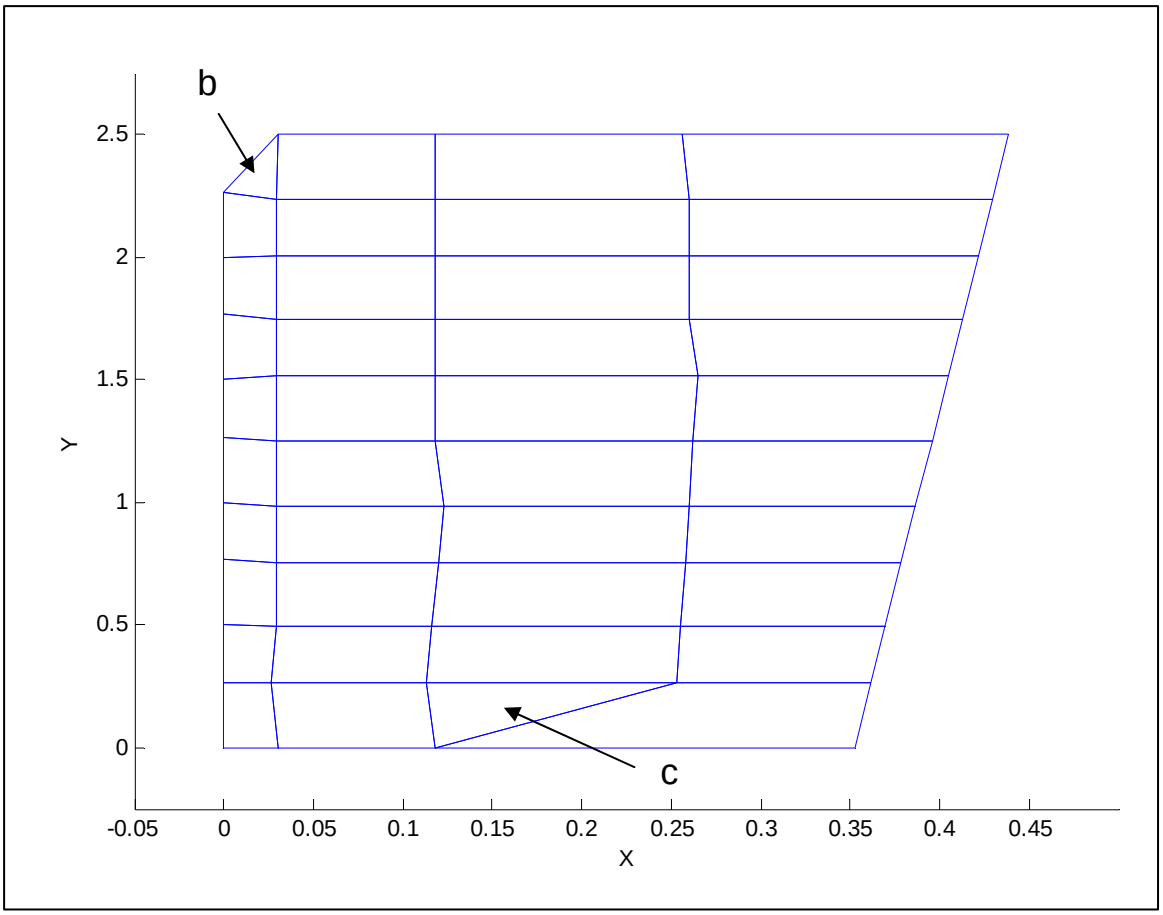


Figure 3.41: Detailed example - initial mesh on the third partition

Mesh is formed the same way on all partitions. The initial mesh for the three partitions before quality checks or boundary matching is shown in Figure 3.42. The boundary between the first and third partitions looks well matched already. The boundary between the first and second partitions is mismatched. A closer look at the boundary between the first and second partitions is shown in Figure 3.43.

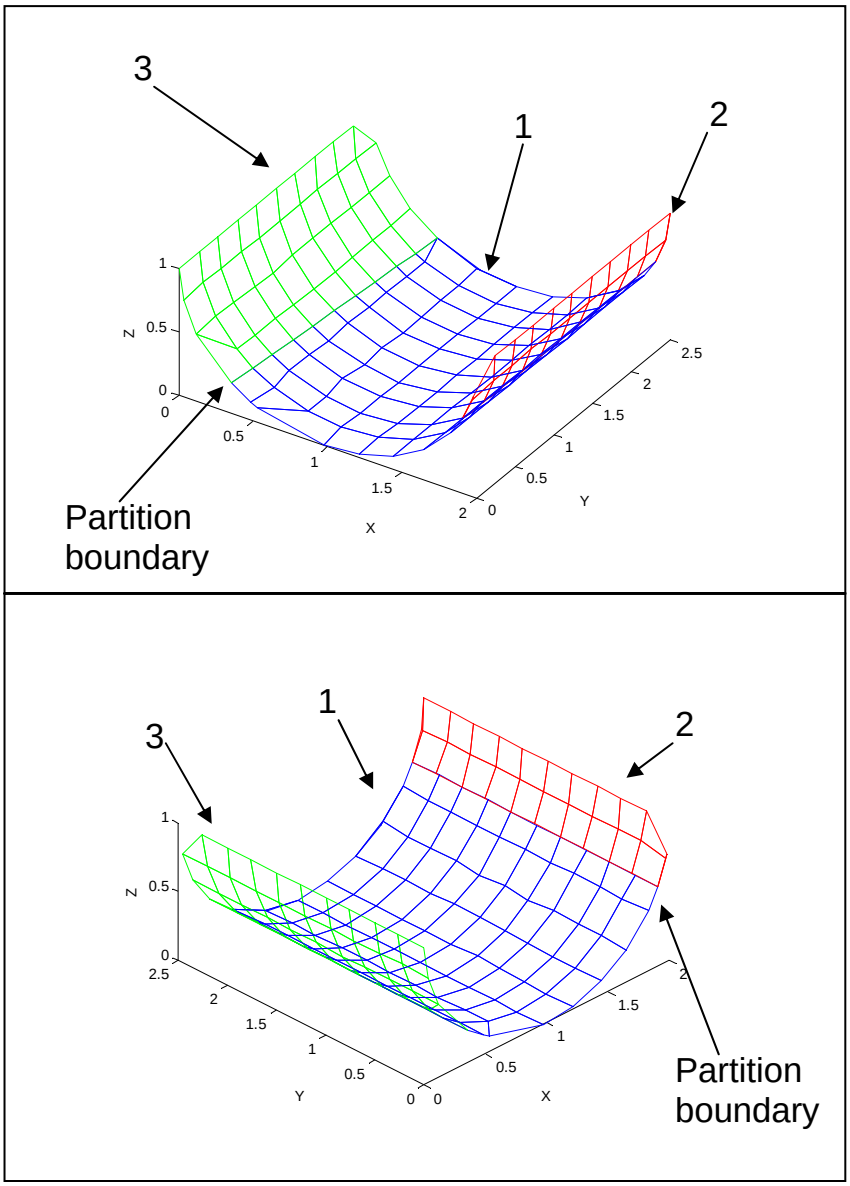


Figure 3.42: Detailed example - initial mesh

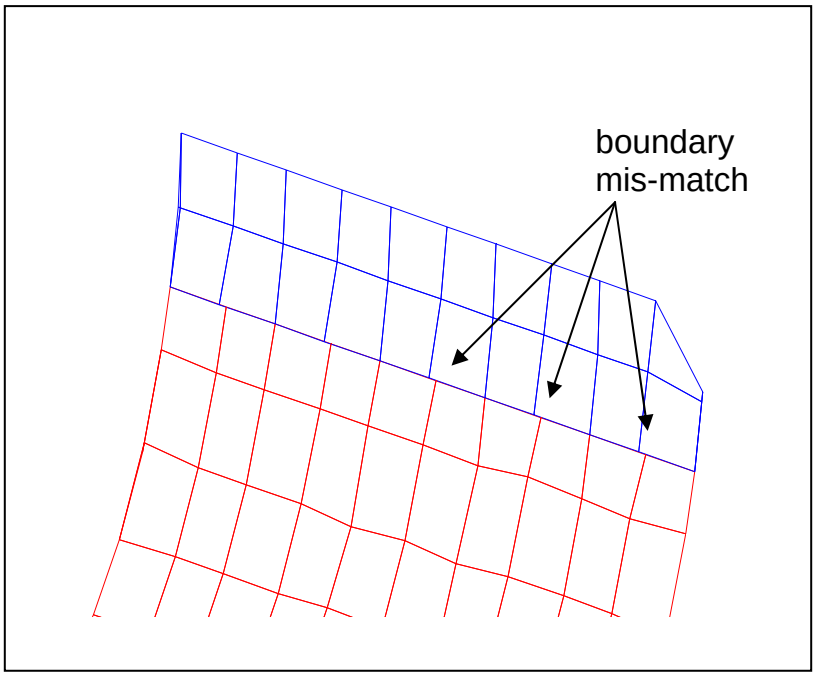


Figure 3.43: Detailed example - mismatched partition boundary

At this point, an initial quality check is performed. This quality check only tests element side lengths. If a side is less than one quarter of the requested element size, then it is collapsed. If this occurs on a quadrilateral element, a triangular element is formed. If it occurs on a triangular element, the element is deleted. Once the quality check is complete, the boundaries are matched. Figure 3.44a and Figure 3.45a below show initial mesh before quality checks and boundary matching operations are performed.

Figure 3.44a shows the mismatched boundary between the first and second partitions and some bad elements on the second partition. The encircled area in Figure 3.44a on the left highlights two very thin triangular elements. The encircled area on the right shows a quadrilateral with one very short side. Figure 3.44b shows the matched boundary and corrected elements. The triangular elements have been deleted, and the quadrilateral element has been reduced to a triangular element.

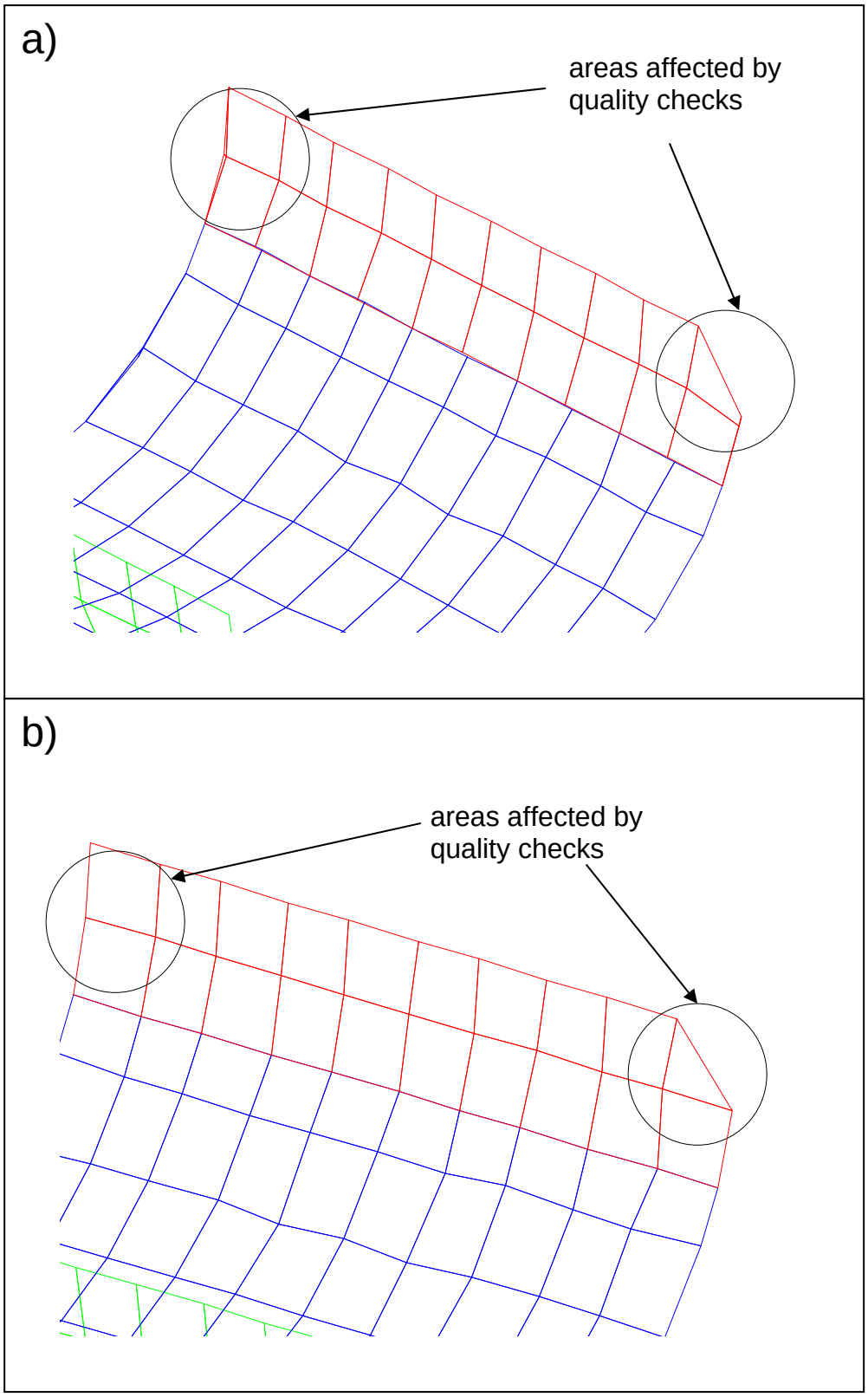


Figure 3.44: Detailed example - boundary match and quality check for partition 1/2

Figure 3.45a shows the boundary between the first and third partitions and some bad elements on the first partition. The boundary is already matched simply because the grids matched to the same sampled points when the initial mesh was calculated. In Figure 3.45a, the upper encircled area shows a triangular element and a quadrilateral element that share a short side. The lower encircled area shows a quadrilateral that has one short side. Figure 3.45b shows the corrected elements. In the case of a triangular element and quadrilateral element sharing a short side, the triangular element is deleted, and a new triangular element is formed by the quadrilateral. The quadrilateral with one short side is corrected to form a triangular element.

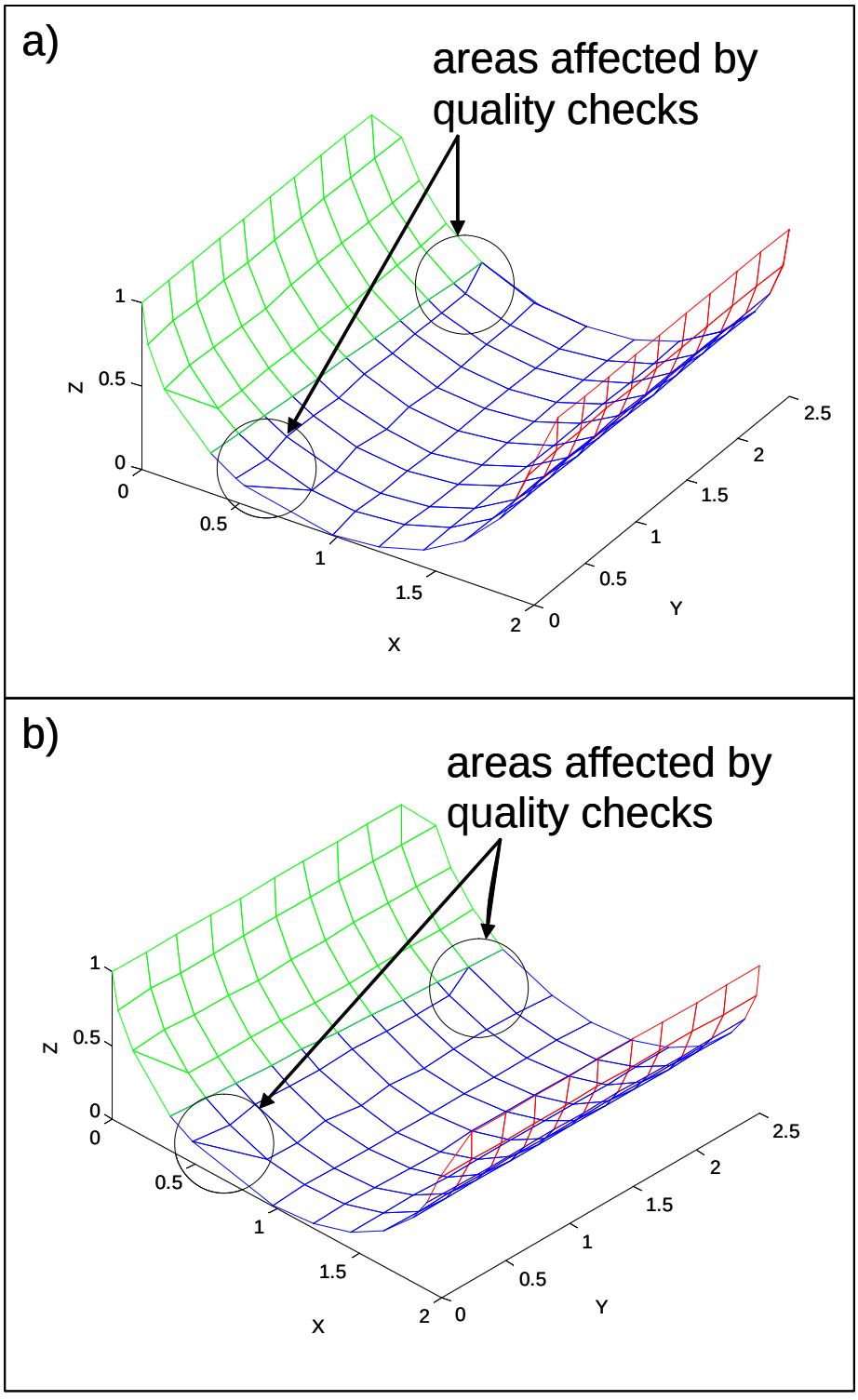


Figure 3.45: Detailed example - boundary match and quality check for partition 1/3

After the boundaries are matched for all partitions, another quality check is performed. This quality check is more thorough. The operations are shown in Figure 3.19. For this example problem, only one element is corrected in the second quality check. Figure 3.46a shows the element to be corrected. An interior angle is above the quality cutoff of 135 degrees. Therefore, that angle will be divided, and a triangular element will be formed as shown in Figure 3.46b. Once the quality check is completed, the surface mesh is finished, and the offset is calculated.

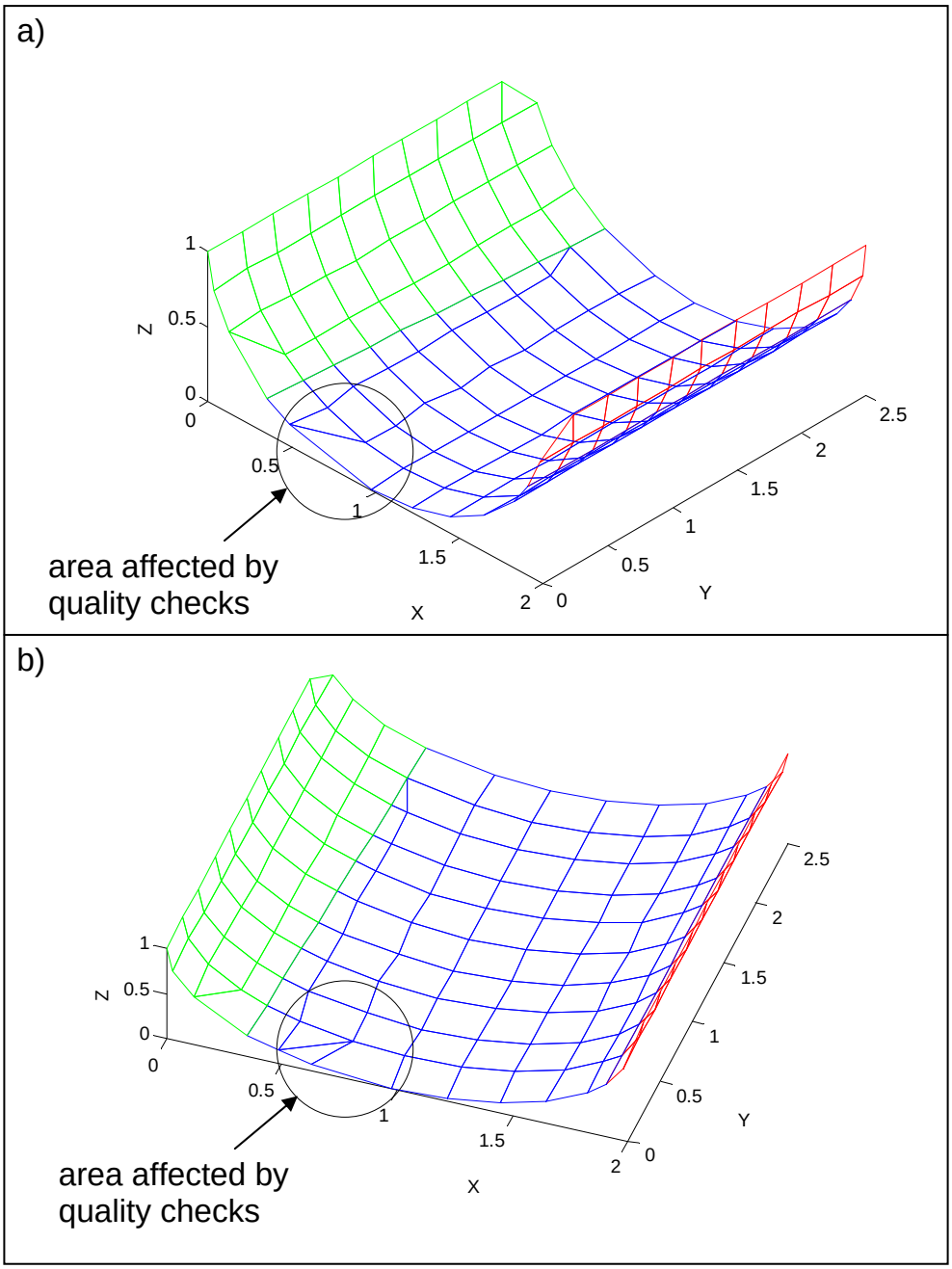


Figure 3.46: Detailed example - secondary quality check

The offset mesh is calculated directly from the surface mesh. Both methods of offsetting match the vertices of the surface mesh to offset points to create the offset mesh. Therefore, the same pattern of mesh, number of triangular elements, etc. will be the same on the surface and offset meshes. Figure 3.47 shows the surface mesh and offset mesh side by side. The offset mesh is shown on the left, and the original surface mesh is shown on the right. The two meshes plotted together are shown in Figure 3.48. All that is left to do to complete the example is to insert cellular material between the two layers of mesh.

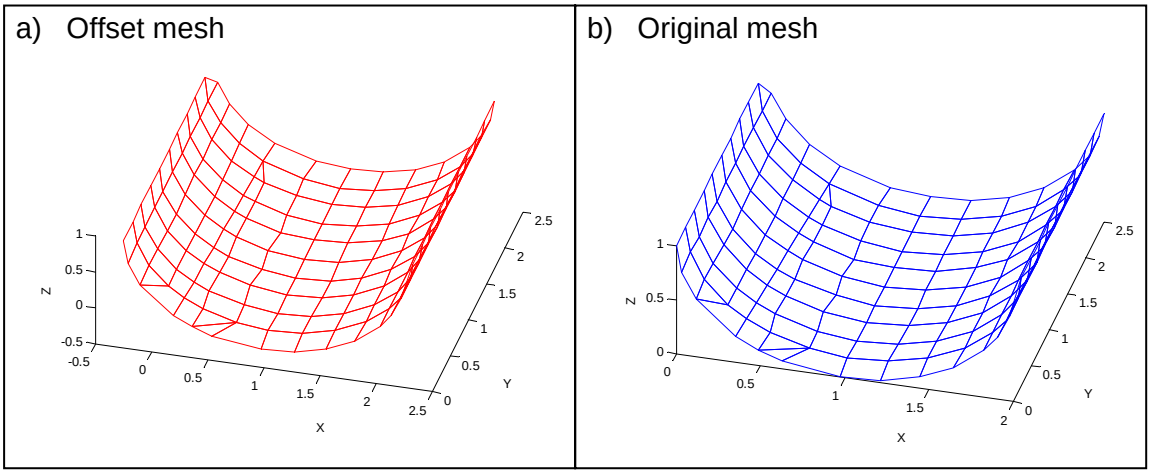


Figure 3.47: Detailed example - calculated offset mesh

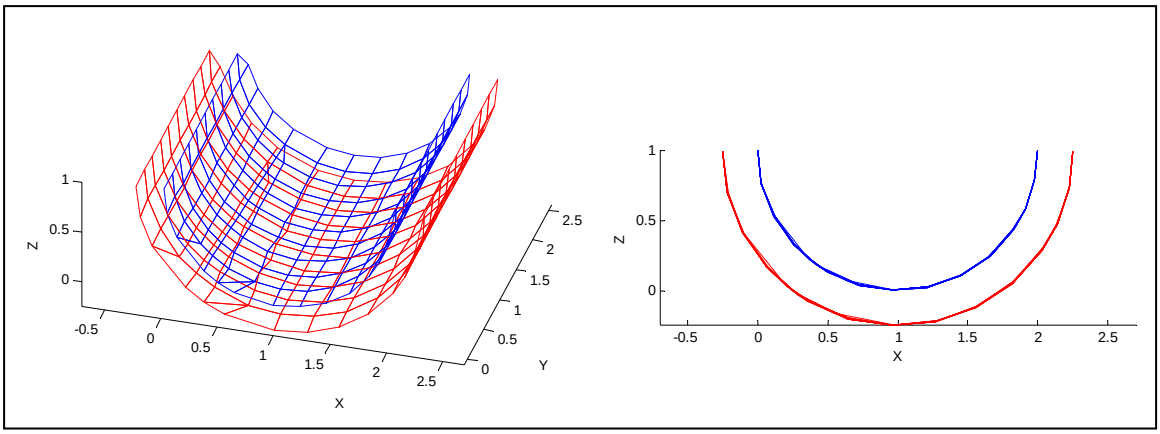


Figure 3.48: Detailed example - surface and offset meshes plotted together

For this example, the cellular material primitives used are shown in Figure 3.49. For every quadrilateral element, a primitive shown in Figure 3.49a will be inserted, and for every triangular element, a primitive shown in Figure 3.49b will be inserted. The resulting cellular material calculated for the input surface is shown in Figure 3.50. Further examples will be covered in the next chapter.

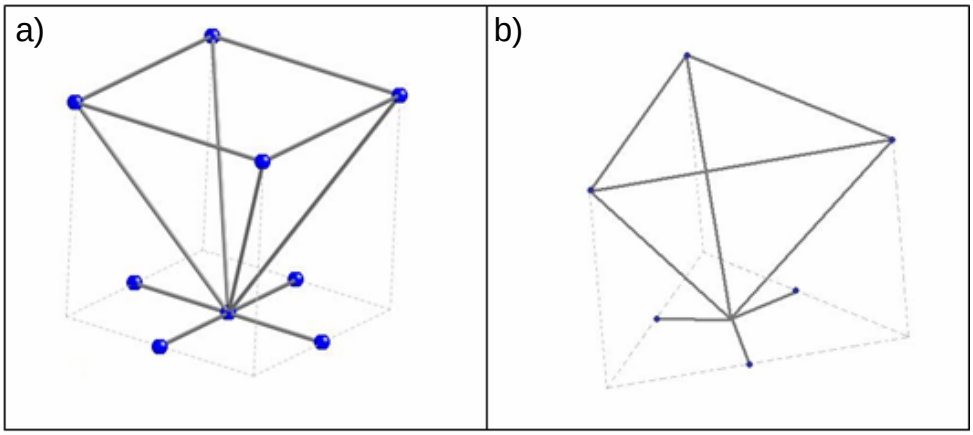


Figure 3.49: Detailed example - cellular material primitive

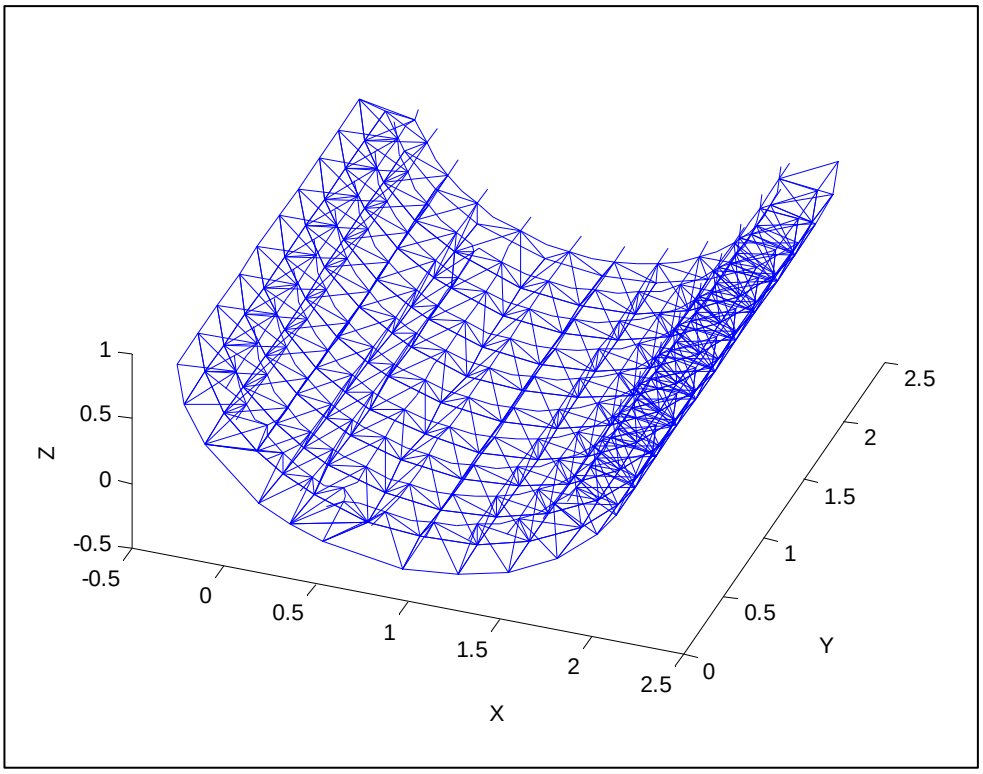


Figure 3.50: Detailed example - cellular material

3.11 Summary

This chapter presents a method for creating cellular structure from an input surface. The input surface is sampled to refine the definition. It is then broken into partitions based on a user input. Each partition is then flattened onto a plane so that a square grid can easily be overlaid. The size of the squares in the grid is a user input also. The vertices of the grid are matched to sampled points on the surface to create quadrilateral finite elements. A quality check is performed that finds element side lengths less than one quarter of the desired length. These sides are eliminated, which results in some elements being deleted or the formation of triangular elements. The finite element meshes on the separate partitions are then joined to create one continuous mesh

on the input surface. Another quality check is performed that again eliminates short side lengths and also checks interior angles of quadrilateral elements. Interior angles above 135 degrees are corrected. The offsetting function is then performed. Two different offsetting functions were developed. One function offsets all sampled points on the surface, creating a point cloud. Each corner of each element on the input surface is then matched to the nearest point in the offset point cloud. The other offsetting function individually offsets only the corner points of the elements. The second method is much faster. Both offsetting methods result in an offset mesh that matches the surface mesh. The volume between the two meshes is then filled with cellular material primitives, forming a continuous cellular material design covering the input surface.

A detailed example problem is stepped through to highlight each of the steps of the method. The example also demonstrates that the process produces viable cellular structure for the input surface geometry chosen. The specific goals of this research call for square mesh on the input surface and a uniform offset. The example problem in this chapter does not necessarily demonstrate that these goals have been met. The next chapter presents several example problems that will demonstrate the quality of the method and the degree to which it meets meshing and offsetting goals.

4 Examples

The goal of the algorithm described in the previous chapter is to design cellular material for an input surface. The algorithm calculates a finite element mesh on the input surface and calculates an offset surface. The mesh on the input surface is then used to create a matching mesh on the calculated offset surface, but the mesh on the offset surface is not directly calculated. The three dimensional space between the input and offset surfaces is then filled with a specified cellular material unit cell.

The purpose of this chapter is to evaluate the performance of the algorithm. The aspects of the cellular material that are directly calculated by the algorithm, the mesh on the input surface and the offset, are investigated using several example problems. The results are compared to desired outcomes using some direct comparison and some general finite element quality criteria.

4.1 Evaluation Criteria

The desired mesh on the input surface is square: Four equal sides and four right angles. To measure how closely the mesh produced by the algorithm comes to being square, several quantities are studied. The first quantity is the side length. Each side length is measured and compared to the desired length, which is an input to the algorithm. The other two quantities are standard finite element quality criteria. They are skew and aspect ratio. Skew relates to the squareness of the element's angles. Aspect ratio relates to the equality of the side lengths of the element.

An arbitrary quality criterion for the minimum allowable side length is checked automatically by the algorithm and used to correct bad elements. The minimum side length is one quarter of the desired input length. Any side length below one quarter of desired length should be eliminated by the algorithm in the process of creating the surface mesh. There is no maximum value imposed. All element lengths should be as close to the input desired length as possible.

Figure 4.1 shows the angle, α, which is used to calculate skew and aspect ratio. The value recorded as skew is ninety degrees minus this angle, given in Equation 4-1 [5]. The minimum skew is zero, which is a perfect square. A standard measure of a good element for finite element analysis is skew less than thirty degrees. This corresponds to accurate analytical results. It is used here as a general guideline, though the goal is to have a skew as close to zero as possible.

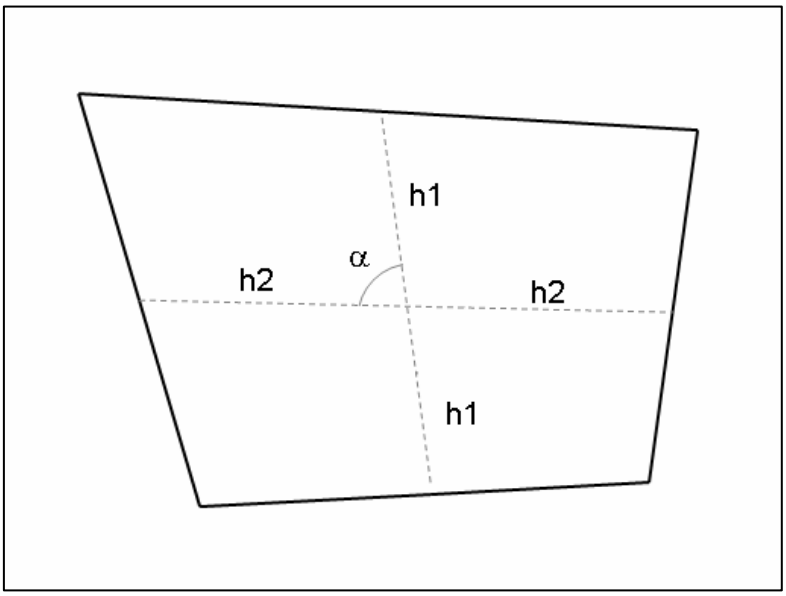


Figure 4.1: Element quality measurements

$$\text{Skew} = |90 - \alpha| \tag{4-1}$$

The angle, α, shown in Figure 4.1 is also used in calculating the aspect ratio. The other quantities needed for the aspect ratio calculation are shown in Figure 4.2. The value recorded as aspect ratio is the worst case, i.e. maximum, of the ratios of perpendicular components of the h-values shown in Figure 4.2 [25]. Equation 4-2 to Equation 4-6 show the calculation of aspect ratio. The minimum value for aspect ratio is one, which corresponds to a perfect square. A general rule for a good element for finite element analysis is aspect ratio less than three. This rule is for accurate finite element analysis results. It is used here as a general guideline. The goal is to have an aspect ratio as close to one as possible.

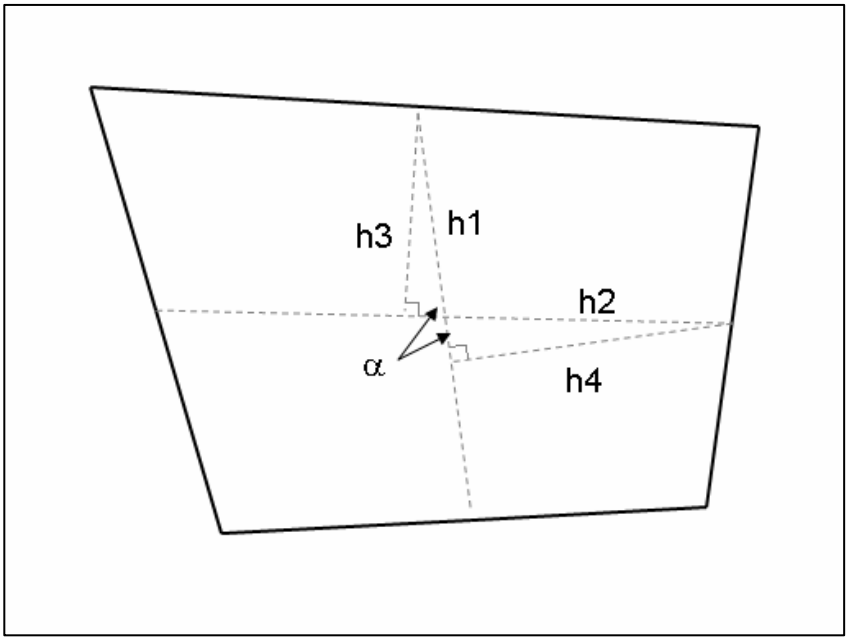


Figure 4.2: Aspect ratio measurements

$$h3 = h1*\sin(\alpha) \quad (4\text{-}2)$$

$$h4 = h2*\sin(\alpha) \quad (4\text{-}3)$$

$$AR1 = h2/h3 \text{ if } h2>h3 \quad \text{or} \quad AR1 = h3/h2 \text{ if } h3>h2 \quad (4\text{-}4)$$

$$AR2 = h1/h4 \text{ if } h1>h4 \quad \text{or} \quad AR2 = h4/h1 \text{ if } h4>h1 \quad (4\text{-}5)$$

$$\text{Aspect Ratio} = AR1 \text{ if } AR1>AR2 \quad \text{or} \quad \text{Aspect Ratio} = AR2 \text{ if } AR2>AR1 \quad (4\text{-}6)$$

The element side length, aspect ratio, and skew angle measure the performance of the meshing components of the algorithm. The number of triangles in the surface mesh is also checked. It is not a direct measure of the algorithm's performance, but it is relevant to cellular structure quality.

To evaluate the performance of the offset function of the algorithm, the length of the actual offset is compared to the desired offset, which is an input to the algorithm. Figure 4.3 shows the distance, D, that is measured to determine the quality of the offset function.

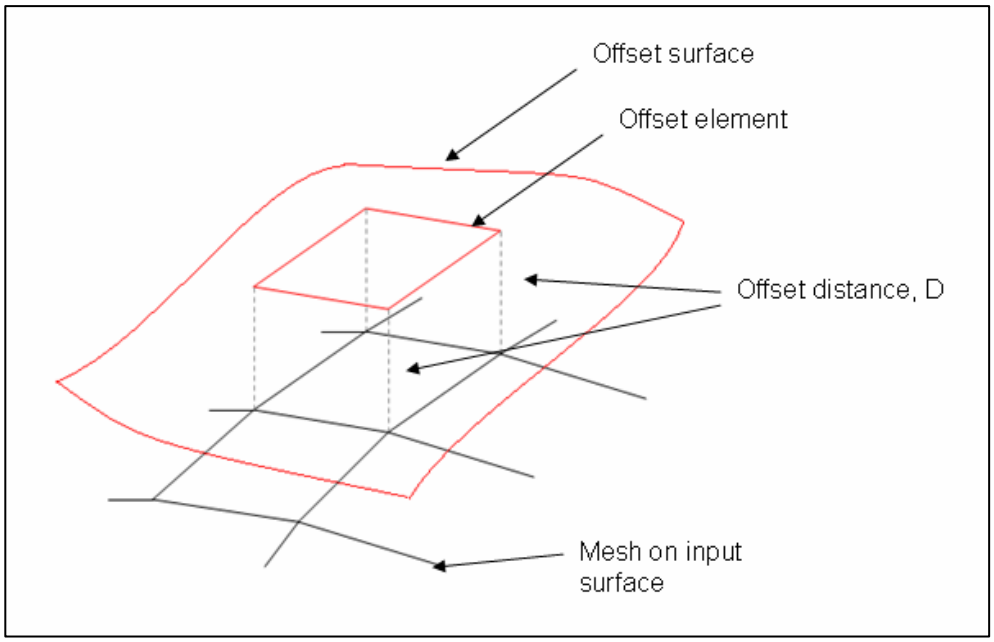


Figure 4.3: Offset distance measurement

The measured offset value should be equal to the requested value. There are two major causes of deviation. The first is due to curvature or angles in the surface. If the surface has concave angles at some points, the offset distance at those points will be larger than the requested value. The other cause of deviation results from the sampling of the surface. For both methods of offsetting described in Chapter 3, the sampling rate used on the input surface will affect the quality of the offset because both methods use the sampled points to calculate the offset surface. The calculation time is also recorded for the offsetting function. Time is not a quality measure, but it is relevant to efficiency of the algorithm.

4.2 Example Problems

The following examples were chosen to show that the algorithm used to design cellular material can be used to create cellular structure for a wide range of geometry. The examples show different cases of boundary matching, surface curvature, sampling rates, and mesh sizes. A summary of the examples presented in this section is shown in Table 4-1.

Table 4-1: Example descriptions

Example	Description	Features
1	Two planar surfaces meeting at an angle	Simple geometry, single partition boundary
2	Half Cylinder	Uniform curvature in one direction, multiple partition boundaries
3	Planar surface meeting a concave surface	Dissimilar geometries interfacing, multiple partition boundaries
4	Planar surface meeting a convex surface	Dissimilar geometries interfacing, multiple partition boundaries
5	Circular Hemisphere	Uniform curvature in multiple directions, multiple partition boundaries
6	Irregular Conic	non-uniform curvature in one direction, uniform curvature in one direction
7	Surface with a saddle point, one partition	Complex curvature, single partition
8	Surface with a saddle point, two partitions	Complex curvature, multiple partitions

4.2.1 Two planar surfaces meeting at an angle

In this example, two planar surfaces meet at an angle, as seen in Figure 4.4. This is the simplest example. The purpose of this example is to show the algorithm's performance given a simple geometry and a single, simple partition boundary. Also, for this example, there is no distortion from flattening because the input geometry has no curvature. Therefore, this example can be compared to examples with curved surfaces to determine if some poor element quality effects are due to the flattening procedure.

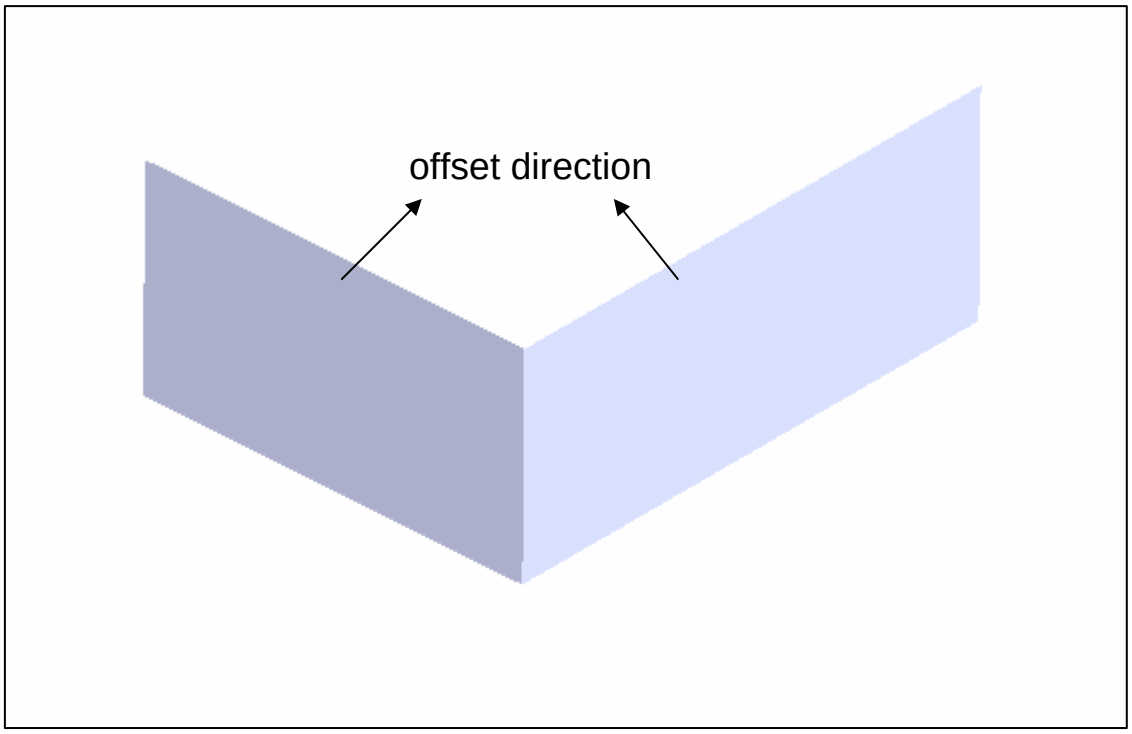


Figure 4.4: Two flat surfaces meeting at an angle

Table 4-2 shows information about the input surface for this example. Table 4-3 gives inputs for the algorithm as well as the number of partitions created and the sampling rate generated from the input sampling criterion. For this example, if the angle between the normal vectors of adjacent triangular faces on the input surface is more than one radian, the faces are put into different partitions. The resulting two partitions created from this input are shown in Figure 4.5. The division of the surface is very straight-forward for this example because of the distinct angle between the two planes.

Table 4-2: Surface data for example problem with two flat surfaces

	Geometry Description		Dimensions			
Example	Surface 1	Surface 2	Height 1 (in)	Width 1 (in)	Height 2 (in)	Width 2 (in)
1	Plane	Plane	3	2	3	2.5

Table 4-3: Algorithm inputs for example with two flat surfaces

	Algorithm Inputs				Calculated Values	
Example	Element Side Length (in)	Offset Distance (in)	Angle Criterion (radians)	Sampling Criterion	Number of Partitions	Sample Length (in)
1	0.175	0.150	1.00	70	2	0.0557

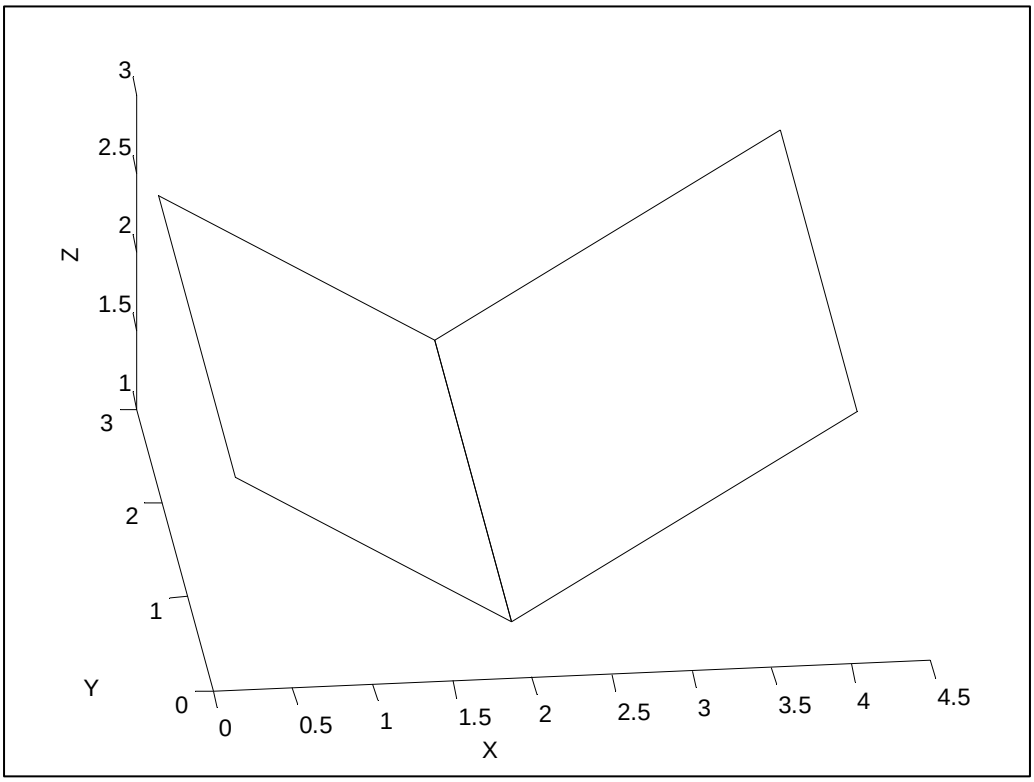


Figure 4.5: Input of two flat surfaces divided into two partitions

The surface mesh produced for this example is shown in Figure 4.6 below. The mesh matches very well at the partition boundary, and the mesh in the middle of each planar surface is approximately square. Near the edges of the surface, however, quadrilateral elements tend to be longer in one direction. This is due to the mesh size

selected for the surface. The surface cannot be exactly divided into an integer number of elements, and elements at the edges are modified to fit to the surface boundary. Figure 4.6 also shows that at some corners, triangular elements are formed. This is because the algorithm only recognizes "corners" when they are part of a partition boundary. When the mesh size comes very close to fitting the surface perfectly, the corners are often captured. In Figure 4.6, the mesh does not fit the planar surface on the left as well as the surface on the right, and the corners for the left surface have been cut off with a triangular element as a result.

Table 4-4 shows the quality data for the mesh in Figure 4.6. For this simple example, there are very few triangular elements. The mean skew is about four degrees with a standard deviation of about four degrees. Therefore, over 93% of quadrilateral elements have skew below ten degrees. The average aspect ratio is 20% past square. Because of the size of the mesh, the distorted border elements comprise a significant percentage of the total element counts. Therefore, those distorted elements impact the aspect ratio statistics. All elements have an aspect ratio below three, which is the quality criterion from finite element analysis.

The side length statistics for this example are also affected by the distorted border elements. The algorithm eliminates element side lengths that are less than one quarter of the desired length as a quality check. There is no cutoff for maximum side length. Table 4-4 shows that the resulting mesh has no side lengths less than the one quarter cutoff length. Element lengths vary from 47% to 129% of the desired length. The standard deviation is approximately 15% of the desired length. Therefore approximately 70% of

the quadrilateral side lengths are within 15% of the desired length, and 50% are within 10% of the desired length.

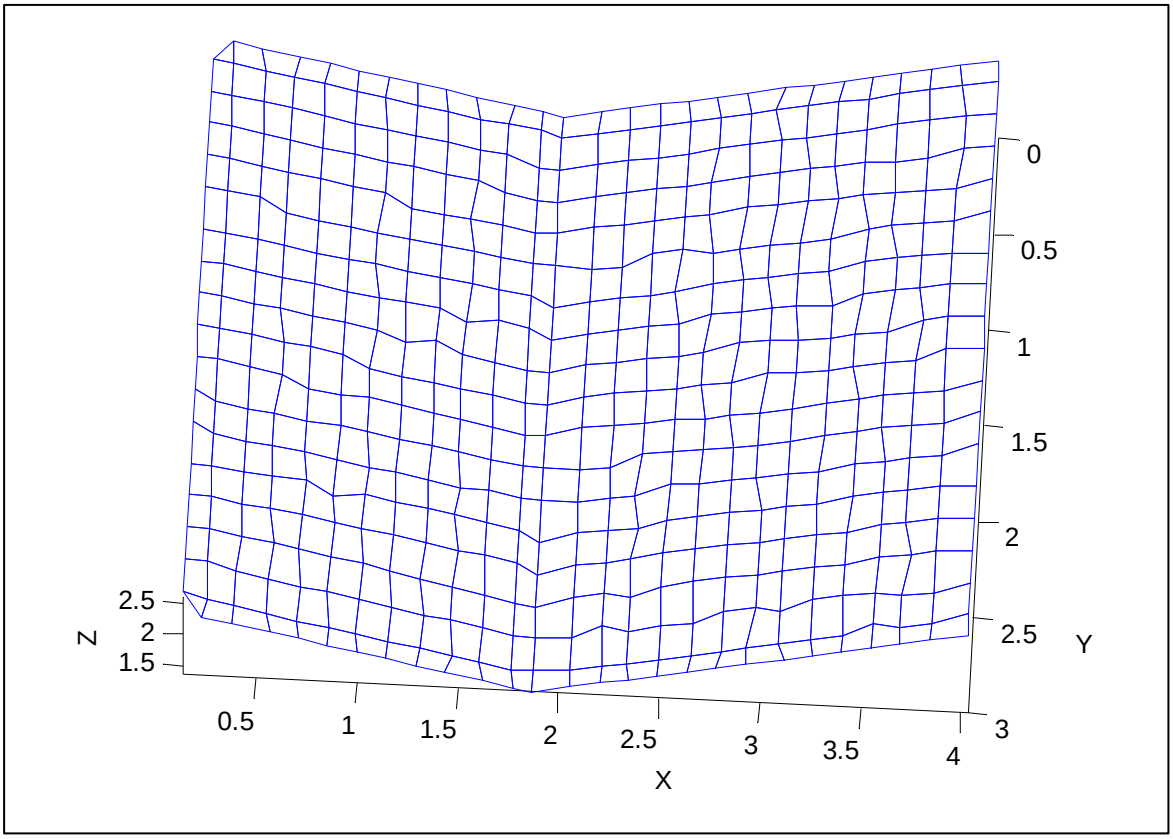


Figure 4.6: Input of two flat surfaces with mesh

Table 4-4: Mesh quality for input of two flat surfaces

Output	Value
Number of Quads	466
Number of Tris	2
Percentage Tris	0.4%
Quad Skew	
Min (degrees)	0.021
Max (degrees)	20.403
Mean (degrees)	4.218
Standard Deviation (degrees)	3.861
Quad Aspect Ratio	
Min	1.005
Max	2.109
Mean	1.208
Standard Deviation	0.243
Quad Side Length	
Desired (in)	0.175
Min (in)	0.082
Max (in)	0.225
Mean (in)	0.171
Standard Deviation (in)	0.026

The input surface with computed cellular structure is shown in Figure 4.7. The cellular structure sits between the mesh on the input surface and the mesh on the offset surface. The mesh on the offset surface is directly translated from the mesh created for the input surface, so it will have the same general quality as the mesh on the input surface. The algorithm has no function to modify or improve mesh on the offset surface. The algorithm is only responsible for defining the offset surface. The offset is computed using two different methods. Data for both methods is presented in Table 4-5.

As seen in Table 4-5, values for both methods are approximately the same. Both have an average offset value very close to the desired value with a very low standard deviation. The maximum value is different from the desired value primarily because of the concave angle in the surface. The major difference for this example is the calculation time. The fast offset method is more than ten times faster than the other method.

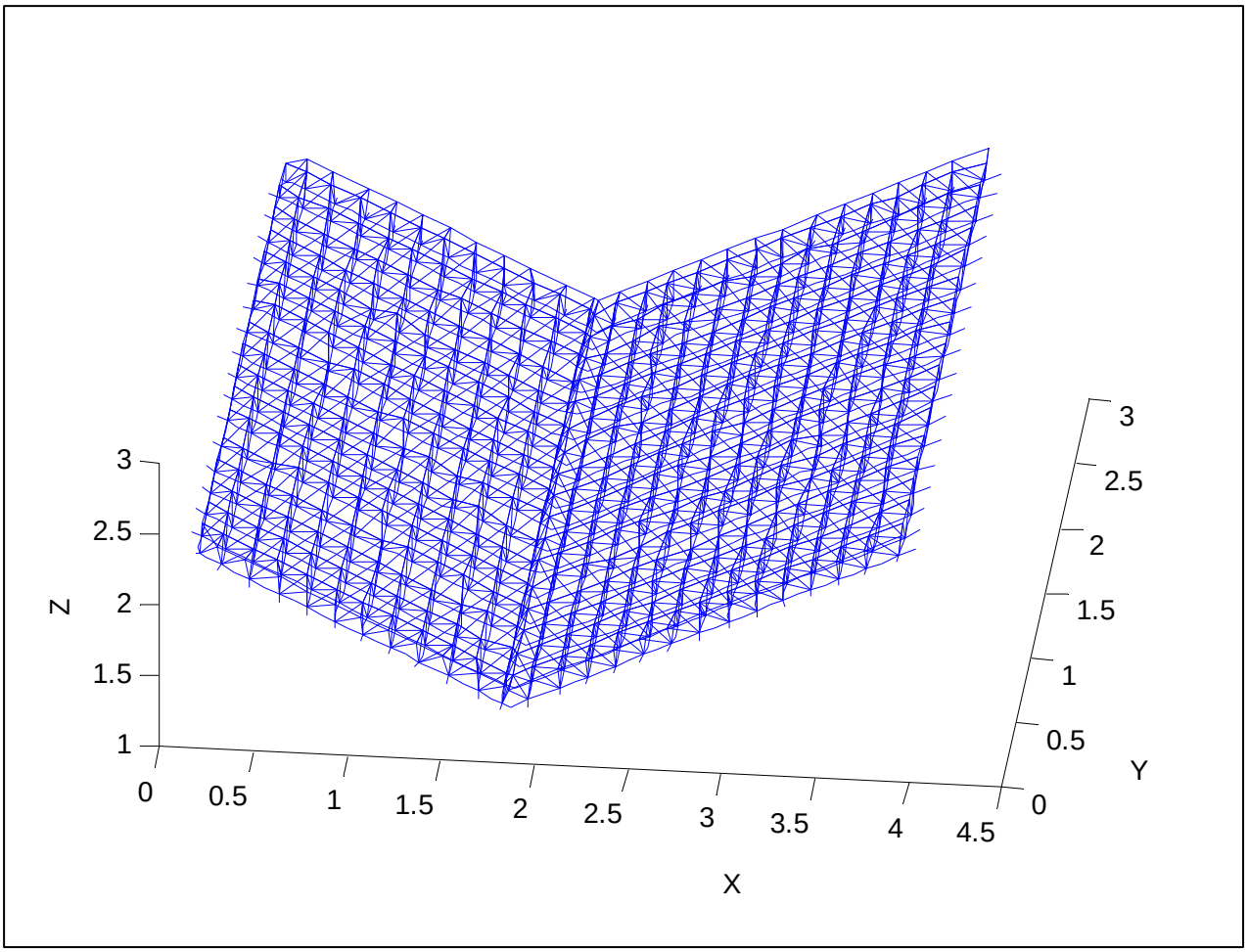


Figure 4.7: Input of two flat surfaces with truss cellular structure

Table 4-5: Offset quality for two flat surfaces

Output	Value
Fast Offset	
Time (seconds)	53
Desired Offset (in)	0.150
Minimun Offset (in)	0.150
Maximum Offset (in)	0.172
Mean Offset (in)	0.151
Standard Deviation (in)	0.0041
Original Offset	
Time (seconds)	584
Desired Offset (in)	0.150
Minimun Offset (in)	0.150
Maximum Offset (in)	0.187
Mean Offset (in)	0.151
Standard Deviation (in)	0.0053

4.2.2 Cylindrical surface

The purpose of this example is to test the function of the cellular material generating algorithm for a surface with curvature in one direction. The cylindrical surface in this example, shown in Figure 4.8, has a uniform radius and arc length along its length. This example will show how well the algorithm can handle multiple partition boundaries and curvature. It is also an example with no concave angles, which gives a straight-forward result for the offsetting function.

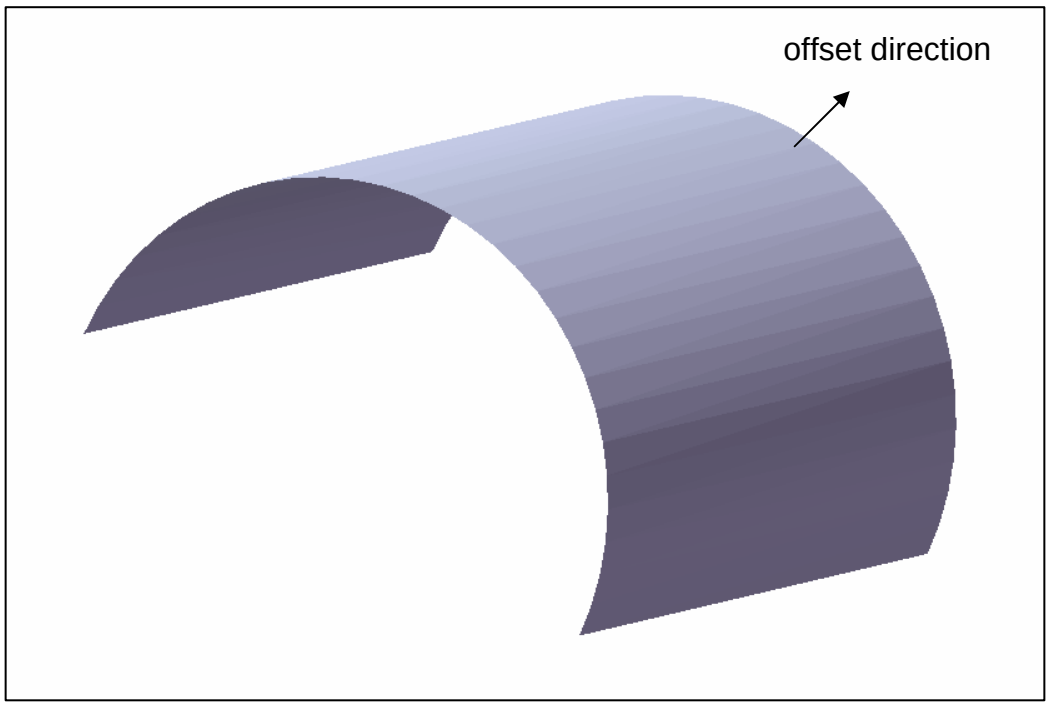


Figure 4.8: Half regular cylinder

Table 4-6 shows the criterion input to the algorithm to create three partitions. If the angle between the normal vectors of adjacent triangular faces on the input surface is more than 0.87 radian, the faces are put into different partitions. The resulting three partitions created from this input are shown in Figure 4.9.

Table 4-6: Algorithm data for cylindrical surface example

	Geometry Description	Dimensions		Algorithm Inputs				Calculated Values	
Example	Surface	Height (in)	Radius (in)	Element Side Length (in)	Offset Distance (in)	Angle Criterion (radians)	Sampling Criterion	Number of Partitions	Sample Length (in)
2	Half Cylinder	2.5	1	0.085	0.080	0.87	80	3	0.025

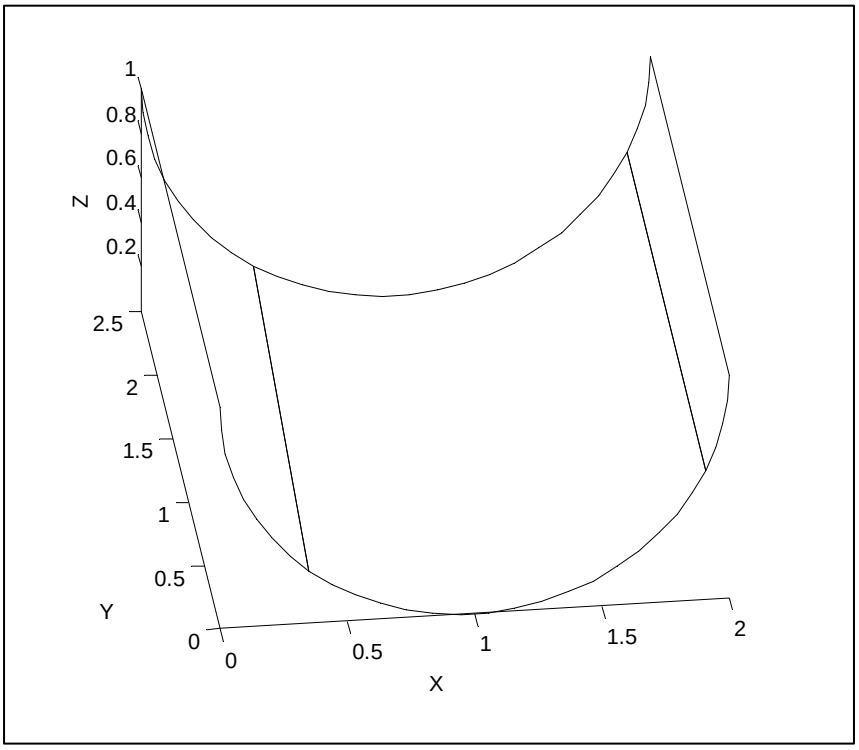


Figure 4.9: Cylindrical input surface divided into three partitions

The division of cylindrical surfaces can sometimes produce partitions that do not flatten to rectangles. An example of two ways a typical cylindrical surface can be divided is shown in Figure 4.10. Figure 4.10a shows an example of an original surface represented by triangular faces. Figure 4.10b shows two partitions formed from the original input surface that will be flattened to rectangles. This partitioning is preferred. Figure 4.10c shows two partitions that will not flatten to rectangles. This produces some

poor elements because square mesh elements are being fit to an irregularly shaped partition. For this example with a cylindrical surface input, one partition can be flattened to a rectangle, and the other two are irregular. The mesh quality is affected by this division of the surface.

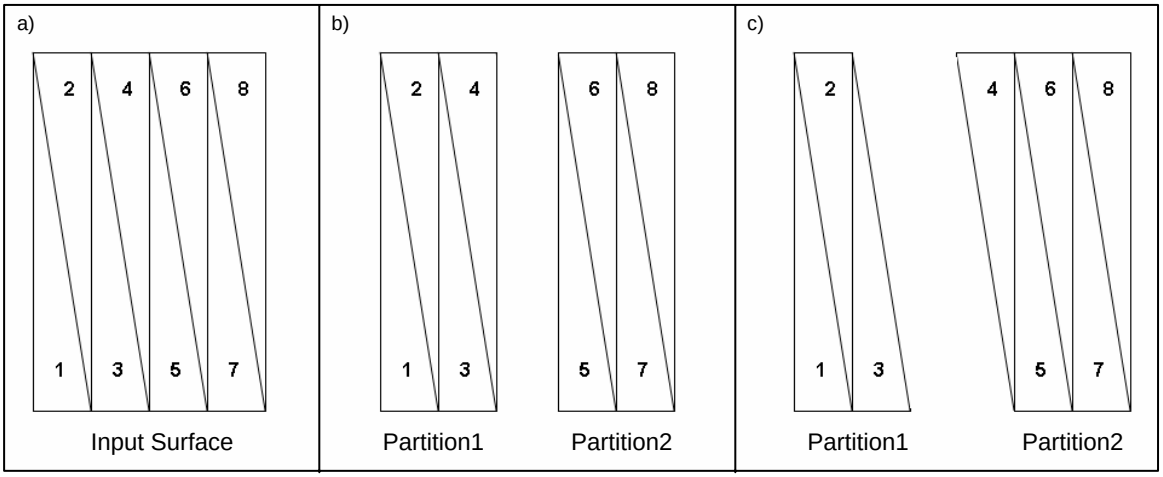


Figure 4.10: Partition division examples

The mesh on the input surface is shown in Figure 4.11. The partition boundaries are easily identified within the mesh. One of the three partitions is small compared to the mesh size, so the portion of elements that are approximately square is fairly low. The other two partitions have a number of triangular and irregular elements at the boundary. However, the two larger partitions have large sections with very regular mesh.

Table 4-7 shows the surface mesh quality statistics. The average skew for quadrilateral elements is approximately 2.5 degrees, with a standard deviation for skew of about three degrees. This is a good result for skew. Over 99.6% of quadrilateral elements have a skew of less than ten degrees. The average aspect is 23% past square. A large contribution to the aspect ratio value is elements at the partition boundaries. The

previous example had only one simple partition boundary and showed an average aspect ratio of only 20% past square. This example has two boundaries, with one that complicates the meshing. Therefore, the slightly increased aspect ratio is not unexpected.

Element side lengths are also affected by the irregular partition boundaries in this example. Side lengths range from 35% to 153% of desired length. This meets the criterion of a minimum of 25% of desired length that is specified in the algorithm. The standard deviation of side length is 15.5 % of the desire length. Therefore, 66% of elements are within 15% of the desired length, and 47% are within 10% of the desired length.

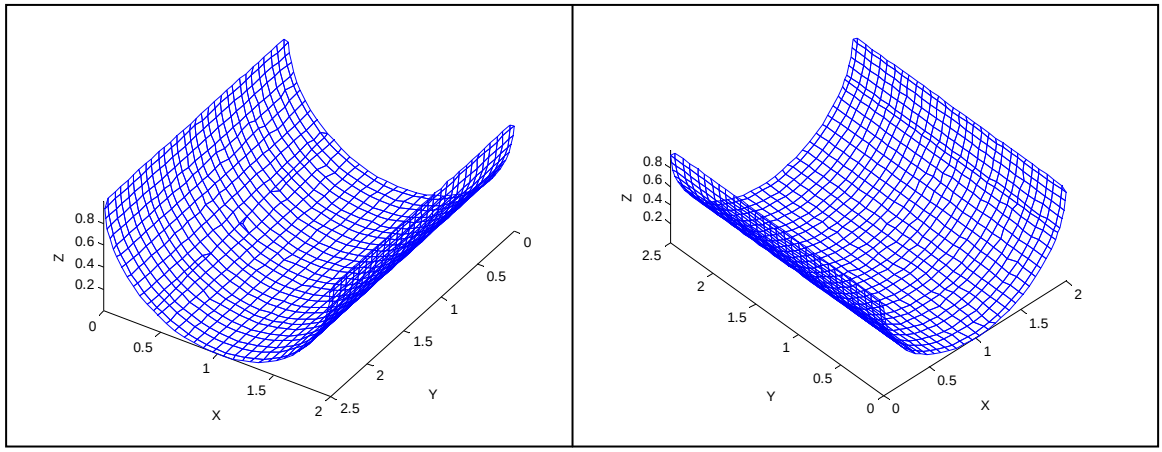


Figure 4.11: Cylindrical input surface with mesh

Table 4-7: Cylindrical surface mesh quality

Output	Value
Number of Quads	1153
Number of Tris	9
Percentage Tris	0.8%
Quad Skew	
Min (degrees)	0.000
Max (degrees)	22.997
Mean (degrees)	2.462
Standard Deviation (degrees)	2.826
Quad Aspect Ratio	
Min	1.0029
Max	3.0799
Mean	1.2318
Standard Deviation	0.2529
Quad Side Length	
Desired (in)	0.085
Min (in)	0.0300
Max (in)	0.1298
Mean (in)	0.0826
Standard Deviation (in)	0.0132

The input surface with cellular structure is shown in Figure 4.12. This example provides a direct comparison of the two offset techniques because of the simple geometry and the lack of concave angles. Table 4-8 shows the result of the offset. Both methods produce very good results. The most distinguishing statistic is calculation time. The faster method time is approximately 20 minutes. The original offsetting method takes approximately 5000 minutes, or 80 hours.

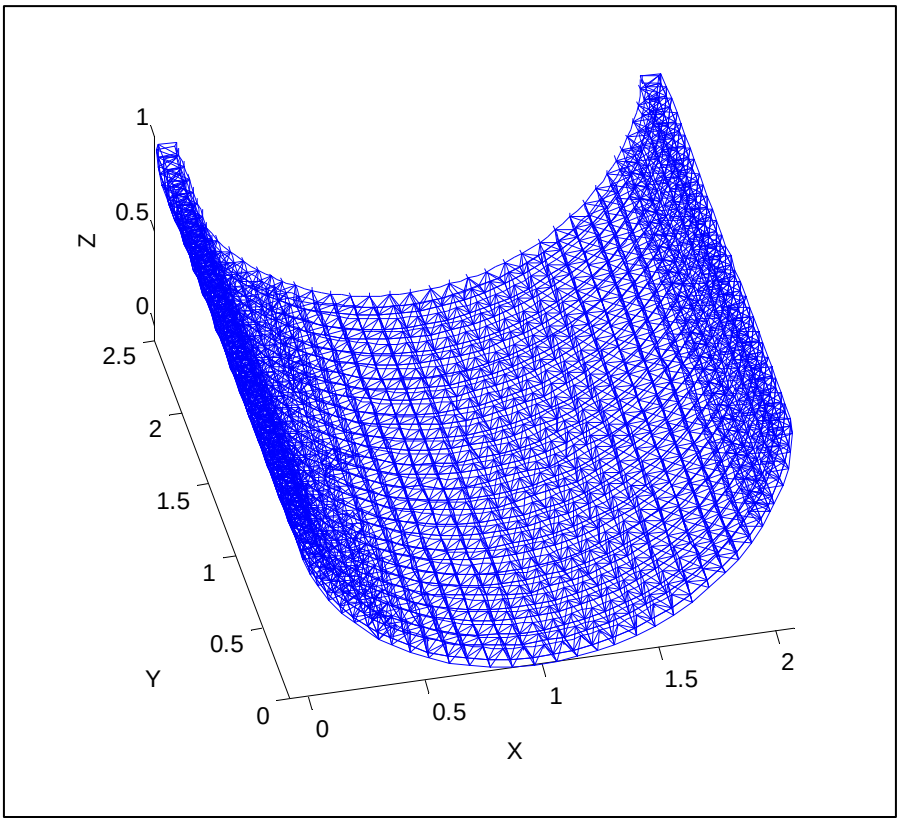


Figure 4.12: Cylindrical input surface with cellular structure

Table 4-8: Cylindrical surface offset quality

Output	Value
Fast Offset	
Time (seconds)	1352
Desired Offset (in)	0.080
Minimun Offset (in)	0.079919
Maximum Offset (in)	0.080
Mean Offset (in)	0.079993
Standard Deviation (in)	1.70E-05
Original Offset	
Time (seconds)	286137
Desired Offset (in)	0.080
Minimun Offset (in)	0.079988
Maximum Offset (in)	0.080
Mean Offset (in)	0.079995
Standard Deviation (in)	6.00E-06

4.2.3 Flat plane meeting a concave surface

This example is slightly more complicated than the example using two planar surfaces. The purpose of this example is to evaluate the performance of the algorithm for concave curved surfaces meeting flat surfaces. It is possible that having curved and flat surfaces come together in this way can produce very bad partition divisions and poor mesh. The input surface used for this example is shown in Figure 4.13.

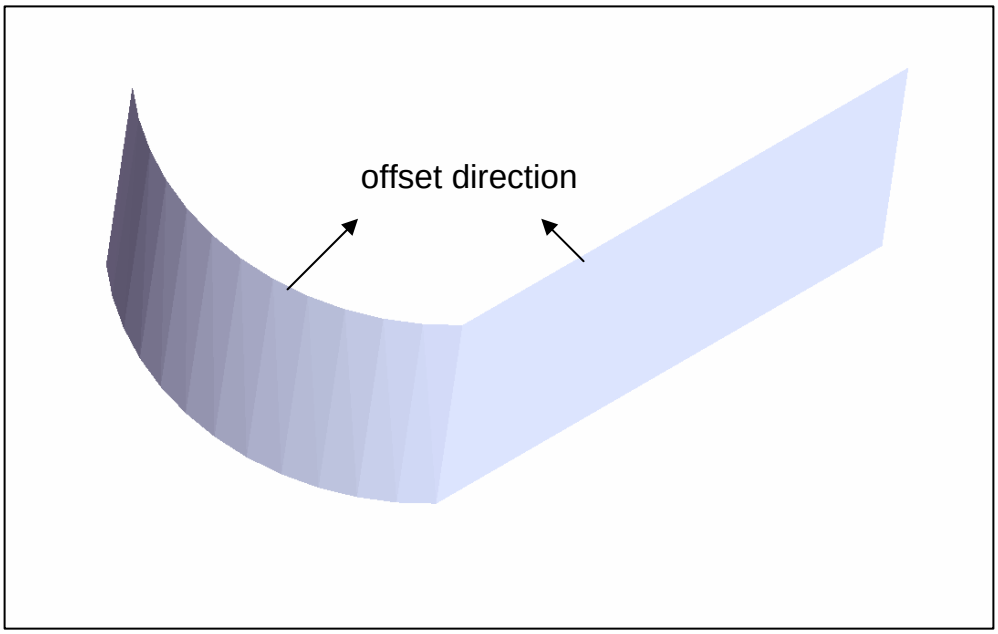


Figure 4.13: Flat surface meeting a concave surface at an angle

Surface data is shown in Table 4-9. Because of the curvature, several partitions will be created. The input for the algorithm is shown in Table 4-10. An input of 0.5 radian for the angle criterion produces four separate partitions, as shown in Figure 4.14. The surface division produces a good result. The flat portion of the input surface is

separated cleanly from the curved portion, and the curved portion is divided nearly evenly into three partitions that flatten to rectangles.

Table 4-9: Surface data for flat/concave surface example

	Geometry Description		Dimensions			
Example	Surface 1	Surface 2	Height 1 (in)	Width 1 (in)	Height 2 (in)	Radius 2 (in)
3	Plane	1/4 Cylinder	5	9	5	5

Table 4-10: Algorithm input data for flat/concave surface example

	Algorithm Inputs				Calculated Values	
Example	Element Side Length (in)	Offset Distance (in)	Angle Criterion (radians)	Sampling Criterion	Number of Partitions	Sample Length (in)
3	0.320	0.200	0.50	40	4	0.125

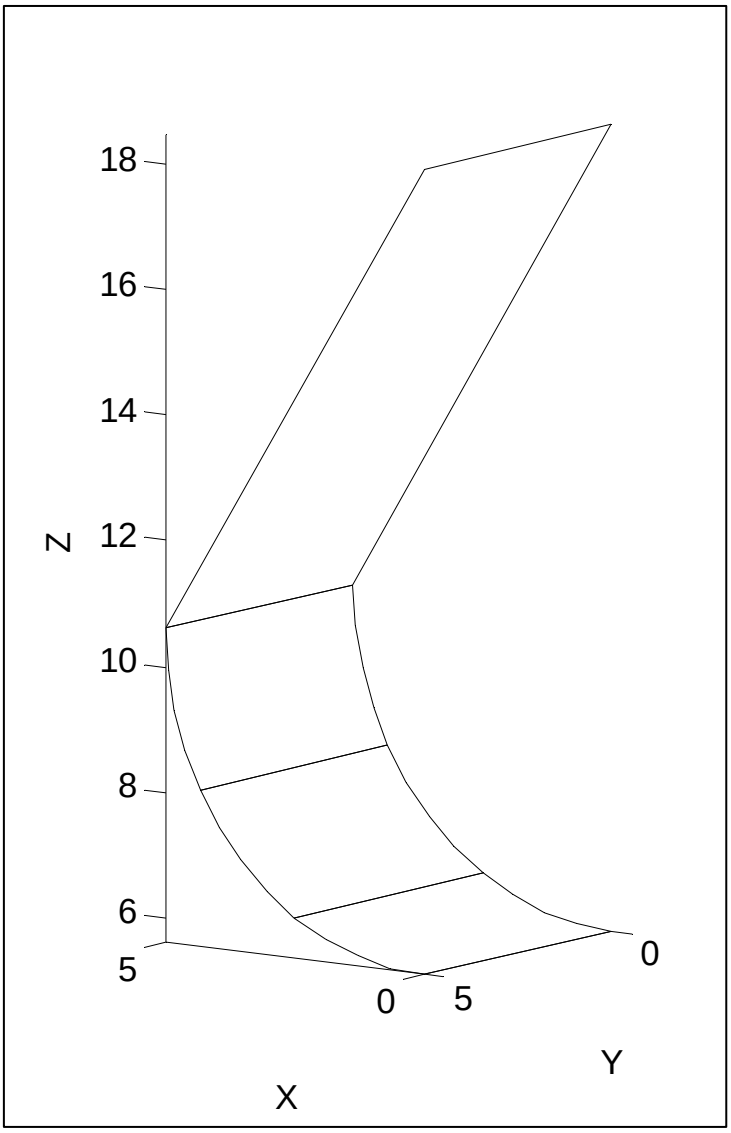


Figure 4.14: Flat/concave input surface divided into four partitions

The surface mesh for the input surface in this example is shown in Figure 4.15. As in both previous examples, elements at the boundaries of the surface are noticeably distorted. However, there are fewer distorted boundary elements, as a percentage of total elements, in this example. There are very few triangular elements. Again, the triangular elements are at the corners of the surface because these corners are not specifically recognized by the algorithm. The mesh at partition boundaries is generally good because there are equal numbers of elements on either side of each partition so that few skewed and no triangular elements are formed.

The mesh quality statistics for this example are shown in Table 4-11. The mean skew for this example is 4.5 degrees. The standard deviation for skew is 3.34, and 95% of quadrilateral elements in this example have a skew of less than ten degrees. The average aspect ratio is 16% past square. This is less than in the example with two flat surfaces, partly due to the fact that there is a lower percentage of distorted elements from the boundary region. Also, the mesh size selected for this example fits the input very well. The partition boundaries are almost indistinguishable on the meshed surface. Again, no elements have an aspect ratio greater than three, which is the finite element analysis quality criterion.

The side length statistics for this example are also affected by the distorted border elements. Table 4-11 shows that the resulting mesh for this example has no side lengths less than the one quarter of the desired length, which is a built in quality criterion. Element lengths vary from 35% to 137% of the desired length. The standard deviation is approximately 15% of the desired length. Therefore approximately 70% of the

quadrilateral side lengths are within 15% of the desired length, and 50% are within 10% of the desired length.

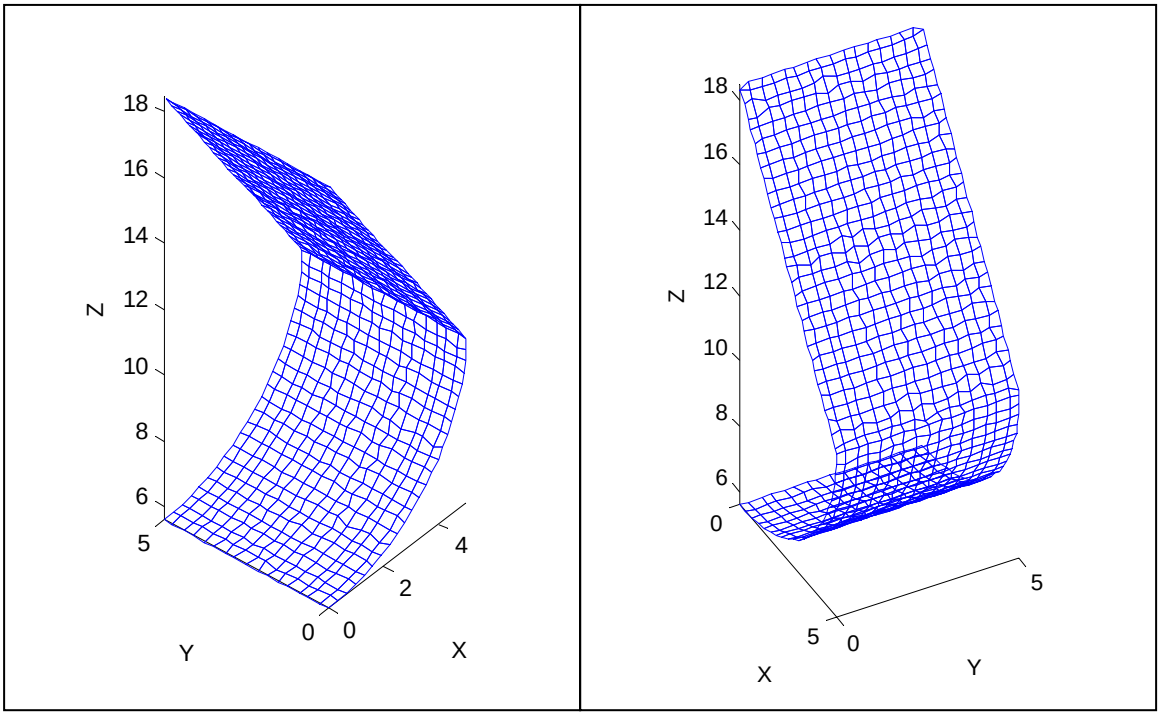


Figure 4.15: Flat/concave input surface with mesh

Table 4-11: Flat/concave surface mesh quality

Output	Value
Number of Quads	862
Number of Tris	2
Percentage Tris	0.2%
Quad Skew	
Min (degrees)	0.000
Max (degrees)	17.325
Mean (degrees)	4.503
Standard Deviation (degrees)	3.334
Quad Aspect Ratio	
Min	1.002
Max	2.121
Mean	1.162
Standard Deviation	0.146
Quad Side Length	
Desired (in)	0.320
Min (in)	0.113
Max (in)	0.438
Mean (in)	0.318
Standard Deviation (in)	0.048

The input surface with computed cellular structure for this example is shown in Figure 4.16. Offset values for both the original and fast methods are shown in Table 4-12. Both have minimum and average offset values very close to the desired value. The maximum values are slightly different, but it is unclear which is “correct”. The maximum value is different from the desired value in this case because of the concave angle. The fast method produces a much smaller standard deviation. The standard deviation for the offset length for the original offsetting method is nearly two percent of the desired length. The standard deviation for the fast method is less than one half of one percent of desired length, which is a better result. The fast method is also over 13 times faster than the original method.

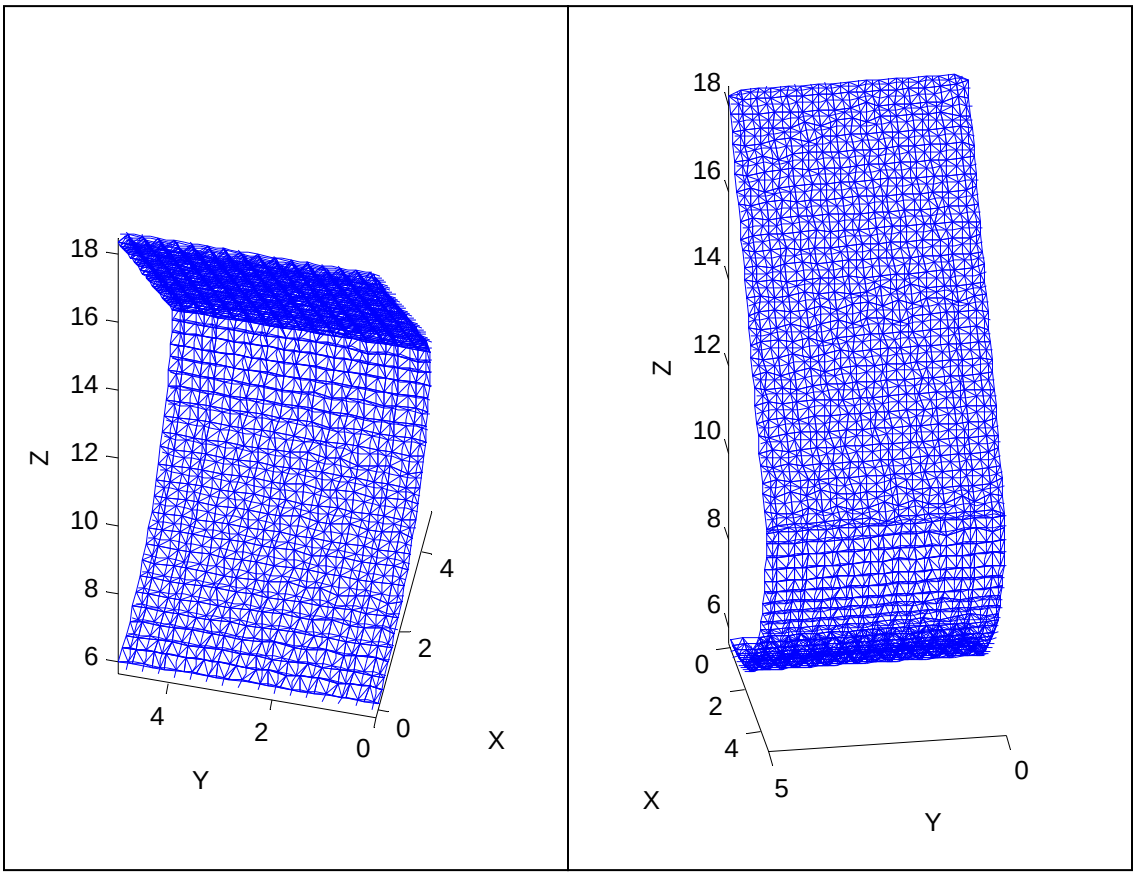


Figure 4.16: Flat/concave input surface with cellular structure

Table 4-12: Flat/concave surface offset quality

Output	Value
Fast Offset	
Time (seconds)	195
Desired Offset (in)	0.200
Minimun Offset (in)	0.19991
Maximum Offset (in)	0.2116
Mean Offset (in)	0.20010
Standard Deviation (in)	0.00087
Original Offset	
Time (seconds)	2629
Desired Offset (in)	0.200
Minimun Offset (in)	0.20000
Maximum Offset (in)	0.2299
Mean Offset (in)	0.20097
Standard Deviation (in)	0.00326

4.2.4 Flat plane meeting a convex surface

This example is similar to the previous example of a flat surface meeting a concave surface. The purpose of this example is to evaluate the performance of the algorithm for convex curved surfaces meeting flat surfaces. Many complex surface inputs may contain areas with this type of interface, and it is possible that surfaces with this configuration can produce problematic partitions or mesh quality. The input surface for this example is shown in Figure 4.17.

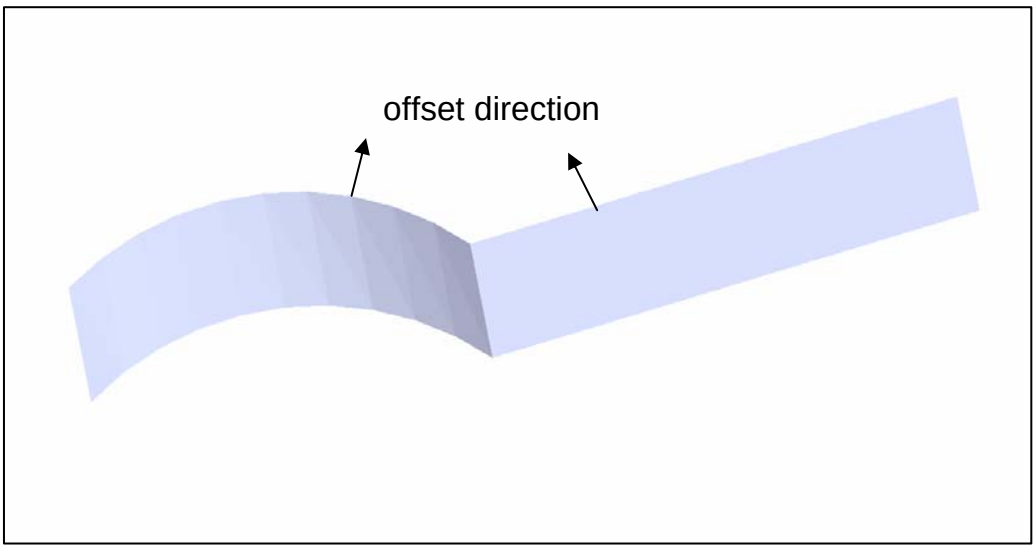


Figure 4.17: Flat surface meeting a convex surface at an angle

A description of the surface used in this example is given in Table 4-13. The input for the algorithm is shown in Table 4-14. An input of 0.5 radian for the angle criterion produces four separate partitions, as shown in Figure 4.18. The surface division produces a good result. The flat portion of the input surface is separated cleanly from the curved portion, and the curved portion is divided nearly evenly into three partitions that flatten to rectangles.

Table 4-13: Surface data for flat/convex surface example

	Geometry Description		Dimensions			
Example	Surface 1	Surface 2	Height 1 (in)	Width 1 (in)	Height 2 (in)	Radius 2 (in)
4	Plane	1/4 Cylinder	5	8.5	5	5

Table 4-14: Algorithm input data for flat/convex surface example

	Algorithm Inputs				Calculated Values	
Example	Element Side Length (in)	Offset Distance (in)	Angle Criterion (radians)	Sampling Criterion	Number of Partitions	Sample Length (in)
4	0.358	0.200	0.50	70	4	0.1426

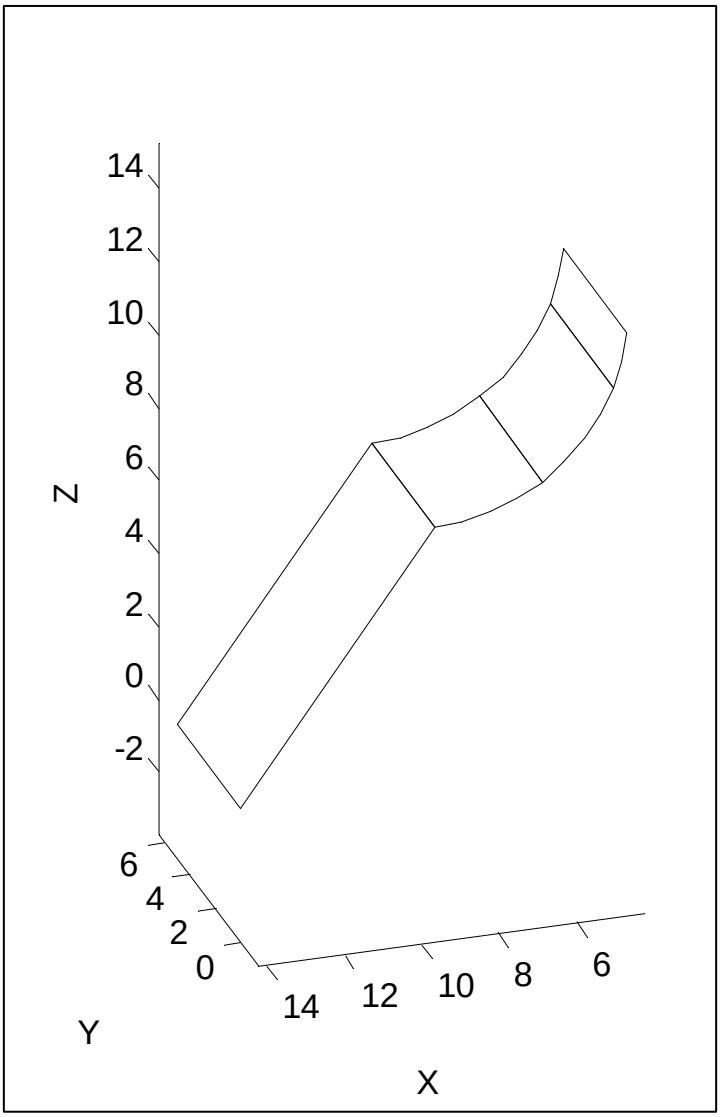


Figure 4.18: Flat/convex input surface divided into four partitions

The input surface with mesh is shown in Figure 4.19. The mesh size used for this input surface fits very well. Elements at the partition and surface boundaries are barely distorted on visual inspection. However, there are some elements on the interior of the surface that are noticeably distorted. This is due to a low sampling density. For the example with a concave surface meeting a flat surface above, the sampling distance was approximately 30% of the desired mesh side length. In that example, the interior elements are more regular. The deviation from square is largely from boundary elements. In this example with a convex surface meeting a flat surface, the sampling difference is 40% of the desired mesh side length, and interior elements are noticeably distorted.

The quality statistics for the mesh shown in Figure 4.19 are given in Table 4-15. The average skew is 3.34 degrees with a standard deviation of about three degrees. Therefore, 99% of elements have a skew below ten degrees. Even though elements appear distorted, the skew values for this example are low. The average aspect ratio is 17% past square, and the maximum aspect ratio is only 1.7, which is well below the aspect ratio of three that is defined as a bad element. These quality statistics are not significantly worse than for any of the previous examples, even though visually the mesh seems more distorted.

The side lengths range from 68% to 137% of the desired length. The average side length is 101% of desired length, and the standard deviation is about 15% of desired side length. Therefore, 50% of element sides are within 10% of desired length, and 69% of elements are within 15% of desired length. These statistics are very similar to results from previous examples.

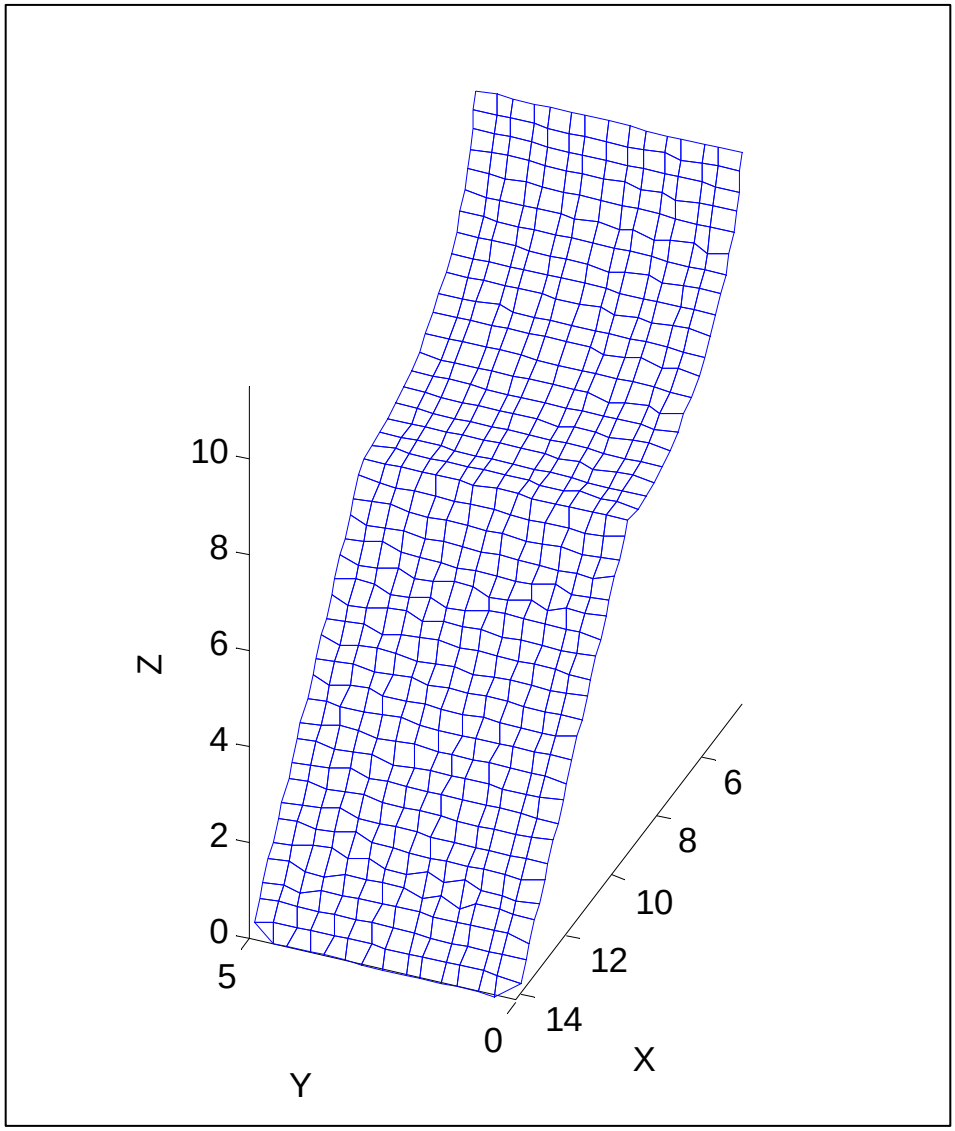


Figure 4.19: Flat/convex input surface with mesh

Table 4-15: Flat/convex surface quality mesh

Output	Value
Number of Quads	614
Number of Tris	2
Percentage Tris	0.3%
Quad Skew	
Min (degrees)	0.000
Max (degrees)	15.820
Mean (degrees)	3.343
Standard Deviation (degrees)	2.897
Quad Aspect Ratio	
Min	1.003
Max	1.695
Mean	1.173
Standard Deviation	0.134
Quad Side Length	
Desired (in)	0.358
Min (in)	0.244
Max (in)	0.489
Mean (in)	0.362
Standard Deviation (in)	0.053

The surface with cellular material applied is shown in Figure 4.20. This represents the input surfaced mesh, offset surface with mesh, and a truss pattern inserted between. The offset quality data for this example are given in Table 4-16. Both the original and fast methods produce good results. Average and minimum values are almost exactly the same for both methods and are very accurate. As with other examples that feature concave angles, the maximum values for the two methods differ. The standard deviations are both very low. Again, the computation time difference is significant. The original method takes almost 14 times longer than the fast method.

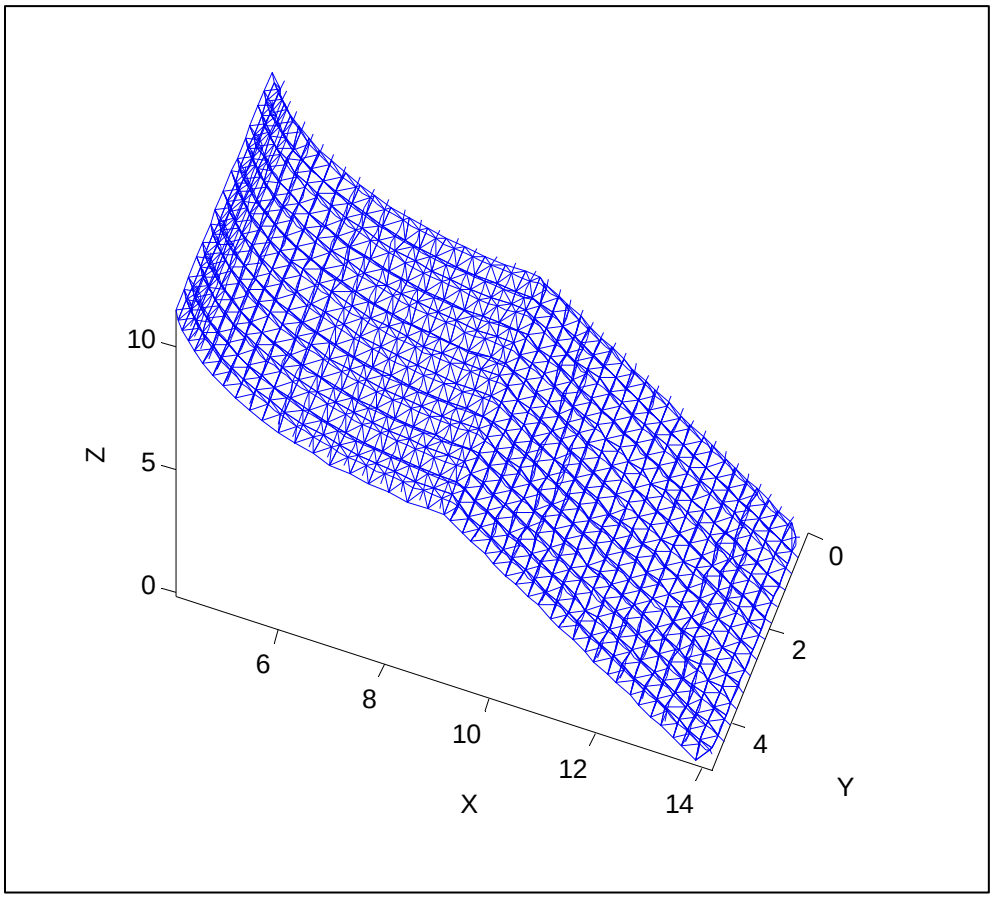


Figure 4.20: Flat/convex input surface with cellular structure

Table 4-16: Flat/convex surface offset quality

Output	Value
Fast Offset	
Time (seconds)	118
Desired Offset (in)	0.200
Minimun Offset (in)	0.19992
Maximum Offset (in)	0.222
Mean Offset (in)	0.2004
Standard Deviation (in)	0.0027
Original Offset	
Time (seconds)	1607
Desired Offset (in)	0.200
Minimun Offset (in)	0.200
Maximum Offset (in)	0.238
Mean Offset (in)	0.2008
Standard Deviation (in)	0.0048

4.2.5 Spherical surface

The purpose of this example is to test the performance of the algorithm for a surface with curvature in two directions. For this example, a spherical hemisphere is used, as shown in Figure 4.21. A spherical hemisphere has uniform curvature.

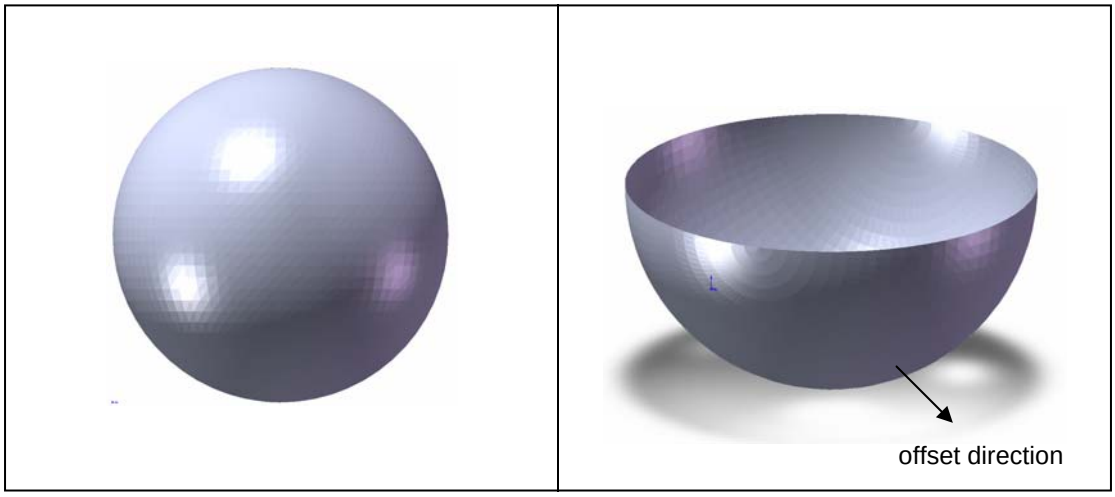


Figure 4.21: Spherical hemisphere

The input for the algorithm is shown in Table 4-17. The curvature of the sphere creates a more complicated set of partitions. The five partitions created are shown in Figure 4.22. A sphere represented in an STL file has many more triangular faces than a flat surface or a cylindrical surface. As a result, the partition boundaries can be very jagged.

Table 4-17: Algorithm data for the hemispherical surface example

	Geometry Description	Dimensions	Algorithm Inputs				Calculated Values	
Example	Surface	Radius (in)	Element Side Length (in)	Offset Distance (in)	Angle Criterion (radians)	Sampling Criterion	Number of Partitions	Sample Length (in)
5	Hemisphere	1	0.110	0.100	1.20	50	5	0.04

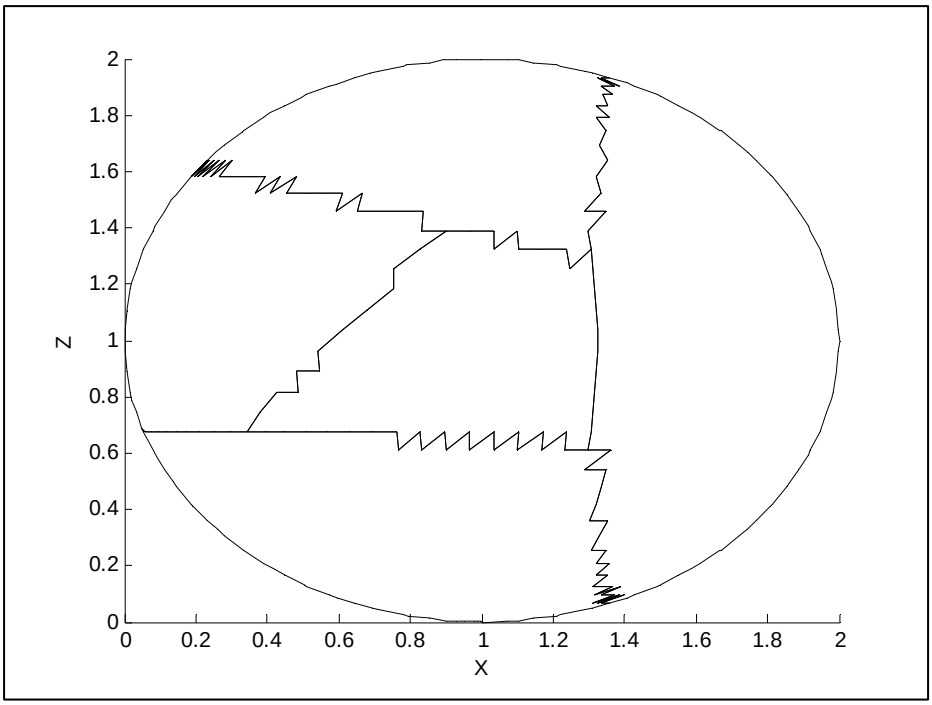


Figure 4.22: Hemispherical input surface divided into five partitions

The mesh produced for the hemispherical surface is shown in Figure 4.23. This is the first example where many triangular elements are present. Triangular elements and other distorted elements are concentrated at the partition boundaries. There are several contributors to the mesh problems at partition boundaries. The meshes from adjacent boundaries are coming in at different angles, which makes even boundaries with equal numbers of elements difficult to match without creating bad elements. Also, the boundary itself is jagged, which can produce bad mesh at the interface.

The mesh quality statistics are shown in Table 4-18. Over 20% of the elements produced are triangular elements. The goal is to have as few triangular elements as possible, so this is undesirable. If a smaller mesh size were chosen, this percentage could likely be decreased. However, a mesh size that is too small may encounter other problems, especially with the jagged boundary. The mesh in areas away from the

boundaries is generally good by visual inspection and covers a substantial portion of the surface area.

The maximum skew for this example is approximately 40 degrees, which is above the cutoff of 30 degrees that is considered a good element for finite element analysis. This value is a single outlier case. It is approximately five standard deviations above the mean. The average skew value is 6.9 degrees, and the standard deviation is 6.8 degrees. Approximately 67% of quadrilateral elements have a skew of less than ten degrees. This result is expectedly worse than previous examples because the complicated boundary interfaces.

The average aspect ratio is 17% past square. This result is not much different than previous examples, and the maximum aspect ratio is 2.3, which is below the cutoff of three. Therefore, for this example, skew is more affected than aspect ratio by the complicated partition division.

Element side lengths range from 36% to 212% of the desired mesh side length. Approximately 46% of quadrilateral elements are within 10% of the desired side length, and 64% are within 15% of desired side length. These results are slightly worse than the previous examples. For instance, the previous example with a convex surface meeting a flat surface has 69% of elements within 15% of the desired length. That example has more quality problems with mesh away from boundaries, and the hemispherical example has more quality problems with boundary elements.

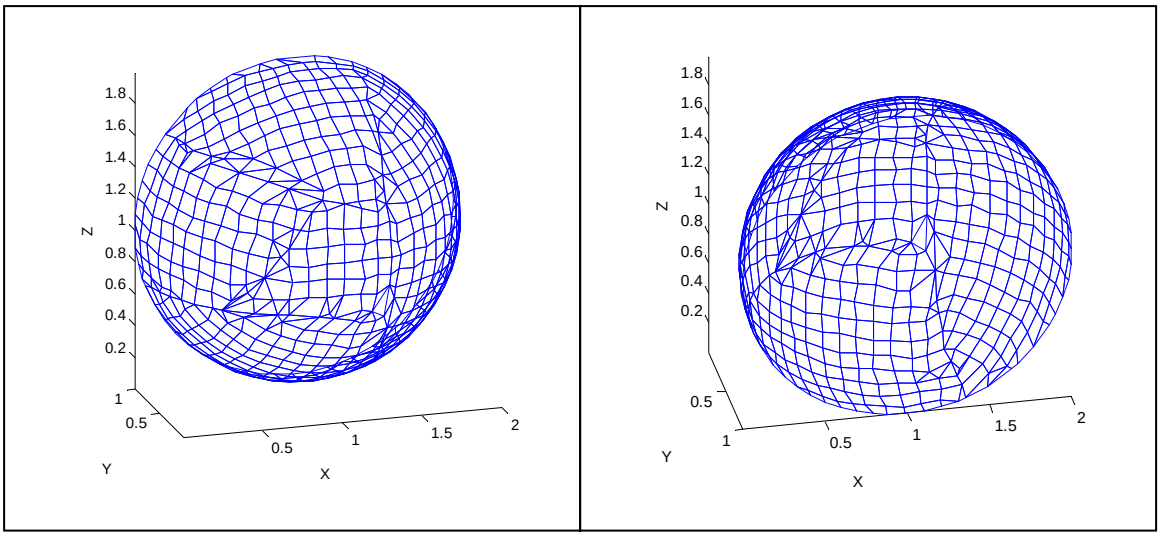


Figure 4.23: Hemispherical input surface with mesh

Table 4-18: Hemispherical surface mesh quality

Output	Value
Number of Quads	480
Number of Tris	136
Percentage Tris	22.1%
Quad Skew	
Min (degrees)	0.001
Max (degrees)	39.927
Mean (degrees)	6.927
Standard Deviation (degrees)	6.757
Quad Aspect Ratio	
Min	1.001
Max	2.282
Mean	1.168
Standard Deviation	0.188
Quad Side Length	
Desired (in)	0.110
Min (in)	0.039
Max (in)	0.233
Mean (in)	0.109
Standard Deviation (in)	0.018

The hemispherical surface with cellular material applied is shown in Figure 4.24. This is another example that features no concave angles. The offset quality results are shown in Table 4-19. The result for the cylindrical surface was so similar for the two methods that the only distinguishing result was computation time. For this example, the mean and minimum values are very similar for the two methods. The fast method tends to be slightly less than the requested value, and the original method tends to be slightly higher than the requested value for offset distance. The result is the same within reasonable limits, though. However, for the maximum value, the difference is noticeable. For the fast method, the maximum offset is only one half of one percent above the requested value. For the original method, the maximum offset is nearly four percent greater than the requested value. The standard deviation is low for both, but it is lower for the fast method. These statistics indicate that the fast method of offsetting is more accurate. Even in cases where the fast method is not noticeably better than the original method, it is preferable because it is more efficient. In this example, the fast method saves nearly seven hours of computation time.

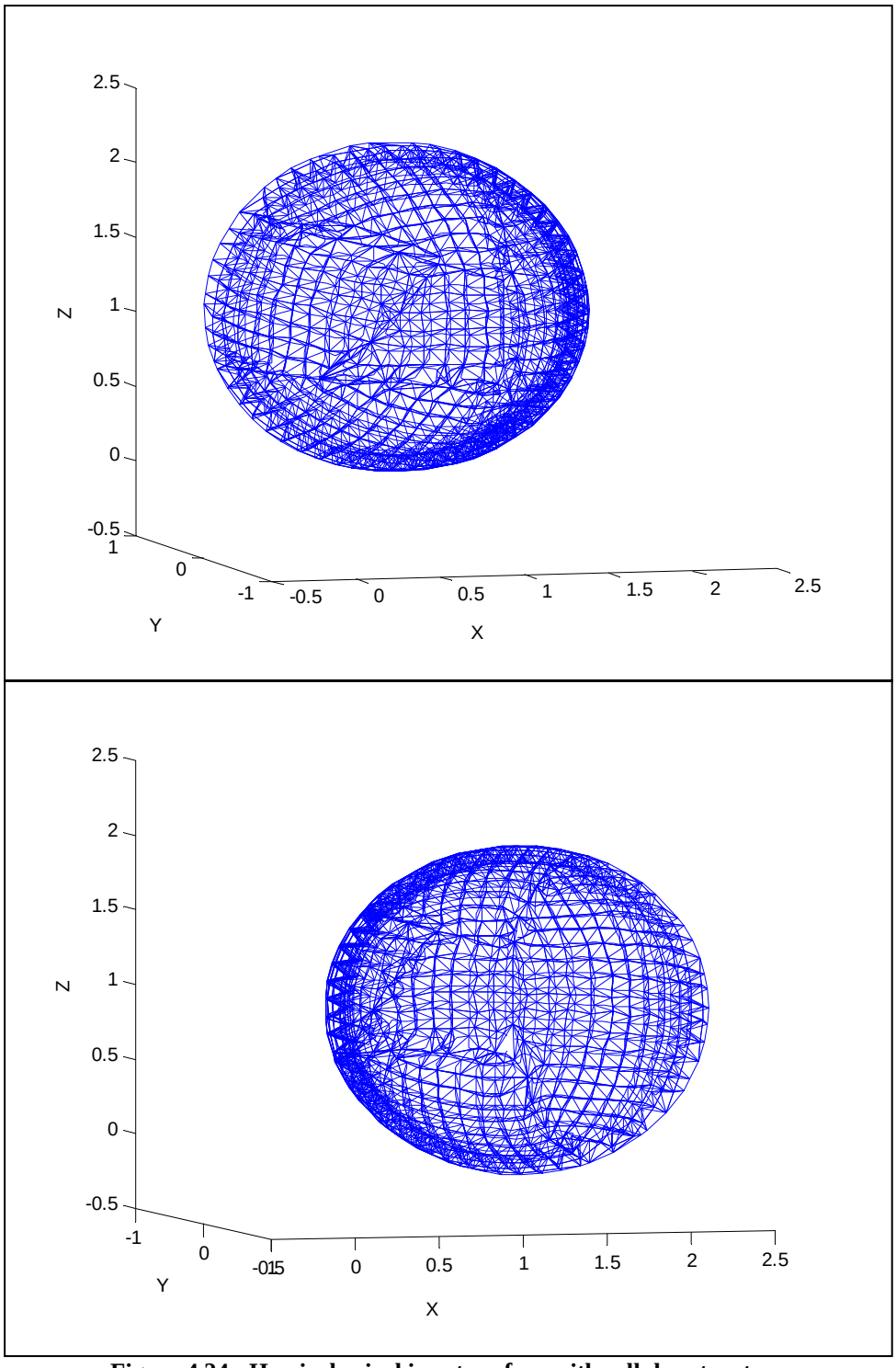


Figure 4.24: Hemispherical input surface with cellular structure

Table 4-19: Hemispherical surface offset quality

Output	Value
Fast Offset	
Time (seconds)	109
Desired Offset (in)	0.100
Minimun Offset (in)	0.09990
Maximum Offset (in)	0.1005
Mean Offset (in)	0.09998
Standard Deviation (in)	2.90E-05
Original Offset	
Time (seconds)	23699
Desired Offset (in)	0.100
Minimun Offset (in)	0.09997
Maximum Offset (in)	0.1036
Mean Offset (in)	0.10006
Standard Deviation (in)	0.00037

4.2.6 Conical surface with varying curvature

This example is similar to the spherical hemisphere example, except the curvature is not uniform. The purpose of this example is to test the function of the algorithm for a surface with varying curvature. All previous examples with curvature have featured curvature that is uniform. This example will test whether irregular curvature produces low quality results for quadrilateral elements due to the non-uniform curvature. The input surface for this example is shown in Figure 4.25.

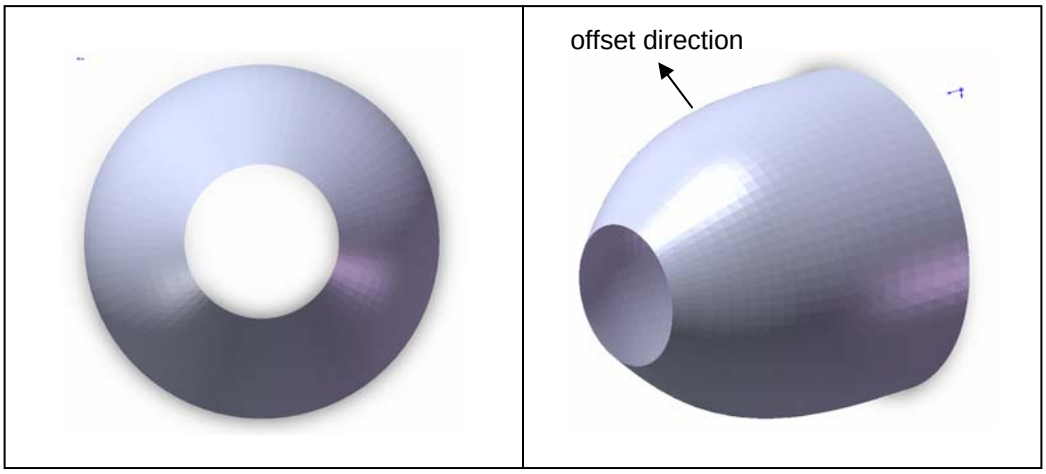


Figure 4.25: Conical surface with varying curvature

The algorithm input is shown in Table 4-20. The angle input is nearly a quarter of a revolution, so it makes sense that the input would produce four partitions for a closed conical surface. The partitions are shown in Figure 4.26. This surface also produces some jagged partition boundaries, but they are less severe than the spherical hemisphere example above.

Table 4-20: Algorithm data for conical surface example

	Geometry Description	Dimensions			Algorithm Inputs				Calculated Values	
Example	Surface	Max Radius (in)	Min Radius (in)	Height (in)	Element Side Length (in)	Offset Distance (in)	Angle Criterion (radians)	Sampling Criterion	Number of Partitions	Sample Length (in)
6	Conic	0.740	0.323	1.368	0.110	0.100	1.30	50	4	0.0296

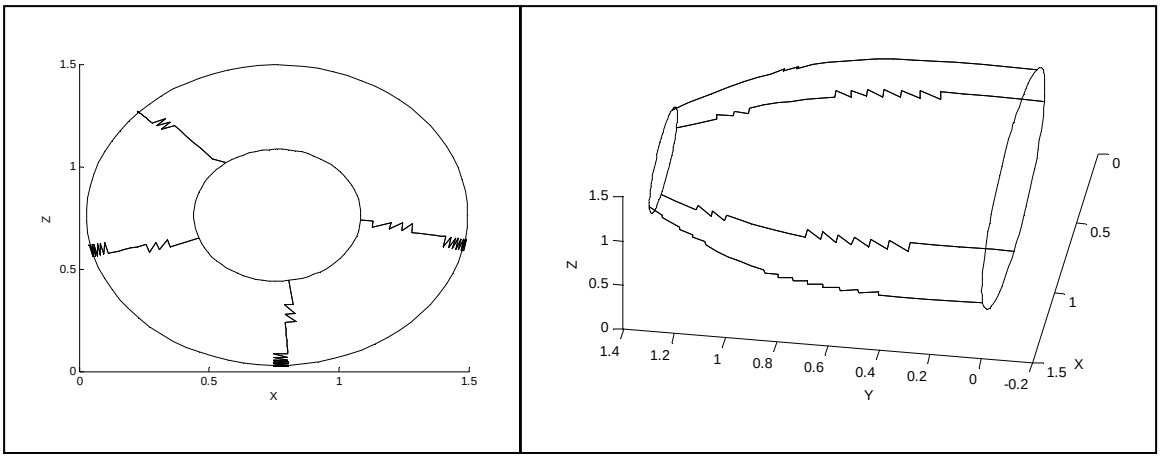


Figure 4.26: Conical input surface divided into four partitions

The mesh for the conical input surface is shown in Figure 4.27. There are some triangular and distorted elements near the boundaries, but the mesh is good for large portions of the surface. The mesh quality statistics are given in Table 4-21.

There are about 11% triangular elements. The maximum skew is 30.6 degrees, which is slightly higher than the finite element analysis cutoff of 30 degrees. This is not much over the cutoff and appears to be a single outlier. The average skew is only 5.6 degrees with a standard deviation of about three degrees. The skew of 30.6 degrees is nearly nine standard deviations above the mean. About 94% of elements have a skew below ten degrees. The average aspect ratio is 16% past square. The maximum aspect ratio is approaching the quality cutoff but is still below.

The side lengths range from 34% to 196% to the desired mesh side length. About 53% of quadrilateral element sides are within 10% of desired side length, and 72% of side lengths are within 15% of the desired side length.

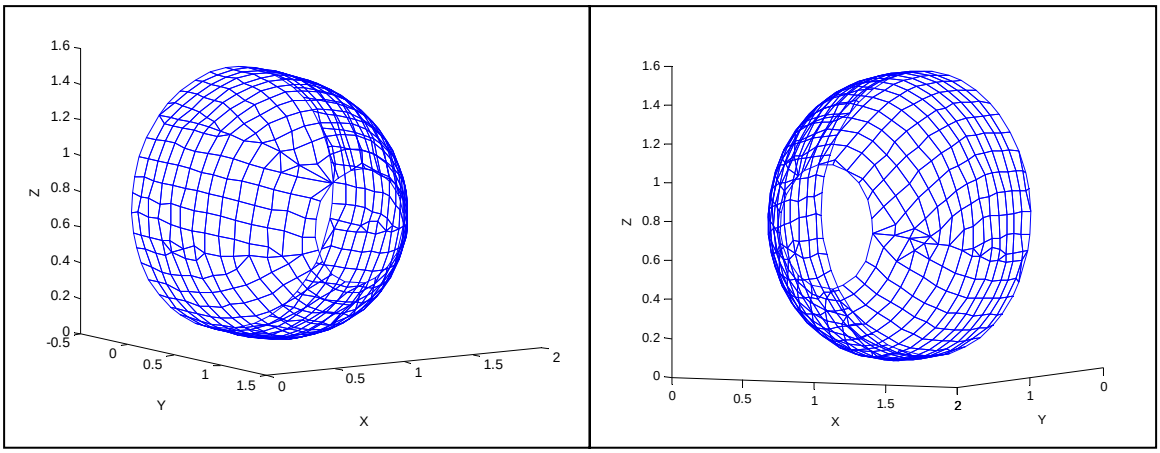


Figure 4.27: Conical input surface with mesh

Table 4-21: Conical surface mesh quality

Output	Value
Number of Quads	499
Number of Tris	62
Percentage Tris	11.1%
Quad Skew	
Min (degrees)	0.000
Max (degrees)	30.592
Mean (degrees)	5.602
Standard Deviation (degrees)	5.713
Quad Aspect Ratio	
Min	1.0010
Max	2.8397
Mean	1.1620
Standard Deviation	0.2418
Quad Side Length	
Desired (in)	0.110
Min (in)	0.0367
Max (in)	0.2162
Mean (in)	0.1074
Standard Deviation (in)	0.0152

The surface with cellular material applied is shown in Figure 4.28. The offset statistics are shown in Table 4-22. The offset result is very similar to the result for the

hemisphere example above. The mean and minimum values are almost identical for the two cases. The differences are in calculation time and the maximum value. Again, the fast offsetting method is more accurate, with a maximum value closer to the requested offset value and a smaller standard deviation for offset distances. The computation is also much faster.

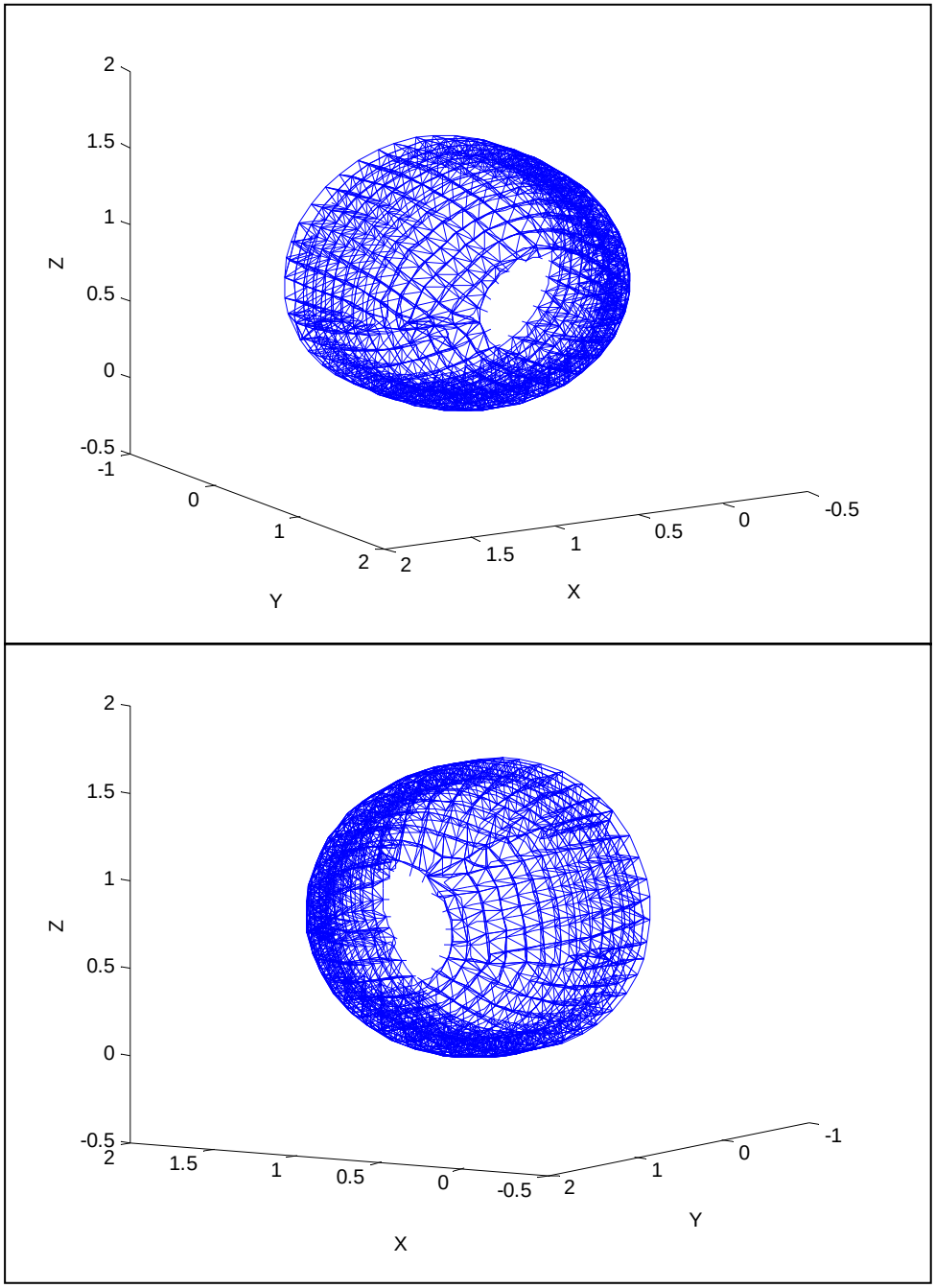


Figure 4.28: Conical input surface with cellular structure

Table 4-22: Conical surface offset quality

Output	Value
Fast Offset	
Time (seconds)	121
Desired Offset (in)	0.100
Minimun Offset (in)	0.09990
Maximum Offset (in)	0.1000
Mean Offset (in)	0.09999
Standard Deviation (in)	0.000019
Original Offset	
Time (seconds)	33584
Desired Offset (in)	0.100
Minimun Offset (in)	0.10000
Maximum Offset (in)	0.1033
Mean Offset (in)	0.10003
Standard Deviation (in)	0.00029

4.2.7 Surface with a saddle point with one partition

The purpose of this example is to test the performance of the algorithm for a surface with complex curvature. The surface used in this example is shown in Figure 4.29. The surface has a saddle point, and the curvature is not constant in any direction.

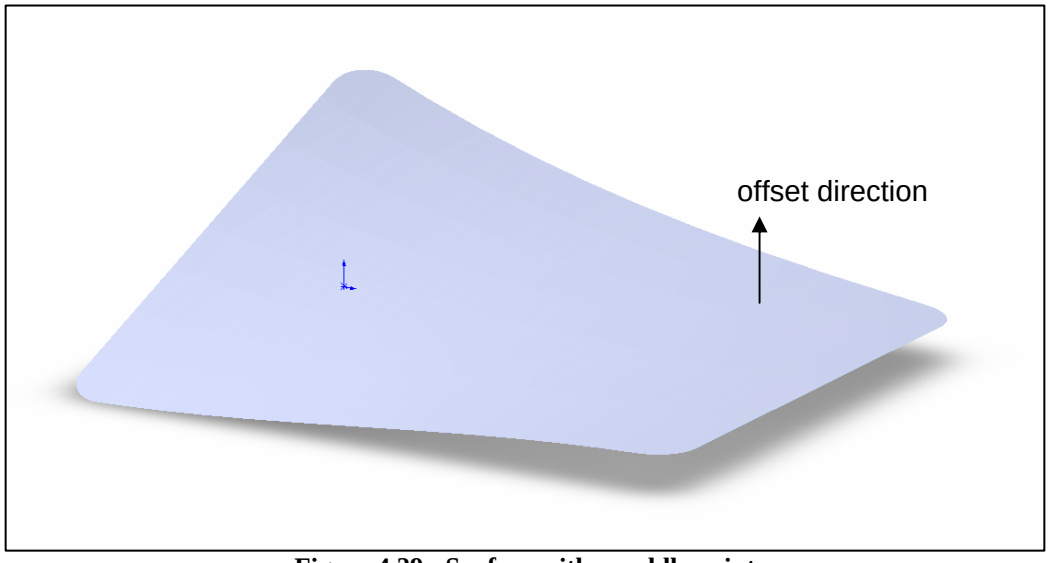


Figure 4.29: Surface with a saddle point

The input to the algorithm for partitioning is shown in Table 4-23. With this example, it is possible to have only one partition. The angle criterion of 0.5 radian produces no partitions breaks, so the input surface is itself the only partition.

Table 4-23: Algorithm data for saddle surface example

	Geometry Description	Dimensions		Algorithm Inputs				Calculated Values	
Example	Surface	Height (in)	Width (in)	Element Side Length (in)	Offset Distance (in)	Angle Criterion (radians)	Sampling Criterion	Number of Partitions	Sample Length (in)
7	Saddle	128	104	6.604	6.350	0.50	50	1	2.5492

The mesh produced for the input surface in this example is shown in Figure 4.30. This mesh has both distorted elements due to boundaries and in the center of the surface due to sampling rate. The mesh quality statistics are given in Table 4-24.

There are few triangular elements in this example. Most are located at the corners, partially because the corners of the surface are rounded. There are a few triangular elements that are the result of quality operations.

The average skew for quadrilateral elements in this example is about five degrees, with a standard deviation of 4.3 degrees. Over 88% of quadrilateral elements have a skew less than ten degrees. The average aspect ratio is 17% past square. Maximum skew and aspect ratio are below the finite element quality cutoffs.

Quadrilateral element side lengths range from 44% to 194% of the desired side length. About 51% of element side lengths are within 10% of desired length, and 70% of side lengths are within 15% of the desired length.

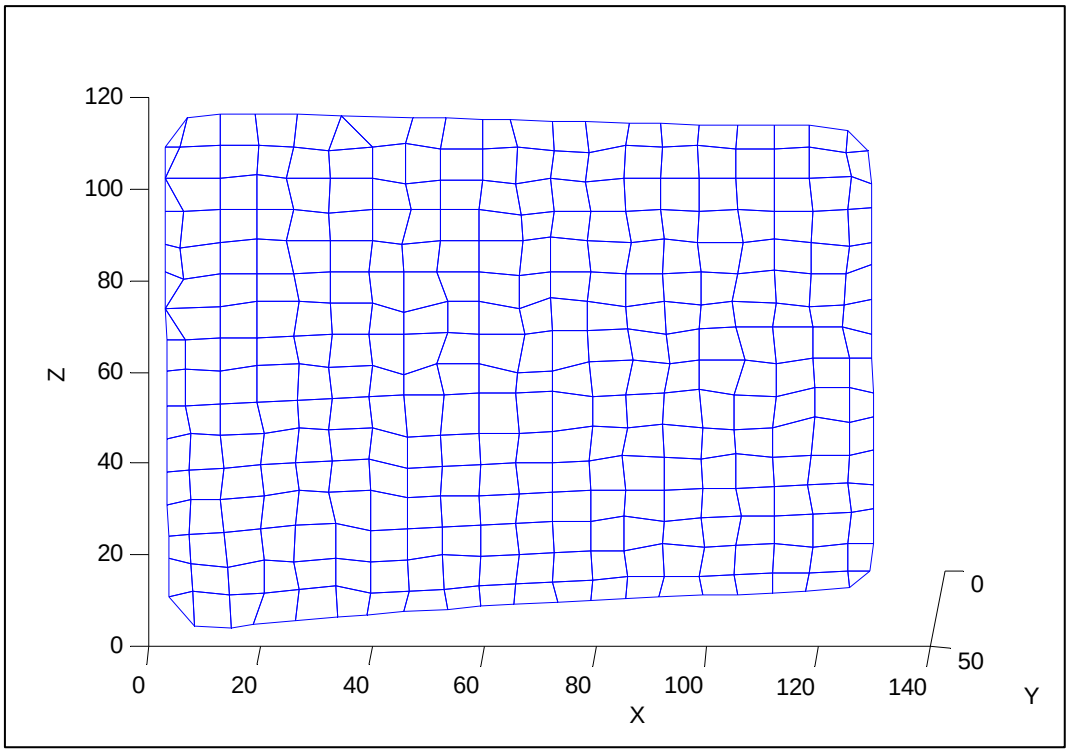


Figure 4.30: Input saddle surface with mesh

Table 4-24: Saddle surface mesh quality

Output	Value
Number of Quads	311
Number of Tris	9
Percentage Tris	2.8%
Quad Skew	
Min (degrees)	0.016
Max (degrees)	25.005
Mean (degrees)	4.870
Standard Deviation (degrees)	4.302
Quad Aspect Ratio	
Min	1.001
Max	2.414
Mean	1.173
Standard Deviation	0.020
Quad Side Length	
Desired (in)	6.604
Min (in)	2.910
Max (in)	12.788
Mean (in)	6.493
Standard Deviation (in)	0.950

The input surface with cellular material applied is shown in Figure 4.31. Offset quality statistics are given in Table 4-25. For this example, it is unclear which method produces a better result. The fast method produces a mean value that is almost exactly equal to the desired result. However, this surface has an area that is concave, so the mean value should not be exactly equal to the desired value. The fast method tends to produce a value that is slightly lower than the desired or correct value, and the original offsetting method tends to produce a value slightly higher than desire or correct for the offset.

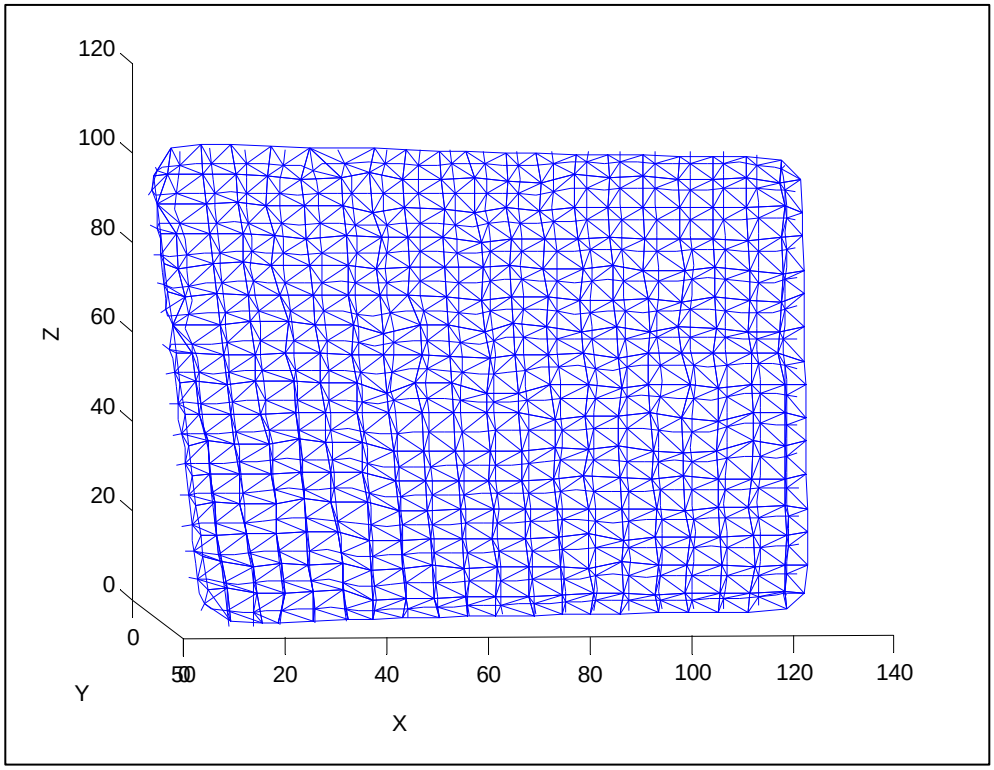


Figure 4.31: Input saddle surface with cellular material

Table 4-25: Saddle surface offset quality

Output	Value
Fast Offset	
Time (seconds)	64
Desired Offset (in)	6.350
Minimun Offset (in)	6.34990
Maximum Offset (in)	6.3517
Mean Offset (in)	6.34998
Standard Deviation (in)	0.00012
Original Offset	
Time (seconds)	1300
Desired Offset (in)	6.350
Minimun Offset (in)	6.34990
Maximum Offset (in)	6.7030
Mean Offset (in)	6.37000
Standard Deviation (in)	0.05987

4.2.8 Surface with a saddle point with two partitions

The purpose of this example is to show the effect of adding a partition boundary to a surface with complex curvature. This will show the partition that will be produced for this type of surface and the element quality effects.

The algorithm input to produce two partitions is shown in Table 4-26. The resulting partitions are shown in Figure 4.32. The partitions and partition boundary for this example are not ideal. The partitions are not approximately rectangular, especially the smaller partition. The boundary between partitions is very jagged.

Table 4-26: Algorithm data for saddle surface example, two partitions

	Geometry Description	Dimensions		Algorithm Inputs				Calculated Values	
Example	Surface	Height (in)	Width (in)	Element Side Length (in)	Offset Distance (in)	Angle Criterion (radians)	Sampling Criterion	Number of Partitions	Sample Length (in)
8	Saddle	128	104	6.400	6.350	0.29	50	2	2.5492

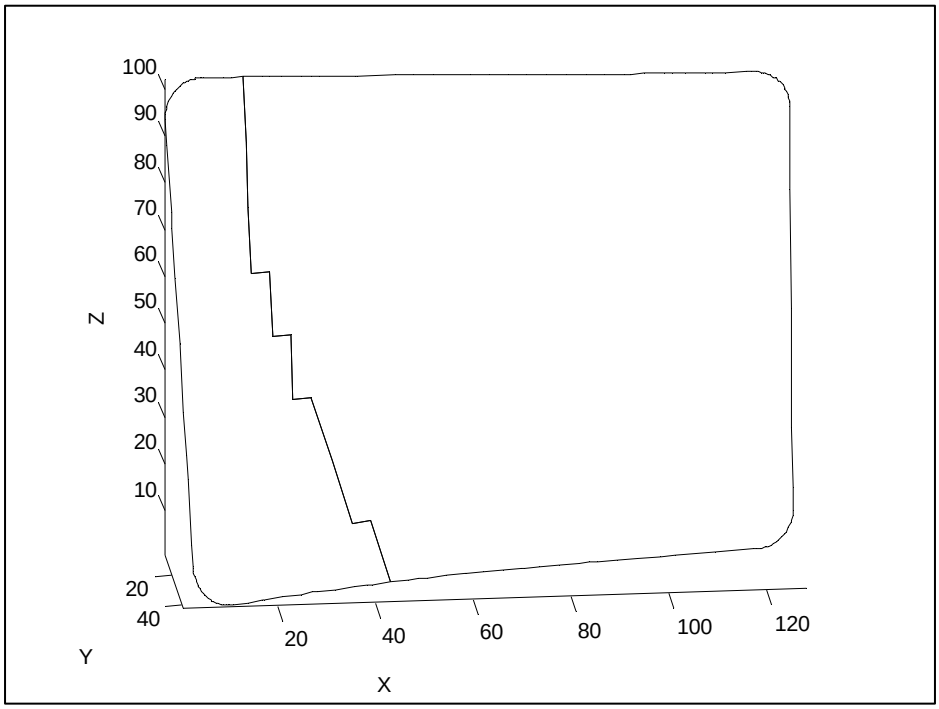


Figure 4.32: Input saddle surface divided into two partitions

The mesh for the surface in this example is shown in Figure 4.33. The element quality is poor near the boundary. There are many triangular elements along the partition boundary. Also, because of the way the mesh on the smaller partition is oriented, there are several triangular elements along the edges of the surface.

The mesh quality statistics are shown in Table 4-27 compared to the statistics for the same surface with only one partition. Surprisingly, the mesh quality for the surface with two partitions is not significantly worse, and in some cases better, than for the surface with one partition. This is generally due to poor elements being broken down into triangular elements by quality control operations, leaving only more well formulated quadrilateral elements to contribute to the mesh statistics.

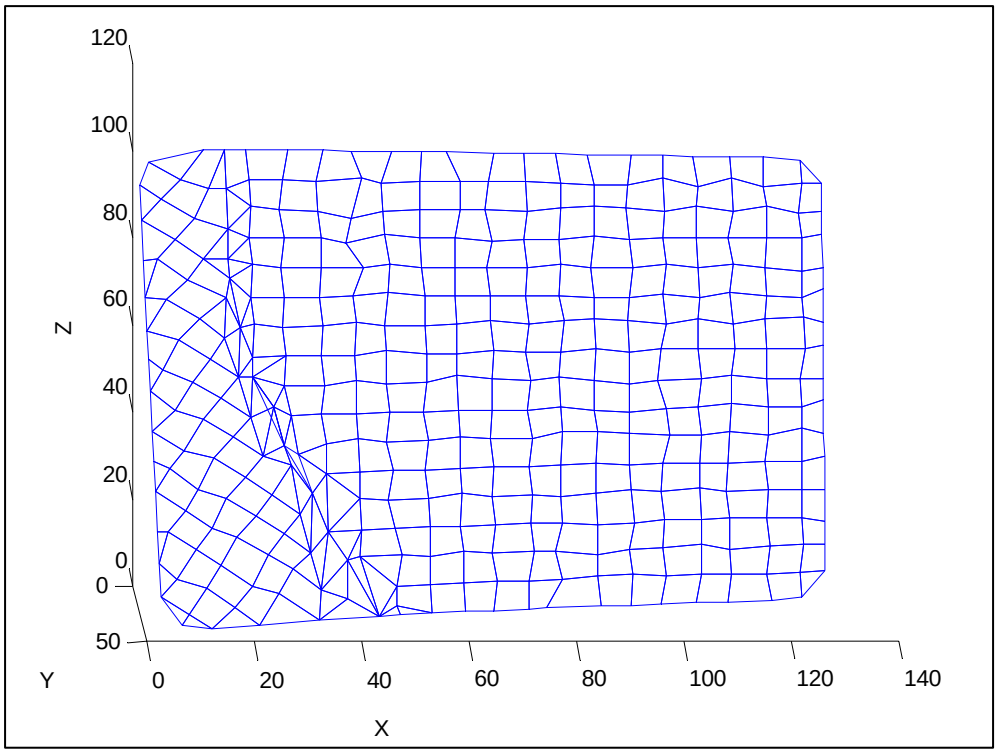


Figure 4.33: Input saddle surface with two partitions with mesh

Table 4-27: Comparison of mesh quality data for a saddle surface

	Partitions	
	1	2
Output	**Value**	**Value**
Number of Quads	311	294
Number of Tris	9	76
Percentage Tris	2.8%	20.5%
Quad Skew		
Min (degrees)	0.002	0.003
Max (degrees)	25.005	21.245
Mean (degrees)	4.870	4.812
Standard Deviation (degrees)	4.302	4.156
Quad Aspect Ratio		
Min	1.0006	1.0016
Max	2.4136	2.0271
Mean	1.1731	1.1489
Standard Deviation	0.0205	0.1665
Quad Side Length		
Desired (in)	6.604	6.400
Min (%desired)	44.06%	34.49%
Max (%desired)	193.64%	162.92%
Mean (%desired)	98.32%	99.85%
Standard Deviation (%desired)	14.38%	13.82%

The input surface with cellular material applied is shown in Figure 4.34. The area near the partition boundary is noticeably affected by the presence of many triangular elements, which is an undesirable result. However, the truss formed by triangular elements will almost always be stiffer than the accompanying quadrilateral elements, which should make the truss shown below viable for all but the most sensitive structural cases.

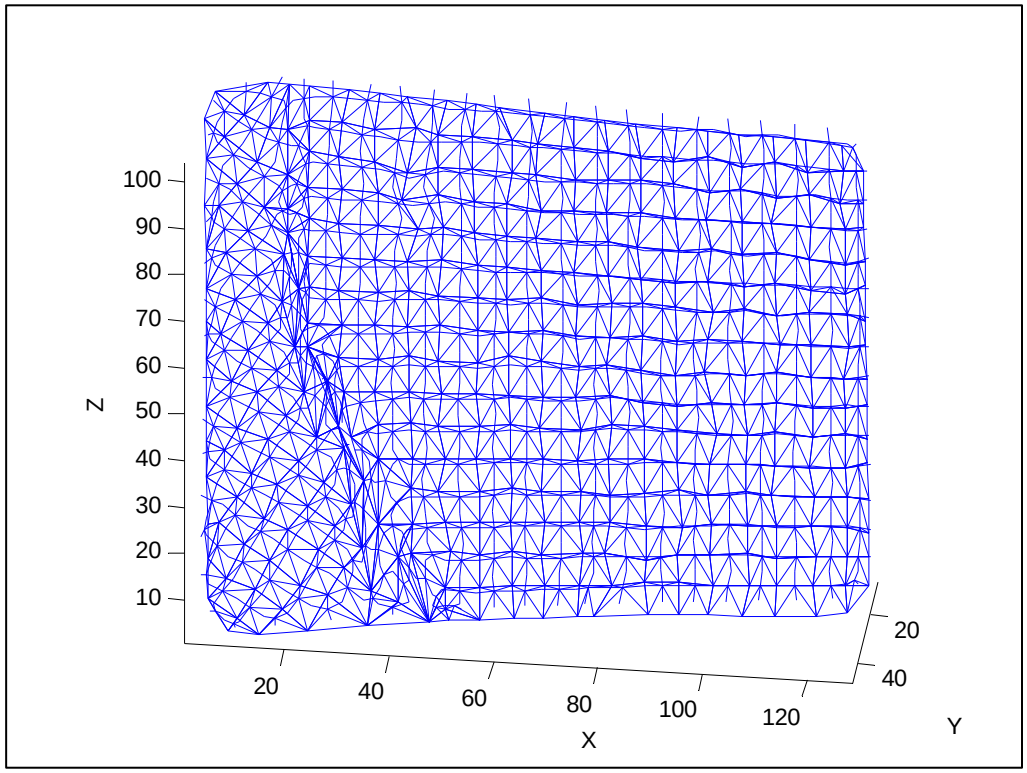


Figure 4.34: Input saddle surface with two partitions with cellular material

Table 4-28: Offset quality for saddle surface with two partitions

Output	Value
Fast Offset	
Time (seconds)	68
Desired Offset (in)	6.350
Minimun Offset (in)	6.3499
Maximum Offset (in)	6.3524
Mean Offset (in)	6.349996
Standard Deviation (in)	2.30E-04
Original Offset	
Time (seconds)	1285
Desired Offset (in)	6.350
Minimun Offset (in)	6.34993
Maximum Offset (in)	6.7191
Mean Offset (in)	6.37546
Standard Deviation (in)	0.06993

4.3 Sensitivity

Some results from the examples above are difficult to directly interpret because many factors are changing, such as mesh size, surface size and shape, sampling rate, etc. The purpose of the following studies is to isolate mesh quality effects from a single input.

4.3.1 Mesh size sensitivity

To test the effect of mesh size on quality, the same surface with the same sampling density and number of partitions is fit with different mesh sizes. The surface and partitioning is the same as in the example in Section 4.2.3. The sampling rate used is 40. The resulting mesh is shown in Figure 4.35. Figure 4.35a shows mesh with a desired quadrilateral side length of 0.22 length units. Figure 4.35b shows mesh with desired side length of 0.32. Figure 4.35c shows mesh with desired side length of 0.42, and Figure 4.35d show mesh with desired length of 0.60.

The result shows two different trends. First, when the mesh is very small, the sampling ratio has a significant effect on mesh quality. The mesh in Figure 4.35a is mostly quadrilateral mesh, but many of the elements that are not affected by boundaries are odd shapes and sizes, compared to mesh in Figure 4.35b, c, and d.

The second trend is that for the large mesh in this example, some boundary problems arise and a number of triangular elements are formed. It is likely that with a higher sampling rate, the larger mesh would not have some of the partition boundary problems. The result will take longer with a very high sampling rate, however. For larger mesh sizes, the percentage of elements that are likely to have quality problems due to surface and partition boundaries is higher than for smaller mesh sizes, also.

From Figure 4.35 and the statistics given in Table 4-29, it can be seen that the best result is for the case with 0.42 size mesh (c). The case for a mesh size of 0.32 also gives good results. This study shows that for a given geometry and sampling rate, not all mesh sizes will give a good result. Small elements that give a poor result are mostly due to sampling rate. Larger elements that give a poor result are due more to boundary effect.

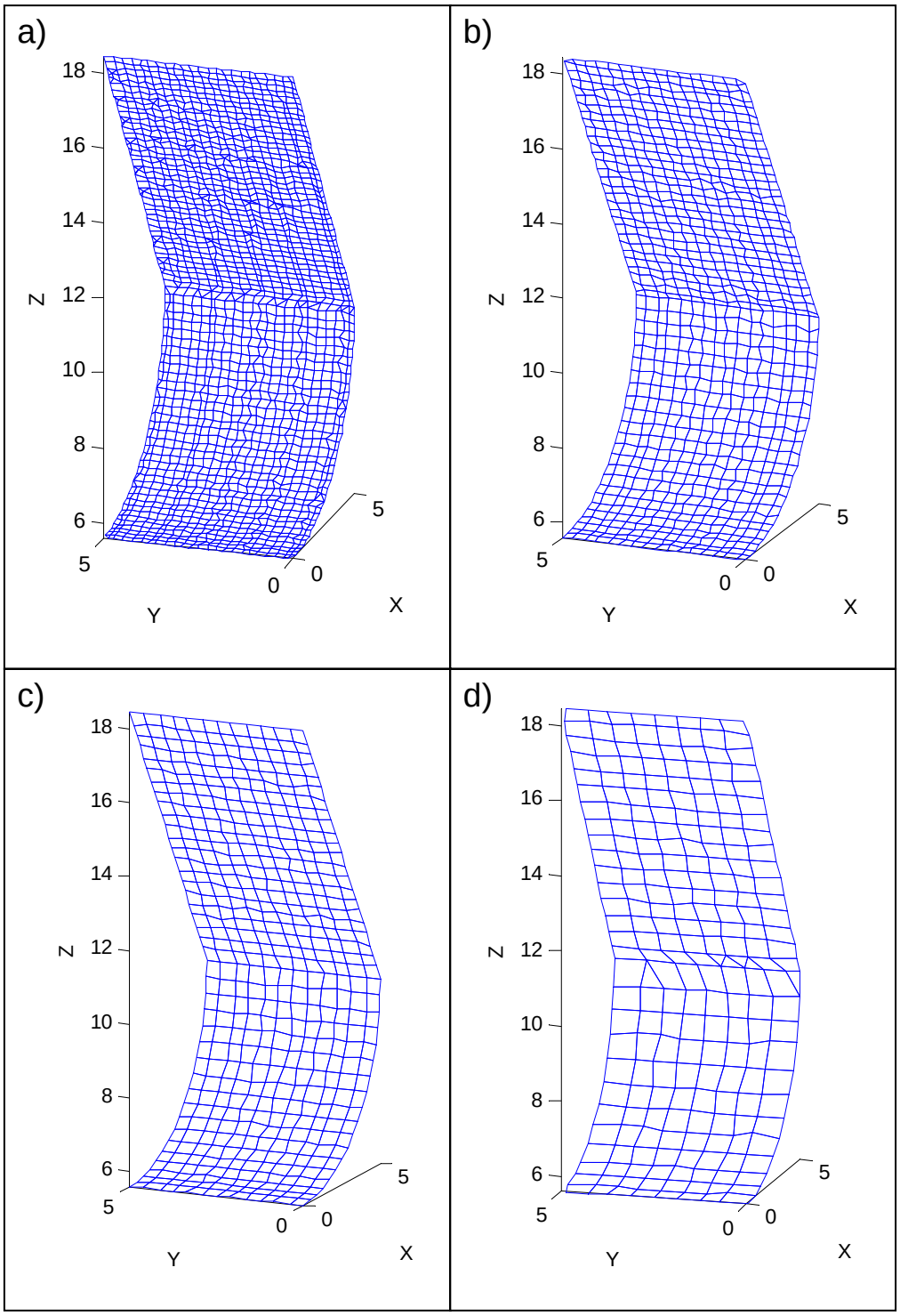


Figure 4.35: Surface mesh with increasing size

Table 4-29: Mesh quality for varying mesh size

	Desired Quad Side Length			
	0.22	0.32	0.42	0.6
Output	Value	Value	Value	Value
Number of Quads	1783	862	480	221
Number of Tris	97	2	0	6
Percentage Tris	5.2%	0.2%	0.0%	2.6%
Quad Skew				
Min	0.00	0.00	0.00	0.00
Max	42.35	17.32	14.06	38.70
Mean	7.62	4.50	3.12	4.14
Standard Deviation	8.53	3.33	2.47	5.79
Quad Aspect Ratio				
Min	1.0010	1.0019	1.0041	1.0047
Max	2.9787	2.1213	1.3533	1.7235
Mean	1.2507	1.1617	1.1258	1.1408
Standard Deviation	0.2637	0.1462	0.0756	0.1157
Quad Side Length				
Min (%desired length)	37.66%	35.44%	78.23%	62.50%
Max (%desired length)	170.45%	136.92%	126.59%	145.83%
Mean (%desired length)	100.09%	99.25%	100.94%	103.75%
Std. Dev. (%desired length)	19.15%	15.08%	10.88%	1.16%

4.3.2 Mesh with varying sampling rate

In this study, the number of partitions, mesh size, and input surface are all kept the same as the sampling rate is decreased. The surface is the same as in the example in Section 4.2.3 above. As in Section 4.2.3, there surface is divided into four partitions. The mesh size used is 0.425 inches. The mesh results are shown in Figure 4.36 and the quality statistics are shown in Table 4-30. It is clear visually and from the statistics that as the sampling rate decreases, the mesh quality decreases. Figure 4.36a corresponds to a sampling criterion of 60. This means that the length of the input part is divided by 60 to establish the universal size criterion used to sample the input face. Figure 4.36b corresponds to a sampling criterion of 50. Figure 4.36c corresponds to a sampling criterion of 30. Figure 4.36d corresponds to a sampling criterion of 20. The decrease in sampling rate not only leads to the formation of poor quadrilateral elements, but

triangular elements are also generated by the quality checks as the quadrilateral quality falls.

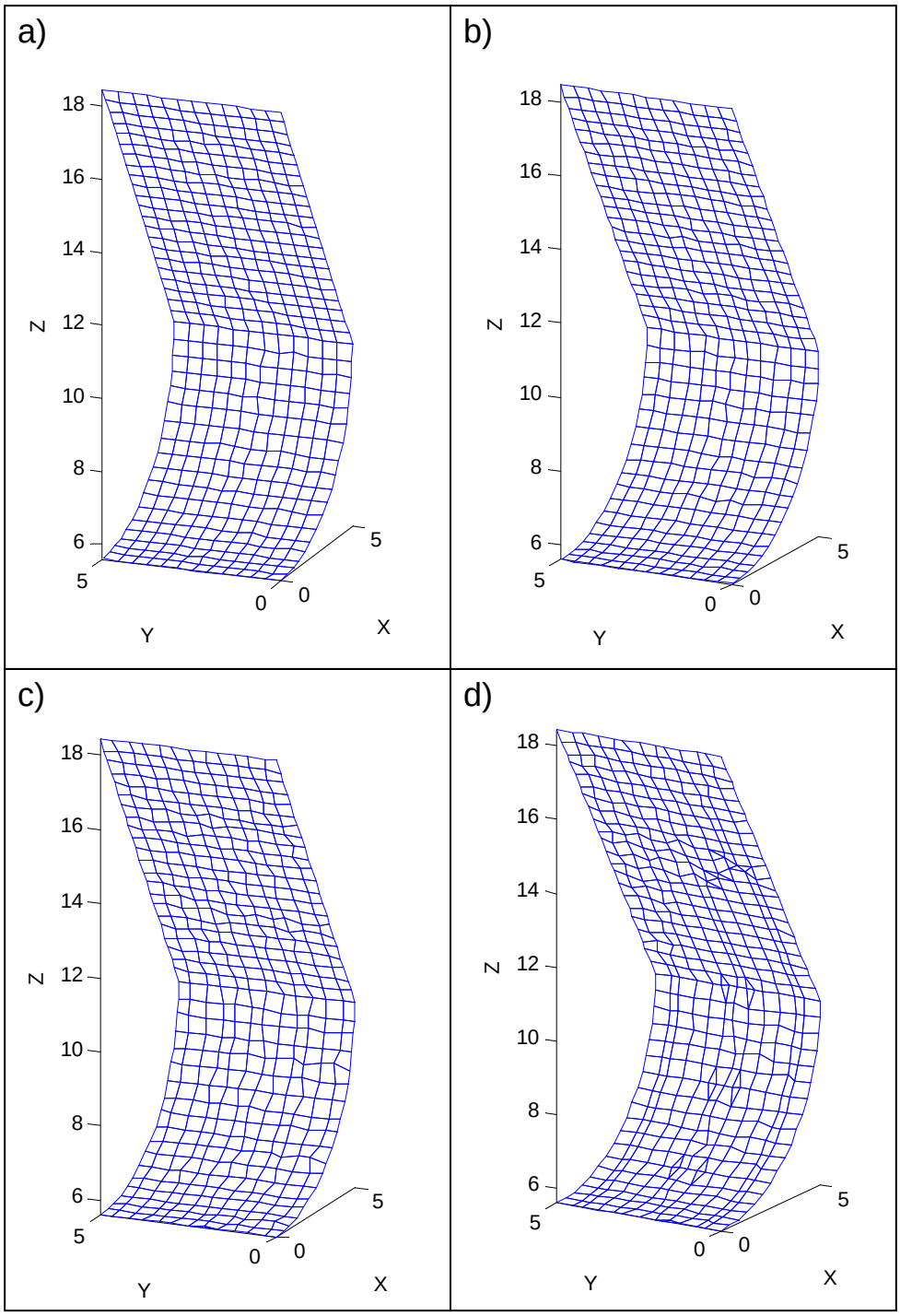


Figure 4.36: Surface mesh with decreasing sampling rate

For a sampling rate of 60 for this example, there are no elements with a skew over ten degrees, and the average aspect ratio is less than 10% past square. The mean side length is almost exactly the requested value, and the maximum and minimum values are only 20% above or below the desired length.

Table 4-30: Quality for mesh with varying sampling criteria

	Sampling Criteria			
	20	30	50	60
Output	**Value**	**Value**	**Value**	**Value**
Number of Quads	474	480	480	480
Number of Tris	26	2	0	0
Percentage Tris	5.2%	0.4%	0.0%	0.0%
Quad Skew				
Min (degrees)	0.00	0.00	0.00	0.00
Max (degrees)	32.91	18.26	14.18	9.48
Mean (degrees)	5.21	4.36	3.11	2.43
Standard Deviation (degrees)	5.05	3.34	2.30	2.06
Quad Aspect Ratio				
Min	1.0139	1.0007	1.0029	1.0049
Max	2.0264	1.6864	1.2668	1.2342
Mean	1.2469	1.1573	1.0868	1.0688
Standard Deviation	0.2626	0.1232	0.0607	0.0508
Quad Side Length				
Min (%desired length)	0.2175	0.2909	0.3513	0.3397
Max (%desired length)	0.5951	0.5818	0.5097	0.5040
Mean (%desired length)	0.4252	0.4254	0.4236	0.4232
Std. Dev. (%desired length)	0.0778	0.0620	0.0354	0.0264

Table 4-31 shows the results of the examples presented in Section 4.2 with sampling rates to show the effect of sampling rate on algorithm performance. The results do not follow a direct trend indicating that a higher sampling rate (smaller sample size) produces better results. This indicates that sampling rate, mesh size, number of partitions, etc. all contribute to mesh quality. The results in Table 4-30 indicate that with all other values held constant, a higher sampling rate does produce better results, as would be expected.

Table 4-31: Effect of sampling size on quality

	Geometry						
	Conic	Concave	Flat	Saddle	Hemishere	Cylinder	Convex
Output	**Value**	**Value**	**Value**	**Value**	**Value**	**Value**	**Value**
Sample Size	0.0296	0.0250	0.0557	0.0400	2.5492	0.1250	0.1426
Sample Size (%desired length)	26.91%	29.41%	31.83%	36.36%	38.60%	39.06%	39.83%
Quad Skew							
Min (degrees)	1.50E-05	0	0.0214	0.0012	0.0163	0	0
Max (degrees)	30.592	22.997	20.403	39.927	25.005	17.325	15.820
Mean (degrees)	5.602	2.462	4.218	6.927	4.870	4.503	3.343
Standard Deviation (degrees)	5.713	2.826	3.861	6.757	4.302	3.334	2.897
Quad Aspect Ratio							
Min	1.0010	1.0029	1.0054	1.0013	1.0006	1.0019	1.0034
Max	2.8397	3.0799	2.1086	2.2816	2.4136	2.1213	1.6953
Mean	1.1620	1.2318	1.2085	1.1679	1.1731	1.1617	1.1726
Standard Deviation	0.2418	0.2529	0.2433	0.1877	0.0205	0.1462	0.1343
Quad Side Length							
Min (%desired length)	33.36%	35.32%	46.97%	35.41%	44.06%	35.44%	68.22%
Max (%desired length)	196.52%	152.74%	128.72%	211.60%	193.64%	136.92%	136.47%
Mean (%desired length)	97.60%	97.18%	97.59%	99.39%	98.32%	99.25%	101.11%
Std. Dev. (%desired length)	13.78%	15.52%	15.05%	16.55%	14.38%	15.08%	14.75%

4.4 Summary

This chapter presents several example problems that test the performance of the cellular material design method presented in Chapter 3. The examples vary in number and complexity of partition boundaries and severity and type of curvature. A summary of the examples is given below. The examples were chosen to represent basic types of geometry that could be present on arbitrary input surfaces.

Figure 4.37: Summary list of example problems

Example	Description	Features
1	Two planar surfaces meeting at an angle	Simple geometry, single partition boundary
2	Half Cylinder	Uniform curvature in one direction, multiple partition boundaries
3	Planar surface meeting a concave surface	Dissimilar geometries interfacing, multiple partition boundaries
4	Planar surface meeting a convex surface	Dissimilar geometries interfacing, multiple partition boundaries
5	Circular Hemisphere	Uniform curvature in multiple directions, multiple partition boundaries
6	Irregular Conic	non-uniform curvature in one direction, uniform curvature in one direction
7	Surface with a saddle point, one partition	Complex curvature, single partition
8	Surface with a saddle point, two partitions	Complex curvature, multiple partitions

The quality of the mesh on the input surface and the offset distance are evaluated for each example. The surface mesh is evaluated for percentage of triangular elements, skew, aspect ratio, and side length. Skew and aspect ratio are quality measures borrowed from finite element analysis. A standard good mesh for analytical purposes is skew less than 30 degrees and aspect ratio less than three. These quality measures are used as a guideline for mesh quality. The side lengths should be as close to the desired length as possible. The offset distance is measured from each corner of each element on the input surface to the corresponding offset element corner. This value should match the input desired offset distance.

The first goal for the method presented in this thesis is to produce surface mesh that consists of square elements. The results show that the surfaces in the examples can be effectively meshed with mostly quadrilateral elements. For some cases, there is a relatively high percentage of triangular elements at partition boundaries. Even in these cases, there are large areas of good quality elements away from the boundaries. There were very few occurrences of elements that exceeded the aspect ratio or skew quality limits, which measure how far from square the elements have deviated. In all cases

where a maximum value for skew or aspect ratio exceeded the limit, the value is well above the mean and likely represents a single element among the hundreds applied to the surface. The average side lengths for all examples are within 3% of the input length. This demonstrates that good quality mesh that is close to square can be created using the method from Chapter 3.

The second goal is to create an offset mesh that is offset a specified distance from the input surface. Error values for average offset lengths for all examples are less than 1% of the desired offset for both methods. The method that offsets only element corners is much faster and produces results that are comparable or preferable. These results show that both methods discussed in Chapter 3 are effective in creating an offset mesh that provides a volume to fill with cellular material primitives.

The final goal is to insert cellular material primitives into the three-dimensional mesh created by the surface and offset meshes. The examples above show that a continuous, oriented cellular material design can be created in all cases. The original design for cellular material was developed for cubic primitives. However, the triangular prisms formed by triangular element can be fit with complimentary cellular material designs to maintain continuity with cubic primitives. Physically, the corresponding cellular material in triangular prisms is likely stiffer than in the cubic primitive, so it would not cause weaknesses in the structure. Triangular elements are undesirable for uniformity and optimization, but their presence can be accommodated. An acceptable cellular material design was produced for all example problems.

5 Conclusion

Advances in rapid manufacturing technologies have expanded its possible applications. As these new manufacturing tools evolve, so too must tools for the design of new structures. The purpose of this thesis is to present a method for designing cellular material, which can only be produced with rapid manufacturing technology and for which design tools are unavailable.

Chapter 2 is a literature and patent review. Two-dimensional and three-dimensional meshing and offsetting methods developed for other applications are studied for possible use. Regular cellular material designs and manufacturing methods are also investigated in order to understand how these materials are currently made and designed.

Chapter 3 presents a new method for cellular material design. An STL representation of a surface is the input. A volume is created which is filled with cellular material primitives to create a continuous cellular structure design that entirely covers the surface. It is a completely automated process that can produce cellular material of almost any design.

Chapter 4 is a collection of example problems that demonstrate the function of the design tool for surfaces of various types of geometry. The examples show that good quality surface mesh can be created, the offsetting function is accurate, and that a continuous cellular structure design is created for each case.

This final chapter summarizes conclusions and accomplishments and offers recommendations for future work.

5.1 Conclusions

The goal of the research presented above is to develop a tool to make the use of meso-scale cellular structure more feasible for a wide range of applications. Three distinct goals were set forth in the opening chapter. The first goal was to apply square (quadrilateral) mesh to the input surface. The second goal was to create a complimentary mesh on a surface offset from the original in order to create a volume in which to place cellular material. The third goal was to insert cellular material primitives to create a cellular structure design for the input surface. The following are conclusions drawn from the research presented in this thesis.

5.1.1 Conclusion One: Surface Mesh

Several example problems with detailed results were presented in Chapter 4 to demonstrate the meshing function. Some statistics that pertain specifically to the surface mesh quality are summarized in Table 5-1. From the table, it can be seen that most cases produce very few triangular elements. Some cases produce a relatively large number of triangular elements. These cases pertain to examples where complicated partition boundaries are present. It should be noted that mesh sizes and sampling rates were chosen to be realistic, to produce a result efficiently, and that could be reasonable viewed in a printed figure. If a smaller mesh size or higher sampling rate were chosen for the surface the percentage of triangular elements could likely be decreased.

Also shown in Table 5-1 are average values for skew, aspect ratio and side length for the quadrilateral elements in each example. Average side lengths are all within 3% of

the input desired length, and average skew for all examples is less than ten degrees. Aspect ratio, like percentage of triangular elements, is considerably affected by the boundaries, and the inputs for the examples presented were not optimized for aspect ratio. Still, all of the example aspect ratios are well below three, with few above 1.17.

Table 5-2 shows results that have been improved by increasing the sampling rate. These results show that very good square mesh can be produced with a high sampling rate.

Table 5-1: Surface mesh quality summary

Surface Geometry	Quads	Tris	%Tris	Avg. Skew (degrees)	Avg. Aspect Ratio	Avg. Side Length (%desired)
Two Planar Surfaces	466	2	0.42%	4.22	1.21	97.59%
Half Cylinder	1153	9	0.77%	2.46	1.23	97.18%
Planar Surface/Concave Surface	862	2	0.23%	4.50	1.16	99.25%
Planar Surface/Convex Surface	614	2	0.32%	3.34	1.17	101.11%
Hemisphere	480	136	22.08%	6.93	1.17	99.39%
Conic	499	62	11.05%	5.60	1.16	97.60%
Saddle Surface, One Partition	311	9	2.81%	4.87	1.17	98.32%
Saddle Surface, Two Partition	294	76	20.54%	4.81	1.15	99.85%

Table 5-2: Quality summary for high sampling rate

Surface Geometry	Quads	Tris	%Tris	Avg. Skew (degrees)	Avg. Aspect Ratio	Avg. Side Length (%desired)
Planar Surface/Concave Surface (60)	480	0	0.00%	2.43	1.07	100.04%

Therefore, the goal of creating quadrilateral mesh for the input surface is reasonably met.

5.1.2 Conclusion Two: Offsetting

Examples demonstrating the offsetting function are given in Chapter 4. Several examples have straight-forward offset surfaces, and those results show that the offset is very accurate. Average offset data for the examples from Chapter 4 are given in Table 5-3. All results are within 1% of the desired value. The table also shows the time required to run each of the offset methods. The Fast offset is much faster, and the results are closer to the requested value for all examples, but often not by a significant amount. Therefore, not only is the offset calculated accurately, but with the faster method developed during this research it is also efficient.

Table 5-3: Offset data summary

Surface Geometry	Offset Time Fast Method (seconds)	Offset Time Slow Method (seconds)	Average Offset Fast Method (%Desired)	Average Offset Slow Method (%Desired)
Two Planar Surfaces	53	584	100.54%	100.71%
Half Cylinder	1352	286137	99.991%	99.994%
Planar Surface/Concave Surface	195	2629	100.05%	100.48%
Planar Surface/Convex Surface	118	1607	100.20%	100.38%
Hemisphere	109	23699	99.98%	100.06%
Conic	121	33584	99.99%	100.03%
Saddle, one partition	64	1300	99.9997%	100.31%
Saddle, two partitions	68	1285	99.99994%	100.40%

5.1.3 Conclusion 3: Cellular material

For each of the examples presented in Chapter 4, continuous cellular structure was successfully created. The desired pattern of cellular material was created to fit a cubic element. However, it was shown that a similar pattern could also be designed to fit the triangular prism elements generated that would maintain continuity with neighboring

structure. The cellular material design for the triangular prisms is likely stiffer than the corresponding structure for cubic elements and should not represent a structural weakness. There were no orientation or continuity issues present for any of the examples.

5.2 Contributions

The major contribution of this thesis is a tool that creates a cellular structure design for an input surface. The method is described in Chapter 3, and examples are presented in Chapter 4. Some of the major accomplishments are listed below.

5.2.1 System of operations

As described in Chapter 2, meshing and offsetting methods have been previously developed that can produce the same output as the steps of the method developed in this thesis. However, because the existing methods are not specifically developed for cellular material design, they often do not produce results that meet the objectives of this application. Meshing tools for finite element analysis are concerned with analytical results instead of uniformity of shape. The meshing method developed in this thesis is designed to produce large areas of essentially square elements. Some elements that deviate from square or triangular elements are produced, but these are generally only on partition boundaries. Also, given only an input surface, several different methods would have to be combined to get the same result as the method presented in this thesis, and the output of a commercial finite element tool might be difficult to combine with output from

some other offsetting tool, etc. Therefore, the system of operations put together to coordinate the different steps with the express purpose of creating cellular material is a major contribution of this research.

5.2.2 Efficient offsetting

Another major contribution of this research is an efficient offsetting technique. The original offsetting technique proposed for use in the cellular design method described in this thesis was borrowed from an existing method [14]. The processing time of this borrowed method was sometimes several hours or days, even for relatively low sampling rates. The new method developed specifically to only offset element corners cut the processing time considerably. This makes it a more viable solution.

5.3 Future work

The method presented in this thesis produces good results. However, some areas for improvement have been identified.

5.3.1 Element quality

A common occurrence after boundary matching and the second quality check is a group of adjacent triangular elements that were produced by different operations. The matched mesh shown in Figure 5.1 has a triangular element next to an element with a large interior angle. The corrected mesh shows that the quality check corrected the

interior angle creating two adjacent triangular elements that combine to form a well shaped quadrilateral element. A relatively simple way to improve mesh quality is to combine adjacent triangular faces, as shown in the figure. This could be easily incorporated to the current process and would not significantly increase calculation time.

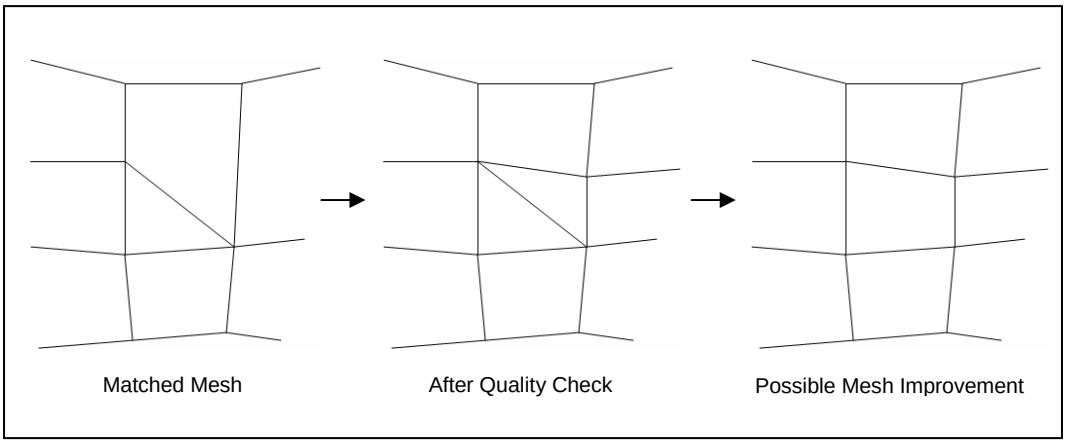


Figure 5.1: Possible mesh correction

5.3.2 Partition orientation

Currently, during the flattening procedure, each partition is reoriented before flattening. Depending on the normal vectors of each face in the partition, the partition will rotate to varying degrees. The mesh in Figure 5.2 represents two partitions that were rotated different amounts during flattening. As a result, the two meshes do not match well, and the mesh on the far left is not in line with the boundaries of the surface. This is an undesirable result. To correct this, an orientation control could be added to prevent unwanted rotation.

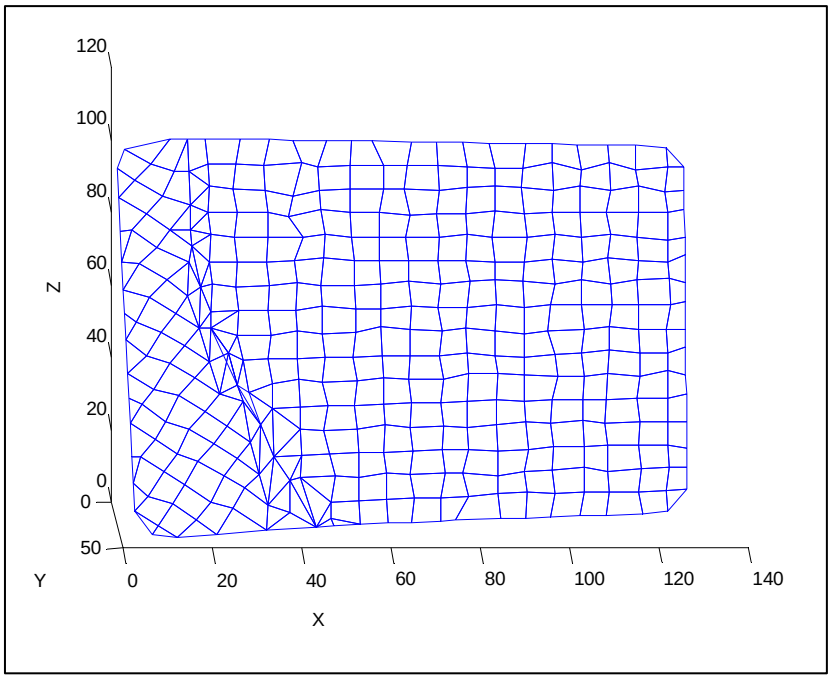


Figure 5.2: Rotated partition example

5.3.3 Cellular material library

There already exist many cellular material designs that are good for different applications. Currently, the method for cellular material design described in this thesis references a pattern for the desired design, but each time a different pattern is desired the reference to the pattern must be manually changed. Also, a comprehensive collection of patterns formatted for input to the method described in this thesis has not been compiled. Therefore, a library of different cellular material patterns that could easily be referenced is a desirable improvement.

5.3.4 FEM interface

For many applications, some analysis may be performed on the cellular structure before production. Finite element analysis would be the most likely tool to check the structure. Finite element pre-processors are also good for visualization because elements can be grouped, hidden, etc. Therefore a goal of future work is to add a function to write out finite element input for some of the more prominent commercial tools on the market.

5.4 Closure

Rapid manufacturing allows the generation of structure that was previously impossible. The freedom to design almost any shape or size imaginable opens up a world of possibilities for new forms and functions. Now more than ever, specialized tools are needed to help take advantage of the unique opportunities offered by these new manufacturing techniques. That is the motivation behind the research. It is hoped that the tool developed herein can serve to make customized cellular structure a realistic solution for many applications.

REFERENCES

1. McCartney J, Hinds BK, Seow BL. The flattening of triangulated surfaces incorporating darts and gussets. Computer-Aided Design, 1999, 31:249-260

2. Tam A, Joneja A, Tang K, Yao Z. A surface Development Method with Application in Footwear CAD/CAM. Computer-Aided Design & Applications, 2007, Vol. 4, Nos. 1-4, pp. 67-77

3. Zhong Y, Xu B. A physically based method for triangulated surface flattening. Computer-Aided Design, 2006, 38:1062-1073

4. Barry J. Quadrilateral mesh generation in polygonal regions. Computer-Aided Design, 1995, Vol. 27, No. 3, pp. 209-222

5. Park C, Noh J, Jang I, Kang J. A new automated scheme of quadrilateral mesh generation for randomly distributed line constraints. Computer-Aided Design, 2007, 39:258-267

6. Owen SJ. A survey of unstructured mesh generation technology. Proceedings of the 7th International Meshing Roundtable, Sandia National Laboratories, 1998. p. 239-267

7. Lo SH. Finite element mesh generation and adaptive meshing. Progress in Structural Engineering and Materials. 2002, 4:381-399

8. Lai M, Benzley S, White D. Automated hexahedral mesh generation by generalized multiple source to multiple target sweeping. International Journal for Numerical Methods in Engineering. 2000, 49:261-275

9. Kawamura Y, Islam M, Sumi Y. A strategy of automatic hexahedral mesh generation by using an improved whisker-weaving method with a surface mesh modification procedure. Engineering with Computers, 2008, 24:215-229

10. Becker TD, Meyers RJ. Seams and wedges in plastering: A 3-D hexahedral mesh generation algorithm. Engineering Computations. 1993, 9:83-93

11. Owen SJ. Hex-dominant mesh generation using 3D constrained triangulation. Computer-Aided Design. 2001, 33:211-220

12. Zhang H, Zhoa G, Ma X. Adaptive generation of hexahedral element mesh using an improved grid-based method. Computer-Aided Design, 2007, 39:914-928

13. Lee YK, Yang DY. A grid-based approach to non-regular mesh generation for automatic remeshing with metal forming analysis. Communications in Numerical Methods in Engineering. 2000, 16:625-635

14. Chen Y, Wang H, Rosen D, Rossignac J. A Point-Based Offsetting Method of Polygonal Meshes.

15. Queheillalt D, Murty Y, Wadley H. Mechanical properties of an extruded pyramidal lattice truss sandwich structure. Scripta Materialia, 2008, 58:76-79

16. Sypeck D, Wadley H. Cellular Metal Truss Core Sandwich Structures. Advanced Engineering Materials, 2002, 4, No. 10, pp. 759-764

17. Daily, Carl. US Patent No. 6170560, 2001

18. Jones, Ronald. US Patent No. 6630093, 2003

19. Gervasi, Vito. US Patent No. 6641897, 2003

20. Wang H, Rosen D. Parametric Modeling Method for Truss Structures. Proceedings of ASME Design Engineering Technical Conferences, 2002

21. Jacobs, Paul F. Rapid Prototyping & Manufacturing: Fundamentals of Stereolithography. Society of Manufacturing Engineers, Dearborn, MI. 1992

22. Varady T, Facello M, Terek Z. Automatic extraction of surface structures in digital shape recognition. Computer-Aided Design, 2002, 39:379-388

23. Razdan A, Bae M. A hybrid approach to feature segmentation of triangle meshes. Computer-Aided Design, 2003, 35:783-789

24. Mangan A, Whitaker R. Partitioning 3D Surface Meshes Using Watershed Segmentation. IEEE Transactions on Visualizations and Computer Graphics, Vol. 5, No. 4, October-December 1999

25. Robinson J. CRE method of element testing and Jacobian shape parameters. Engineering Computations. 1987, Vol. 4, June, p. 113-118

26. S. H. Lo. Generating quadrilateral elements on plane and over curved surfaces. Conputers & Structures, 1989, 31:421-426

27. S. Weyer, A. Frohlich, H. Riesch-Oppermann, L. Cizelj, M. Kovac. Automatic finite element meshing of planar Voronoi tessellations. Engineering Fracture Mechanics. 2002, 69:945-958