

This item was submitted to Loughborough's Institutional Repository (<https://dspace.lboro.ac.uk/>) by the author and is made available under the following Creative Commons Licence conditions.



**CC creative commons**  
COMMONS DEED

**Attribution-NonCommercial-NoDerivs 2.5**

**You are free:**

- to copy, distribute, display, and perform the work

**Under the following conditions:**

**BY:** **Attribution.** You must attribute the work in the manner specified by the author or licensor.

**Noncommercial.** You may not use this work for commercial purposes.

**No Derivative Works.** You may not alter, transform, or build upon this work.

- For any reuse or distribution, you must make clear to others the license terms of this work.
- Any of these conditions can be waived if you get permission from the copyright holder.

**Your fair use and other rights are in no way affected by the above.**

This is a human-readable summary of the [Legal Code \(the full license\)](#).

[Disclaimer](#) 

For the full text of this licence, please go to:  
<http://creativecommons.org/licenses/by-nc-nd/2.5/>

# The Investigation of a Method to Generate Conformal Lattice Structures for Additive Manufacturing

James Brennan-Craddock

*Doctoral thesis submitted in partial fulfilment of the requirements for the award of Doctor of Philosophy of Loughborough University*

November 2011



## Abstract

Additive manufacturing (AM) allows a geometric complexity in products not seen in conventional manufacturing. This geometric freedom facilitates the design and fabrication of conformal hierarchical structures: three-dimensional arrays of struts arranged into lattices that fit within a particular shape. Entire parts or regions of a part can be populated with lattice structure, designed to exhibit properties that differ from the solid material used in fabrication. Multiple components can be consolidated into a single part.

Current computer aided design (CAD) software used to design products is not suitable for the generation of lattice structure models. Although conceptually simple, the memory requirements to store a virtual CAD model of a lattice structure are prohibitively high. Conventional CAD software defines geometry through boundary representation (B-rep); shapes are described by the connectivity of faces, edges and vertices. While useful for representing accurate models of complex shape, the sheer quantity of individual surfaces required to represent each of the relatively simple individual struts that comprise a lattice structure ensure that memory limitations are soon reached. Small arrays of geometrically simple structures can be generated, however this severely restricts the design opportunities afforded by AM. Additionally, the conventional data flow from CAD to manufactured part is arduous, involving several conversions between file formats. As well as a lengthy process, each conversion risks the generation of geometric errors that must be fixed before manufacture.

A cut down version of B-rep modelling was implemented that, despite measures to simplify the process was still prohibitively slow. Before the potential of AM fabricated conformal structures could be thoroughly investigated a method was needed that could generate large arrays of complex conformal lattice structures. A literature review was conducted to investigate alternative methods of geometry representation that are more suitable towards representing lattice structure geometry.

A method was developed to specifically generate large arrays of lattice structures, based on a general voxel modelling method identified in the literature review. The method is much less sensitive to geometric complexity than conventional methods and thus facilitates the design of considerably more complex structures. The ability to grade structure designs across regions of a part (termed 'functional grading') was also investigated, as well as a method to retain connectivity between boundary struts of a conformal structure. In addition, the method

streamlines the data flow from design to manufacture: earlier steps of the data conversion process are bypassed entirely.

The effect of the modelling method on surface roughness of parts produced was investigated, as voxel models define boundaries with discrete, stepped blocks. It was concluded that the effect of this stepping on surface roughness was minimal. This thesis concludes with suggestions for further work to improve the efficiency, capability and usability of the conformal structure method developed in this work.

## Acknowledgements

I would like to sincerely thank my supervisors, Prof. Richard Hague and Prof. Ricky Wildman for their advice, guidance and motivation over the last few years.

I would also like to thank the EPSRC through the IMCRC for funding this work, and my fellow Scuta project researchers who offered critique, direction and collaboration.

To past and present members of the AMRG, I thank you for your support. In particular, I would like to thank Mark East (for always making space), Dr. Guy Bingham (for advice and assurance) and Dave Brackett (for the expertise and endless patience exhibited while I learned Matlab!).

Finally, my thanks goes to my friends and family. You have provided the encouragement, support and invaluable proof-reading skills that I could not have done without.

# Table of Contents

Abstract .....	i
Acknowledgements.....	iii
List of Figures .....	viii
<b>1 Introduction and Background Work .....</b>	<b>1</b>
1.1 Additive Manufacturing .....	1
1.2 Overview of Scuta .....	5
1.3 AM Lattice Structures .....	7
1.4 Foam as an Energy Absorbent Material.....	8
1.4.1 Mechanisms of Energy Absorption.....	9
1.5 Design of Energy Absorbent AM Lattice Structures.....	11
1.5.1 Compromises in the Design of Foam-Inspired AM Lattice Structures.....	12
1.5.2 The Kelvin Cell as a Mathematical Representation of Foam .....	13
1.5.3 Concept Designs for Energy Absorbent AM Lattice Structures .....	14
1.6 The Greater Potential of Lattice Structures.....	16
1.7 Conclusions from Background Work: Issues with Modelling Lattice Structures .....	17
<b>2 The Generation of 3D Geometry and the Implications for Lattice Structure Design - the Conventional Route.....</b>	<b>19</b>
2.1 Conventional route - Generation of Lattice Structures through Conventional CAD .....	21
2.1.1 Overview of Conventional CAD.....	21
2.1.2 Applications - Structure Generation using Conventional CAD.....	27
2.1.3 Suitability of Conventional CAD for Structure Design.....	30
2.2 Stage 2 of Conventional Route - Generating Lattice Structures at STL .....	32
2.2.1 The STL File Format.....	32
2.2.2 Applications - Processes that use STL .....	34
2.2.3 Advantages and Disadvantages of STL manipulation .....	38
2.3 Stage 3 of Conventional Route - Generating Structures at Slice Level.....	40
2.3.1 Slice formats.....	40
2.3.2 Applications - Processes that use Slice Files .....	42
2.3.3 Suitability .....	48
2.4 Summary of the Conventional Route.....	49

<b>3</b>	<b>The Generation of 3D Geometry and the Implications for Lattice Structure Design - Alternate Methods.....</b>	<b>50</b>
3.1	STL 2.0.....	51
3.2	Voxels.....	55
3.2.1	Voxel Methods for Structure Generation.....	56
3.2.2	Suitability of Voxels for Structure Generation.....	60
3.3	Function Representation.....	64
3.4	Conclusions.....	66
<b>4</b>	<b>The Design of Conformal Lattice Structures.....</b>	<b>68</b>
4.1	Overview of General Lattice Structure Design.....	68
4.2	Methods to Generate Conformal Lattice Structures.....	69
4.2.1	Trimmed Structures.....	69
4.2.2	Swept Structures.....	74
4.2.3	Meshed Structures.....	76
4.2.4	Voronoi Tessellations.....	79
4.3	Summary of Conformal Structure Methods.....	82
<b>5</b>	<b>Research Methodology.....</b>	<b>84</b>
5.1	Problem Identification.....	84
5.2	Research Aims.....	85
5.3	Research Approach.....	85
<b>6</b>	<b>Preliminary Work: Adapting Boundary Representation.....</b>	<b>86</b>
6.1	Skinning a Trimmed Structure.....	86
6.1.1	The Net Skin.....	87
6.2	B-rep Net Skin Construction Method.....	90
6.3	Basic Implementation of the Method.....	90
6.3.1	Defining the Conformal Shape.....	90
6.3.2	Defining the Base Tessellation.....	91
6.3.3	Connecting Intersection Points.....	94
6.3.4	Constructing Net Skin Geometry.....	95
6.3.5	Writing Formats Suitable for Manufacture.....	97
6.3.6	Summary of the Basic Implementation.....	99



6.4	Advanced Implementation of the Method .....	100
6.4.1	Representation of the Conformal Shape .....	100
6.4.2	New Unit Polyhedron.....	103
6.4.3	Reduction of Intersection Checking.....	106
6.5	Discussion.....	109
6.6	Summary .....	111
<b>7</b>	<b>The Development of a Conformal Structure Method .....</b>	<b>112</b>
7.1	The Conformal Structure Method.....	112
7.1.1	Importing a CAD Model for Processing.....	113
7.1.2	Trimming Structure to a Conformal Shape .....	116
7.1.3	Constructing the Net Skin .....	119
7.1.4	Combining Structure Elements .....	124
7.1.5	Conversion to Slice File .....	125
7.2	Functional Grading.....	126
7.3	Ensuring Net Skin Connectivity .....	128
7.4	Constructing a Skin from a Conformal Shape .....	132
7.5	Structure Visualisation.....	133
7.6	Strengths and Limitations of the Conformal Structure Method.....	135
7.6.1	Speed and Robustness.....	135
7.6.2	Initial Overhead.....	137
7.6.3	X-Y Plane .....	137
7.6.4	Pixel Stepping.....	138
7.7	Summary .....	138
<b>8</b>	<b>The Effect of Pixel Stepping on the Surface Roughness of Conformal Structure Method Produced Parts.....</b>	<b>140</b>
8.1	Experiment Aims.....	141
8.2	Experiment Method .....	141
8.2.1	Sample Design and Manufacture.....	141
8.2.2	Surface Roughness Measurement .....	143
8.3	Fourier Analysis.....	145
8.4	Methods to Reduce Pixel Stepping.....	158
8.5	Conclusions .....	159

<b>9</b>	<b>Conclusions and Further Work.....</b>	<b>160</b>
9.1	Achievement of Research Aims .....	160
9.2	Key Conclusions .....	160
9.3	Recommendations for Further Work.....	161
	<b>References .....</b>	<b>165</b>

## List of Figures

Figure 1-1: AM parts, clockwise from top left: 'Trabecular' tray [9], 'Lightpoem' personalised candle holder [10], Prosthetic leg [11], 'Dahlia' wall light [9], volume and flow optimised frontplate [12], hydraulic component [13], bionic handling assistant [14]..	2
Figure 1-2: Topology optimised bracket concept for use in a commercial aircraft [16].....	3
Figure 1-3: A 16-part duct assembly consolidated into a single part for AM [2] .....	3
Figure 1-4: A self-supporting powder bed versus support structures. The 'low angled' face on the right side of the part does not require support structures .....	4
Figure 1-5: A schematic of the laser sintering process .....	5
Figure 1-7: a) An equidistant mesh, b) a single AM textile link and c) a conformal AM textile [24] .	7
Figure 1-8: Examples of lattice structures: a) 'octet truss' and b) bitruncated cubic array of truncated octahedra .....	8
Figure 1-9: Formation of cellular structure; bubble expansion pushes material into walls and struts .....	9
Figure 1-10: the cellular structure of closed-cell [26] and open-cell [34] foam.....	9
Figure 1-11: Comparison of elastic behaviour of solid and foam made from same material, for a given peak stress (taken from [27]) .....	10
Figure 1-12: Compression of a hexagonal honeycomb. Although not a true representation of foam, it illustrates localised collapsing and how it propagates [25] .....	11
Figure 1-13: A Kelvin cell (with curved faces) and the planar-faced truncated octahedron it is based on .....	13
Figure 1-14: An FE model of a Kelvin cell structure [45] .....	14
Figure 1-15: Straight strut structure design with a) no node fillet and b) large node fillet.....	15
Figure 1-16: The number of triangles required to represent a) cylindrical and b) triangular struts is vast.....	15
Figure 1-17: The compressive response of examples of the straight strut samples <b>Error! Bookmark not defined.</b>	
Figure 1-18: Helical strut structure design.....	16
Figure 1-19: The compressive response of examples of the helical strut samples.. <b>Error! Bookmark not defined.</b>	
Figure 1-20: Straight and helical struts samples before, during and after maximum compression .....	<b>Error! Bookmark not defined.</b>
Figure 1-21: Shin protector concept, designed as a demonstration part somewhere between football shin guard and cricket shin pad .....	18
Figure 2-1: Conventional route of data flow from design to additive manufacture.....	20
Figure 2-2: Screenshots from NX of hole feature with CNC-influenced parameters.....	21
Figure 2-3: a) A cube and it's implicit functions, b) the ' $Y < Y_{max}$ ' half space and c) the half-spaces that define a cube from infinite space .....	22
Figure 2-4: Boolean operations between two primitives .....	23
Figure 2-5: CSG hierarchy tree .....	23
Figure 2-6: Planar, toroidal and quadratic B-rep surfaces .....	24
Figure 2-7: Union of two B-rep cubes - vertices are first modified which cascades down into edge and face modification .....	25
Figure 2-8: Simple CAD model with two parametric controls.....	26

Figure 2-9: A selection of cell types available in CASTS [80] .....	27
Figure 2-10: a) Input shape, b) tessellated structure & c) section of trimmed structure with hip joint model [56] .....	28
Figure 2-11: 3D printed tissue scaffolds [55] .....	28
Figure 2-12: a) Cell placement, b) cell designs with torus-shaped common interface visible & c) photograph of fabricated part [54] .....	29
Figure 2-13: Porous structure modelled in CSG [64].....	29
Figure 2-14: a & b) valid models & c) invalid model, but each with a Euler's characteristic of 2 (based on a figure from [72]).....	31
Figure 2-15: Example geometry as a) a CAD model & b) an STL model.....	32
Figure 2-16: The ASCII STL file format.....	33
Figure 2-17: The relationship between triangle count and size of a binary STL file .....	34
Figure 2-18: AutoFab structure generation software - showing lattice structure and solid skin option [92] .....	35
Figure 2-19: Conforming a structure to a shape through sweeping.....	35
Figure 2-20: Watertight connection between two unit cells [93] & inset: surface model of a unit cell.....	36
Figure 2-21: Examples of the complexity achievable with the Gibson et al. conformal structure method [70] .....	36
Figure 2-22: Efficient geometry design in STL modelling.....	37
Figure 2-23: TetraLattice structure [95].....	38
Figure 2-24: Manifold errors: a) mismatching connecting surfaces and b) intersecting triangles ..	39
Figure 2-25: Layers of a slice file corresponding to AM process' layer manufacture .....	40
Figure 2-26: Relationship between slice format and AM machine type.....	41
Figure 2-27: Trimming a structure to a shape at the slice level with netFabb Studio software [106] .....	42
Figure 2-28: Approximate trimmed structure visualisation in netFabb Studio [106].....	43
Figure 2-29: Pillar, diagonal and octahedral struts .....	44
Figure 2-30: Fine metal structures produced with Manipulator [52,108] .....	44
Figure 2-31: Side view of interpolated laser profiles of a diagonal strut in Manipulator software.	45
Figure 2-32: Constructing regular tessellations with structured placement of seeding points [96]	45
Figure 2-33: Generating multiple slices of structure by 'animating' seeding points [96] .....	46
Figure 2-34: SEM images of FDM bead lattice structures [109] .....	47
Figure 2-35: SEM images of spaced scan patterns [112] .....	47
Figure 3-1: Integrating into the conventional route from design to additive manufacturing .....	50
Figure 3-2: Redundant code in the ASCII STL file format (facet A and facet B are neighbours) .....	51
Figure 3-3: Defining multiple and graded materials in the AMF format [115] .....	52
Figure 3-4: Defining a structure in the AMF format [115] .....	53
Figure 3-5: AMF pseudo-code, based on [118].....	53
Figure 3-6: a) Planar face with 1 normal, b) curved face with three and c) subdivided further to more accurately represent a curved surface [118] .....	54
Figure 3-7: Low resolution pixel image of a circle and voxel model of a sphere .....	55
Figure 3-8: Tissue scaffold modelled with voxels [125] .....	56
Figure 3-9: Filling a slice of voxels with structure [125].....	57
Figure 3-10: Random porous structures generated from voxel models [127] .....	57

Figure 3-11: a) Voxel model - stepping in three axes. b) approximation of a laser sintered model - stepping in one axis. ....	58
Figure 3-12: a) A voxel model and b) the isosurface constructed from it .....	59
Figure 3-13: a) Eight voxel group, b) midpoint selection and c) midpoints connected to form triangular faces .....	59
Figure 3-14: a) Triangulation of a flat face of a B-rep model (2 triangles), b) triangulation through isosurface construction of a voxel representation of a flat face (200 triangles).....	60
Figure 3-15: 2D quadtree decomposition .....	61
Figure 3-16: 3D-Coat voxel modeller: capable of complex and organic geometry, including surface texture [134].....	62
Figure 3-17: The implicit functions and the Booleans to generate the cubic lattice structure .....	64
Figure 3-18: Modifying structure geometry through manipulation of a function [59] .....	65
Figure 3-19: F-rep structure dimensions grading as a function of a) distance along x-axis, b) distance from surface of sphere and c) random noise [59] .....	65
Figure 3-20: Integrating alternate methods into the conventional route. F-rep and voxel methods can generate geometry independently or import from CAD (hence dashed arrows), whereas STL 2.0 (as primarily a translation language) requires a CAD model .....	67
Figure 4-1: Lattice structure naming convention.....	69
Figure 4-2: Trimming a structure to fit a shape .....	70
Figure 4-3: Trimmed structures investigated for medical applications [78][79] .....	70
Figure 4-4: Trimmed structure generation by Wettergreen et al. [54] .....	71
Figure 4-5: Topology optimisation - selection of structure through a density map [81].....	72
Figure 4-6: Trimming a structure to a thin-walled conformal shape .....	72
Figure 4-7: A lattice structure partially covered by a skin.....	73
Figure 4-8: Sweeping a structure around a shape .....	74
Figure 4-9: Swept structure mapped to thin-walled geometries [49,70] .....	75
Figure 4-10: Sweeping a structure between two surfaces.....	75
Figure 4-11: Difficulty in sweeping a structure around a shape with tight bends and no clear 'top' and 'bottom' .....	76
Figure 4-12: A finite element mesh.....	77
Figure 4-13: Mesh and structure [81] .....	78
Figure 4-14: a) tet, b) penta and c) hex finite elements .....	79
Figure 4-15: Construction of a 2D Voronoi diagram (based on a figure from [153]).....	79
Figure 4-16: Constructing a random structure from a 3D Voronoi cell .....	80
Figure 4-17: Applications of 3D Voronoi tessellations: a) in modelling of granular flow [159] and b) general modelling [151] .....	81
Figure 4-18: Constructing a regular Voronoi tessellation .....	81
Figure 4-19: Conformal structure methods: trimmed, swept, meshed and Voronoi.....	82
Figure 4-20: a) alternating cubic and b) bitruncated cubic tessellations.....	83
Figure 6-1: A trimmed structure with a) no skin and b) a solid skin .....	87
Figure 6-2: a) a trimmed structure, b) a trimmed structure with net skin and c) just the net skin .....	88
Figure 6-3: Constructing a net skin for a single cell by considering a structure as a based on a tessellation of polyhedra .....	88
Figure 6-4: Types of net skin .....	89
Figure 6-5: 'Nearest neighbour' failure to properly re-connect cut struts on a thin-walled part....	89

Figure 6-6: Representation of point cloud data .....	91
Figure 6-7: Constructing a surface from the point cloud .....	91
Figure 6-8: Structure of unit polyhedron matrix .....	92
Figure 6-9: A polyhedron intersecting with a surface and interpolated points generated on its edges.....	93
Figure 6-10: Instances of a face intersecting a surface .....	93
Figure 6-11: Finalised intersection points and correctly joining those points .....	94
Figure 6-12: The necessity of spheres when mapping geometry to struts.....	95
Figure 6-13: Generating the points to be enclosed by a convex hull.....	96
Figure 6-14: 2D convex hull example .....	96
Figure 6-15: Net skin struts generated from convex hulls .....	97
Figure 6-16: Varying level of faceting of convex hull struts .....	97
Figure 6-17: STL ASCII format [2].....	98
Figure 6-18: The completion the basic implementation .....	99
Figure 6-19: Mesh grid stretching on steeper curvature .....	101
Figure 6-20: a) FE mesh generated from CAD model of a section of a body armour concept b) Net skin struts constructed from intersection with FE mesh nodes (every 10th node shown) .....	102
Figure 6-21: Triangulation of a shape: a) standard STL conversion and b) FE mesh .....	103
Figure 6-22: Complete and partial unit polyhedra - duplication apparent when tessellated .....	103
Figure 6-23: Partial polyhedron definition matrices .....	104
Figure 6-24: Using the partial polyhedron definition matrices to determine which intersection points connect .....	105
Figure 6-25: Constructing base tessellation to fit a conformal shape .....	106
Figure 6-26: Identification of selection windows for individual polyhedra .....	107
Figure 6-27: Number of times fewer checks required by advanced method compared to basic method .....	108
Figure 6-28: Irregularities in pattern of a surface mesh .....	110
Figure 6-29: An intersection tolerance that is suitable for a region of fine mesh may not be for a coarser region.....	110
Figure 7-1: Straight strut (with triangular cross section) and helical strut structures.....	113
Figure 7-2: Data flow .....	114
Figure 7-3: The conventional flow of data from modelling to manufacture .....	115
Figure 7-4: A single layer of a vector and raster slice file .....	116
Figure 7-5: Basic code structure of the conformal structure method .....	116
Figure 7-6: Bitmap to matrix conversion of a single slice .....	117
Figure 7-7: Trimming structure to a conformal shape - analogous to a Boolean intersection.....	118
Figure 7-8: Generating trimmed structure slice - analogous to a Boolean intersection .....	119
Figure 7-9: Comparison of cell and 'hole'.....	120
Figure 7-10: Difference between conformal shape and skin .....	120
Figure 7-11: Net skin matrix calculation .....	121
Figure 7-12: Net skin generation by Boolean subtraction .....	122
Figure 7-13: Method of generating a 'hole cell' from a given structure .....	123
Figure 7-14: Partially formed net skin where hole cell tessellation does not completely intersect solid skin .....	123

Figure 7-15: Modified hole cell that completely intersects skin in dashed region, although the same modifications on other instances of cell destroys neighbouring struts (compare with Figure 7-14).....	124
Figure 7-16: Combining structural elements.....	124
Figure 7-17: Tracing boundary of pixellated shape.....	125
Figure 7-18: Structure of a Common Layer Interface (CLI) file .....	126
Figure 7-19: Functionally grading a conformal shape .....	127
Figure 7-20: Functionally graded structures generated from input gradients .....	128
Figure 7-21: Misalignment between a complex strut and the net skin .....	129
Figure 7-22: A graded boundary .....	129
Figure 7-23: Convolution calculation for a single element of a conformal shape matrix.....	130
Figure 7-24: Trimming the blurred conformal shape matrix to the original conformal shape.....	131
Figure 7-25: Re-aligning a complex strut with the net skin .....	132
Figure 7-26: Generating a hollow skin from a conformal shape.....	133
Figure 7-27: Structure visualisation with isosurfaces .....	134
Figure 7-28: Effect of sampling resolution on isosurface quality.....	135
Figure 7-29: Shapes with different volumes in the same 3D envelope .....	136
Figure 7-30: Orientation lock of sliced parts.....	138
Figure 7-31: A concept chest protector for taekwondo designed by the author on the Scuta project.....	139
Figure 8-1: Pixel stepping at a boundary.....	140
Figure 8-2: Data flow for conventionally generated and method generated samples.....	142
Figure 8-3: Repetition of pixel stepping.....	143
Figure 8-4: Excessive reflection off a rough surface with optical measurement.....	144
Figure 8-5: Talysurf CLI 2000 test bed.....	144
Figure 8-6: Profile and Fourier transform of 10° sample .....	145
Figure 8-7: Low frequency aliasing.....	146
Figure 8-8: Transformation from a sawtooth profile to closer representation of actual pixel-stepped profile.....	147
Figure 8-9: Transformation from a sawtooth profile to a closer representation of an actual pixel-stepped profile.....	148
Figure 8-10: Calculating expected periodicity of the pixel stepping on a 10° sample .....	149
Figure 8-11: Uniform and non-uniform pixel stepping when approximating a straight, angled line .....	149
Figure 8-12: Identified pixel stepping patterns.....	150
Figure 8-13: Differences between pixel profiles and traced profiles.....	151
Figure 8-14: Edge effects - the source of low frequency peaks .....	152
Figure 8-15: Comparing original signal with Hann windowed signal for 10° method sample.....	154
Figure 8-16: Comparing windowed samples of all 10° samples - conventionally and method generated.....	155
Figure 8-17: Hann windowed signal for 5° method samples .....	155
Figure 8-18: Hann windowed signal for 5° conventional samples.....	156
Figure 8-19: Fourier spectra for all samples generated through conformal method.....	157
Figure 8-20: Hann windowed signal for 25° and 40° conventional samples.....	158

Figure 9-1: A B-rep cell constructor where cell type, cell diameter, strut diameter and helical diameter can be adjusted..... 163

Figure 9-2: Integrating a density map from a topological optimisation approach into the conformal structure method..... 164



# 1 Introduction and Background Work

The geometric freedom afforded by additive manufacturing (AM) allows the fabrication of complex lattice structures. These lattice structures can be designed to possess a variety of physical properties that have potential applications in a number of fields. However, conventional computer-aided design (CAD) tools are unsuitable for designing lattice structures due to the underlying method in which they handle geometry. The work in this thesis addresses this assertion with the investigation of a modelling method specifically for the generation of complex lattice structures, however this chapter first summarises the background work conducted by the author that led to it. The chapter is split into sections that detail each stage of this background work as well as an overview of the literature used as direction.

## 1.1 Additive Manufacturing

The technologies behind the emerging field of AM stem from those developed under the term 'Rapid Prototyping' (RP) [1]. As the name suggests, the application for RP technologies is to quickly create a prototype part, while the aim of AM is to provide the freedom to design end use parts exactly as require [2] (2) (2) (2) (1. Gibson, David W. Rosen, & Stucker, 2009) d [1,2]. Sometimes referred to as rapid manufacturing' (RM), 'solid freeform fabrication' (SFF), 'layer manufacturing technologies' (LMT) and others, the term 'additive manufacturing' has nonetheless been adopted by an ASTM International standard [2-6].

There are many existing or potential advantages of additive manufacturing, from the reduction of product development time and costs, reduction of resources required and the removal of tooling associated with conventional manufacturing [1,2]. Arguably the most significant of these is the removal of the tooling, as this has two major advantages. The first is that tooling (e.g., moulds or machine tools) is expensive to manufacture, which prohibits product variation [1]. The second is that the tooling itself obstructs a part during manufacture, limiting the geometric complexity of conventionally manufactured products [7].

Parts produced by injection moulding, for example, require draft angles on faces parallel to the direction the mould closes, to ensure easy removal after moulding [8]. Some features of a design may not be possible with a standard two-part mould and may require a more expensive multi-part mould. Certain features (like internal geometries) are impossible in injection moulding [1,8]. Essentially, production cost is linked to part complexity. This is not the case with additive manufacture [1]. A selection of products designed for AM are shown in Figure 1-1.

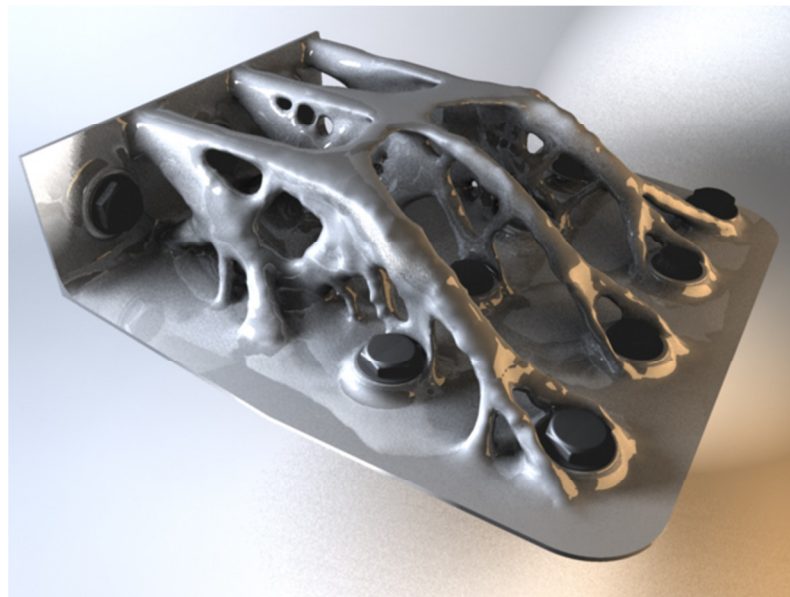


Figure 1-1: AM parts, clockwise from top left: 'Trabecular' tray [9], 'Lightpoem' personalised candle holder [10], Prosthetic leg [11], 'Dahlia' wall light [9], volume and flow optimised frontplate [12], hydraulic component [13], bionic handling assistant [14]

Rather than forming parts in a subtractive (e.g., machined) or formative (e.g., moulded) manner, additive manufacturing 'prints' parts in layers. AM covers a broad range of manufacturing processes, which all share this layer by layer approach to manufacturing. This allows the fabrication of geometrically complex products that cannot be made by any other method [1], and reduces the link between cost and complexity. The lack of tooling allows the design of customised and personalised products, rather than the conventional 'one size fits all' [15].

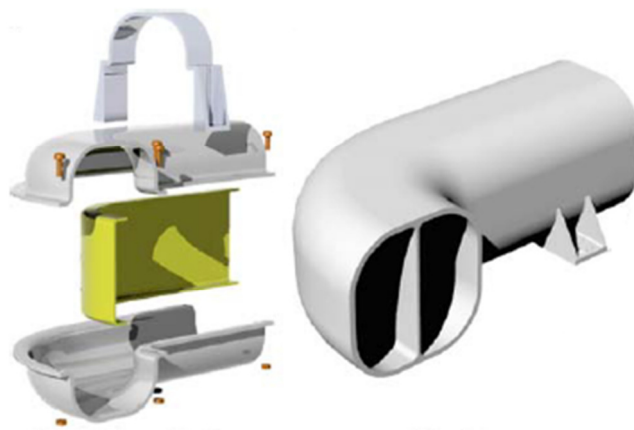
The freedom to design a part unconstrained by conventional manufacturing processes allows its function to become the main consideration of the design process. It has been proposed that design optimisation techniques commonly used in the construction industry could be used extensively in the design of products when fabricated by AM processes [1]. Figure 1-2 shows an example aircraft bracket that has been designed using OptiStruct, a topology optimisation software package. Boundary conditions and loads are applied to a 3D design envelope and the topology optimisation software determines the optimum shape to withstand the environment. If performance of a product

is not the most important consideration, AM allows products to be optimised in terms of weight, aesthetics or environmental impact.



*Figure 1-2: Topology optimised bracket concept for use in a commercial aircraft [16]*

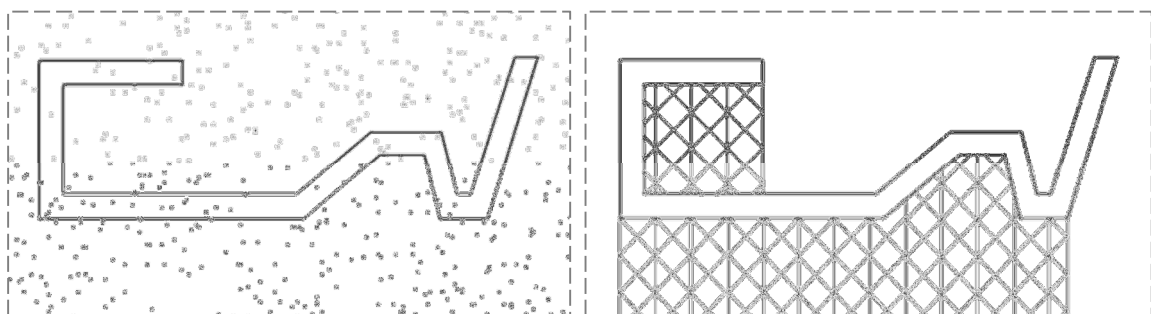
Another advantage of AM technologies is that of part consolidation [1]. Manufacturing constraints in conventional manufacturing mean that a product may have to be made in several parts, to allow for their removal from tooling. As AM processes have no physical tooling within the build area, an assembly can be reduced down to a minimum number of parts, where once separate components can be merged together. This will reduce assembly time as well as eliminating any potential weaknesses between mated surfaces [2]. An example of part consolidation applied to the design of an aircraft duct is shown in Figure 1-3.



*Figure 1-3: A 16-part duct assembly consolidated into a single part for AM [2]*

Arguably the most significant disadvantage currently associated with AM is the available material selection. The narrow range of materials that can be processed have relatively poor material properties compared to conventional alternatives [4]. As a relatively new set of technologies, AM is expensive to run, and the materials expensive to buy [17]. AM is also much slower than conventional processes for the mass production of parts and accuracy is also an issue [4,18]. However, the combination of customisation, geometric complexity and part consolidation present a new way to design products, a 'new design paradigm' [17].

There are a wide range of technologies that fall under the term 'additive manufacturing', each fabricating parts in a layer by layer fashion. 3D printers run a print head over a powder bed that deposits a coloured binder that glues successive layers together [19]. The powder bed is lowered by the thickness of one layer, a new layer of powder is rolled over and the process repeats. Jetting is a similar technique that uses a print head to deposit droplets of thermosetting polymer onto a build platform, that is subsequently cured with a UV lamp [1,20]. A 'support structure' must also be built at the same time for overhanging portions of the part to build onto, although 'low angled' faces may be self-supporting, as shown in Figure 1-4. This support structure is removed in post processing. Fused deposition modelling (FDM) machines work in a similar fashion to the jetting process. Rather than a multitude of print heads depositing droplets of thermosetting polymer, FDM extrudes a single fine bead of nearly molten thermoplastic (such as ABS) [21]. Each layer is 'drawn' by the FDM nozzle, which also requires a support structure to be built concurrently.



*Figure 1-4: A self-supporting powder bed versus support structures. The 'low angled' face on the right side of the part does not require support structures*

Stereolithography (SLA) uses a laser to scan over the surface of a vat of resin, the laser spot curing the resin into a solid polymer [1]. Unlike the other processes, the part is submerged in a liquid resin vat during manufacture, however like jetting and FDM a support structure is still required to prevent the part drifting away. Laser sintering (LS) utilises a similar setup to SLA, but rather than a vat of resin, the laser scans over a powder bed, sintering (or melting, depending on the process) powder together [21]. Laser sintering can process a range of materials, from polymers (such as nylon-11 and

nylon-12), to ceramics and metals (such as stainless steel and titanium). Like 3D printing, the powder bed supports the fabricated part so an extra support structure is not required. Metal laser sintering is an exception, as often a support structure is required to prevent the sintered part from warping during building [22]. A schematic of the laser sintering process is shown in **Error! Reference source not found.**

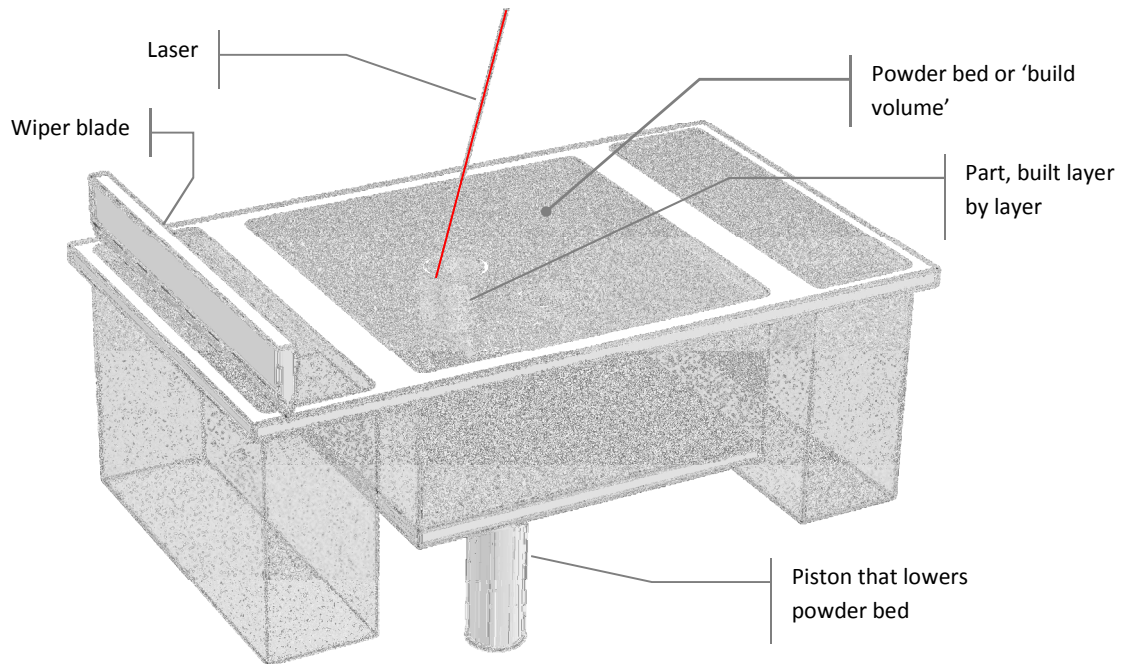


Figure 1-5: A schematic of the laser sintering process

## 1.2 Overview of Scuta

The work in this thesis was undertaken as part of an EPSRC-funded, multi-disciplinary research project called Scuta. The aim of Scuta (Latin for 'shields') was to investigate the use of AM in the production of protective garments for sports. The geometric freedom afforded by AM potentially allows the design of PPE that is tailored to the unique shape of an individual's body. The impetus being that by more closely fitting an individual athlete than generic PPE can, a conformal garment could potentially both improve performance and reduce the risk of injury [23]. Three sports were selected for investigation: football, cricket and taekwondo. To provide focus to the research, a single piece of PPE from each sport was nominated: a shin guard, a shin pad and a hogu (a wrap-around chest protector) respectively.

Scuta was a multi-disciplinary project involving several departments and research groups across Loughborough University. Sport and Exercise Science focused on the simulation of human-inflicted impacts and the associated reactions/injuries, i.e. the working environment of sports PPE. Electrical and Electronic Engineering developed wearable wireless impact sensors to facilitate automatic scoring (specifically for taekwondo). The Sports Technology Institute (STI) tested existing conventional PPE: both the mechanisms behind energy transfer and absorption as well as users' perceptions of comfort and fit. Finally, the Additive Manufacturing Research Group (AMRG) investigated the implementation of AM technologies in sports PPE.

The research carried out by the AMRG was itself split into two broad fields: the investigation of more flexible, elastomeric materials that could be processed by laser sintering and the investigation of a means to design conformal, energy absorbing garments. The work in this thesis was instigated to address the latter research question, officially titled 'Work Package 4 (WP4): Generation of 3D Conformal Data', its position within the project structure detailed in Table 1-1.

WP1: Human Related Impact Intensity during Contact Sports	(Sport and Exercise Science)
WP2: Perception of Comfort of Protective Equipment	(STI)
WP3: Advanced Modelling of Protective Equipment of Sports	(STI)
<b>WP4: Generation of 3D Conformal Data</b>	<b>(AMRG, focus of this thesis)</b>
WP5: Materials Analysis	(AMRG)
WP6: Instrumentation and Validation	(Electrical and Electronic Eng.)

*Table 1-1: Scuta project structure*

Laser sintering was determined to be the AM process with the most potential for sports personal protective equipment (PPE). Compared to other AM processes, laser sintered polymer parts are relatively strong and suitable for functional applications [21]. Significantly, polymer laser sintering does not require additional support structures to be built as parts are self-supporting within the powder bed [19,21]. This is advantageous because complex parts can be built without having to consider strategies to minimise or remove supports. This is particularly advantageous for designs with complex internal geometries, where support removal may be laborious or impossible.

To achieve the goal of conformal energy absorbent AM garments, WP4 was further split into two areas of research: the design and development of energy absorbing AM samples and the investigation of a method to conform these samples to fit a particular shape. Initially, AM textiles were investigated as a means to generate energy absorbent garments, such as the example shown in Figure 1-6 [24]. However, the capability of energy absorbent textiles were limited, specifically in

terms of the capability to conform to a given surface. It is not possible to map textiles links across particularly tight curves, due to the size of the links.

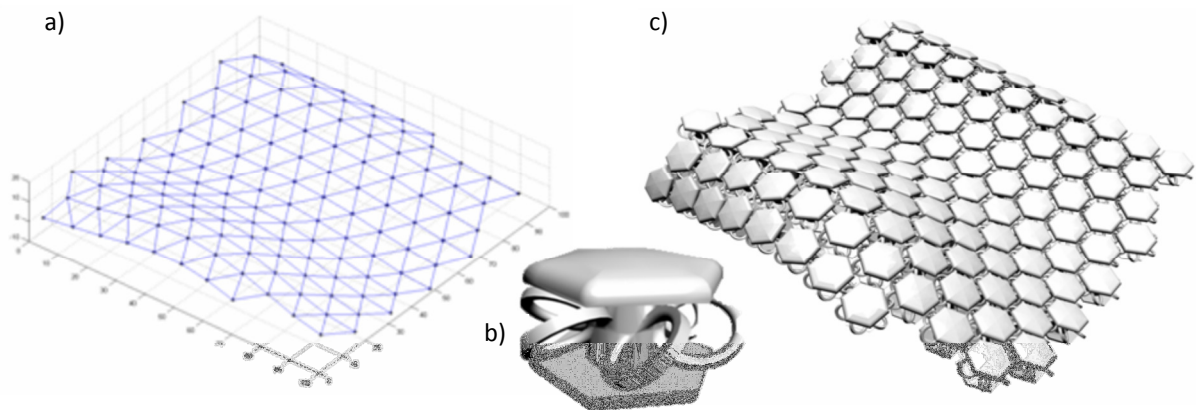


Figure 1-6: a) An equidistant mesh, b) a single AM textile link and c) a conformal AM textile [24]

In terms of the mechanical properties of AM textiles, because AM textiles are generally composed of discrete links, loosely linked together to facilitate flexibility, they are not suitable for energy absorption. An AM textile cannot efficiently transfer energy from a localised impact over a large area, due to this loose assembly. For this reason, all the energy transferred from a localised impact would have to be absorbed by the few links in the local area. In terms of energy absorbing conformal PPE, AM textiles lack potential both as energy absorbers and conformal capability.

### 1.3 AM Lattice Structures

The three items of sports PPE selected for focus on the Scuta project (football shin guard, cricket shin pad and taekwondo hogu) all perform in a broadly similar role. The energy from a projectile (either ball or body contact) impacts the PPE at a single point. Enough of this energy must be absorbed by the PPE to prevent injury. While AM textiles were deemed unsuitable for further development, lattice structures were identified as a potential basis for energy absorbing sports PPE. Like AM textiles, a lattice structure is composed of a repeating element that are tessellated over a region. Rather than a loose array of interlocking links however, a lattice structure is a solid arrangement of interconnected struts, examples shown in Figure 1-7. As a single body, movement, force or energy are more easily transferred from one region to another. This theoretically allows the energy of an impact to spread over a wider region of an AM sports PPE garment, reducing the strain in any one area and making the PPE concept more resilient.

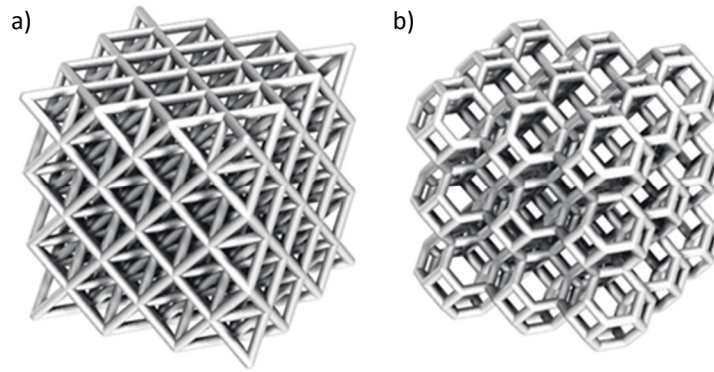


Figure 1-7: Examples of lattice structures: a) 'octet truss' and b) bitruncated cubic array of truncated octahedra

Preliminary work for this thesis investigated compliant lattice structures that could be used for energy absorption. In the design of energy absorbent lattice structures, inspiration was found while investigating the mechanisms behind the compression of foam.

#### 1.4 Foam as an Energy Absorbent Material

The most commonly used material in energy absorbing applications [25], polymeric foams are also used in the majority of existing sports PPE. Polymeric foams are also used in general PPE, footwear, bumpers, seating, mattresses and packaging [26-28]. Foams are suited to absorbing energy because they have a low density, cellular structure that deforms readily under load [25].

Polymeric foam products are often moulded; a viscous mixture of the liquid polymer, blowing agent and water (as well as a surfactant and catalyst) is poured into a mould [29]. The blowing agent and water react, producing gas which is absorbed into the liquid mixture. Upon the point of super saturation within the mixture, nucleation of bubbles occurs [30]. As the bubbles expand, the polymer between bubbles is stretched to a membrane, pushing excess polymer into struts between these bubbles. These struts form irregular polyhedral-like cells, as illustrated in Figure 1-8. The manufacturing process is a source of anisotropy in foams. For example, in moulding the foam expands in a mould much like bread rising, and cells elongate in the rise direction [31,32].



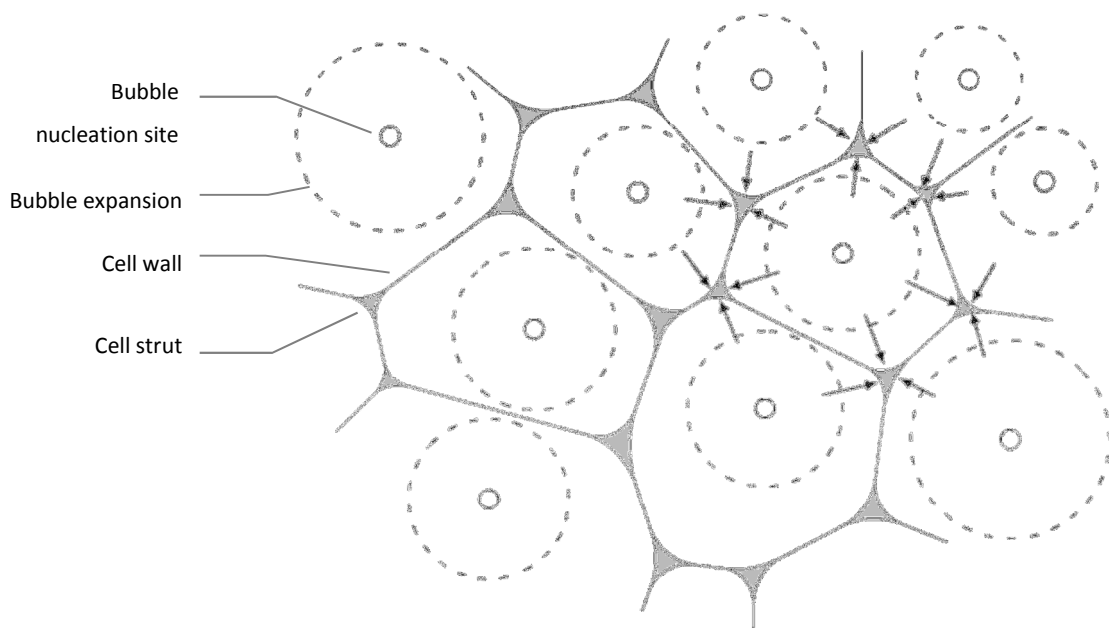


Figure 1-8: Formation of cellular structure; bubble expansion pushes material into walls and struts

There are two broad categories of foam: open cell and closed cell. A closed-cell structure consists of walls and struts, forming isolated cells of gas. An open-cell structure consists solely of struts – during manufacture, the cell walls formed between nucleating bubbles are pierced and shrink into the forming struts. This creates a structure where a fluid (ie: air) can flow freely through it [25,33].

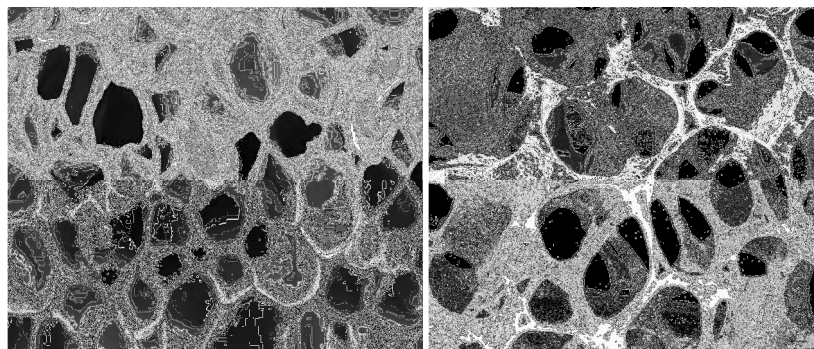


Figure 1-9: the cellular structure of closed-cell [26] and open-cell [34] foam

#### 1.4.1 Mechanisms of Energy Absorption

Foams are particularly good at absorbing energy, due to their low density cellular structure that deforms elastically. Energy absorption is the ability to compress when kinetic energy is exerted on the object without rebounding [28]. A foam absorbs the energy from an impact, reducing the transfer of energy to the object it is protecting to specified levels. The mechanisms of energy absorption vary depending on the type of foam. An open cell foam absorbs energy through strut

buckling and the work done in pushing air out of the complex structure. Closed cell foams absorb energy through strut buckling, cell wall bending and the compression of the gas trapped within each cell [27].

The cellular structure also lends itself well to absorbing oblique impacts [25]. Compared to a solid block of the same material, foams will always generate a lower peak force when absorbing the same amount of energy [27]. This is demonstrated by the difference in area under the curves in Figure 1-10.

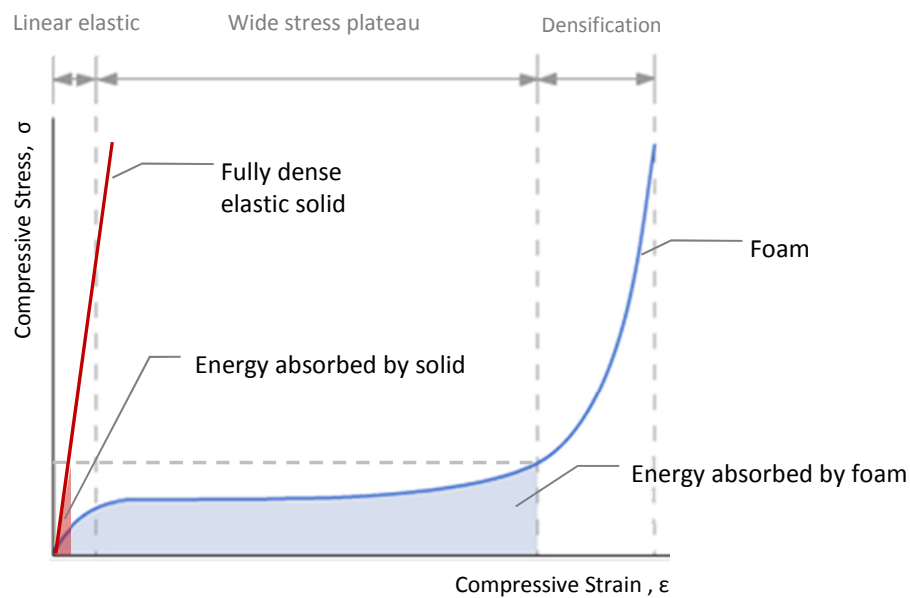


Figure 1-10: Comparison of elastic behaviour of solid and foam made from same material, for a given peak stress (taken from [27])

Foams in general undergo compressive behaviour of three regimes [35,36]. Initially, the foam exhibits a linear elastic region where very little energy is absorbed. A wide plateau region follows, with a densification phase where stress rises steeply [36]. Low relative density foams only have two regimes of compression: a longer linear elastic region, followed by densification [35].

For high relative density foams, the initial linear elastic region is short, it only occurs at small deformations. The foam structure compresses relatively uniformly across the area of impact [35]. In open-cell foams, energy is absorbed through the bending of the structure's struts. Closed-cell foams will generally absorb higher impacts as compression of the gas trapped inside cells will contribute to its energy absorption mechanisms [37]. However, it has been shown that closed-cell foams will suffer from fatigue as air permeates through its cell walls and escapes. Over repeated compressions this lowers the foam's resilience (the ability to recover to the original shape) [37].

The wide 'stress plateau' illustrated in Figure 1-10 is a characteristic of foams; the reason why they make such good energy absorbing materials. It is a stage of elastic, localised collapse where weaknesses in the structure buckle [32,35]. Further deformations will spread from these localised areas until the foam is completely compressed. This is illustrated in Figure 1-11.

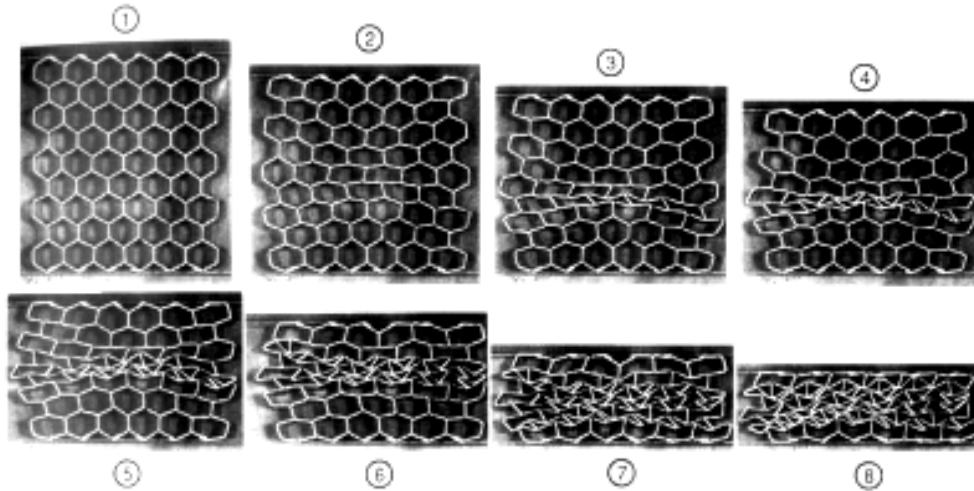


Figure 1-11: Compression of a hexagonal honeycomb. Although not a true representation of foam, it illustrates localised collapsing and how it propagates [25]

The final regime of compression sees significant stiffening of the foam in a phase known as densification [25,35]. The point at which this occurs is at the foam's peak stress and is shown by the rapid gradient increase at about 70% strain in Figure 1-10 [27,28]. Cells have completely collapsed and the struts of the structure are compressed into contact. The structure now more closely approximates the properties of a solid block, hence the similarity in gradients between the 'fully dense elastic solid' and the densification phase of the foam Figure 1-10. Plastic deformation will start to occur sporadically throughout the structure, reducing the foam's resilience [32]. This will occur primarily at the joints between struts, forming 'plastic hinges' [25].

## 1.5 Design of Energy Absorbent AM Lattice Structures

An energy absorbing material would have a stress plateau just below that which was identified as the damaging level of the object it is protecting [25]. In effect, the foam would not reach the densification phase during normal operating conditions. This would ensure that the product does not suffer fatigue, but it would also ensure that minimal impact energy is transferred to the object that it is protecting.

With an understanding of the effects of the cellular structure on the compressive behaviour of foam, concept energy absorbing AM lattice structures were developed. With the control that AM brings in structure design, AM lattices have the potential to exhibit the exact properties and fit the exact shape required, as opposed to conventional foams bound by the foaming process and tooling constraints. Despite the freedom that AM brings however, there are key differences between the structure of a perfect foam and what can be fabricated through AM.

### 1.5.1 Compromises in the Design of Foam-Inspired AM Lattice Structures

The difference in scale between conventional foams and AM lattice structures means that replicating the random cellular structure would not be conducive to purposefully designed lattice structures. While foams can consist of individual cells less than a millimetre in diameter, a key constraint of current AM fabrication is resolution. Laser sintering, for example, is limited to a minimum feature size of 0.4mm [38]. In terms of lattice structure design, this means that practically the smallest feasible strut diameter imposed is 0.4mm. With this as a consideration, cell diameter must be roughly an order a magnitude higher than strut diameter, making minimum cell diameters in the order of 5 to 10mm.

Another constraint of AM is the removal of support material after fabrication. Whether the process requires the building of an actual support structure (such as stereolithography or jetting) or merely the removal of unsintered powder (like laser sintering), the lattice structure design must facilitate this. For a structure inspired by foams, only an open cell foam could be feasibly replicated by AM.

These two constraints remove a number of the energy absorption mechanisms attributed to conventional foams from the potential of AM structures based on foam. Namely, cell wall bending and trapped air cannot contribute as closed cell foams cannot be manufactured. The support material could not be removed from a closed cell lattice structure. Also, the scale possible in AM is an order of magnitude higher than the micro-scale of conventional foam cellular structure. Air flow will not be significantly impeded by an AM structure at the macro scale like a micro-scale foam, so will not contribute to energy absorption to any useful level. This leaves cell strut bending as the only remaining mechanism for energy absorption in an AM lattice structure.

Just as varying cell size, relative density and anisotropy of the cellular structure of foams affects compressive behaviour, it can be assumed that modifying the geometry of a lattice structure will have an effect on its mechanical properties. In essence, cells of different size and shape will exhibit different properties. This is true for the random cellular structure of conventional foams, however

the scale of a foam's cellular structure is an order of magnitude below the scale of any impacting objects. When a cellular structure is composed of many thousands of cells, the properties of each cell are averaged out [39]. In their analysis of the effect of the sample size on shear properties of a foam, Rakow and Waas found that when a sample was reduced to a size that consisted of only tens of cells, the mechanical behaviour of individual cells was magnified [39].

Because AM lattice structures consist of much larger cells than conventional foams, a randomised structure will effectively ensure that the properties of the lattice structure will also have a degree of randomness. AM lattice structures have the potential to exhibit purposeful and exactly designed properties, but an underlying random cellular structure would 'overwrite' this. It was therefore required that a regular foam structure was investigated as the base of an energy absorbing lattice.

### 1.5.2 The Kelvin Cell as a Mathematical Representation of Foam

During the manufacture of foams, the foaming process generates a three-dimensional cellular structure that is controlled by the principle of minimal surface energy [25,40]. In (1887), Sir William Thomson (later Lord Kelvin), investigating the minimum surface qualities of aqueous foams, developed a repeatable unit cell for a perfectly ordered, regular foam. This later became known as the 'Kelvin cell' [41]. The Kelvin cell is a modified truncated octahedron (a polyhedron composed of six square and eight hexagonal faces): the hexagonal faces have zero mean curvature, while the square faces are flat with outwardly curved edges [41], as illustrated in Figure 1-12.

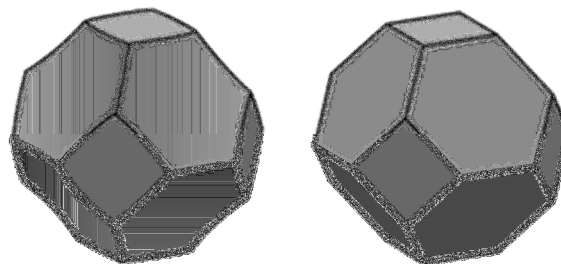


Figure 1-12: A Kelvin cell (with curved faces) and the planar-faced truncated octahedron it is based on

The Kelvin cell has since been used as the basis for finite element analyses of foam structures to better understand the mechanical properties of foam [42-44]. Closed- and open-cell foams can be modelled by taking the Kelvin cell as either a surface model of edges and faces or just by modelling the edges. For more accurate FE models, The Kelvin cell can be augmented by assigning geometry to edges to more closely represent the struts of a foam structure (as shown in Figure 1-13) [44]. Finite element analysis of the deformation mode of a Kelvin cell by Gong *et al.* was found to be similar to

that from empirical foam testing, exhibiting cell strut buckling - characteristic of the wide stress plateau found in foam compression [25,43].

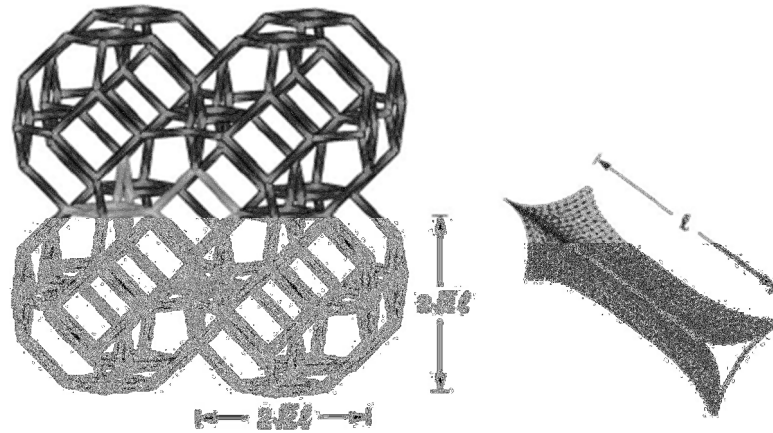


Figure 1-13: An FE model of a Kelvin cell structure [45]

Unlike random foams, the Kelvin cell has an orthotropic elastic response: it is isotropic in the three orthographic axes [46]. The Young's modulus of the Kelvin cell has been predicted to vary about 10% with a change in direction from these orthographic axes [47]. As a regular representation of the cellular structure of foam, the Kelvin cell makes a promising candidate for the basis of energy absorbent AM lattice structures.

### 1.5.3 Concept Designs for Energy Absorbent AM Lattice Structures

Concepts for energy absorbent AM lattice structures were designed, based on the Kelvin cell model of foams. A simple design is shown in Figure 1-14 with straight struts. The designs were modelled in the CAD software NX, which allowed the integration of parametric expressions to control key dimensions. As an extension of the foam-inspired background, the dimensions used to define these AM lattice structures is based in foam terminology. An individual repeating unit is termed a 'cell', the width of which is measured as 'cell diameter'. The thickness of individual struts is termed 'strut diameter'. An optional feature of this design is the 'node fillet', which eliminates sharp corners that may act as the starting point for cracks and more closely approximates the smooth and organic shape of foams.

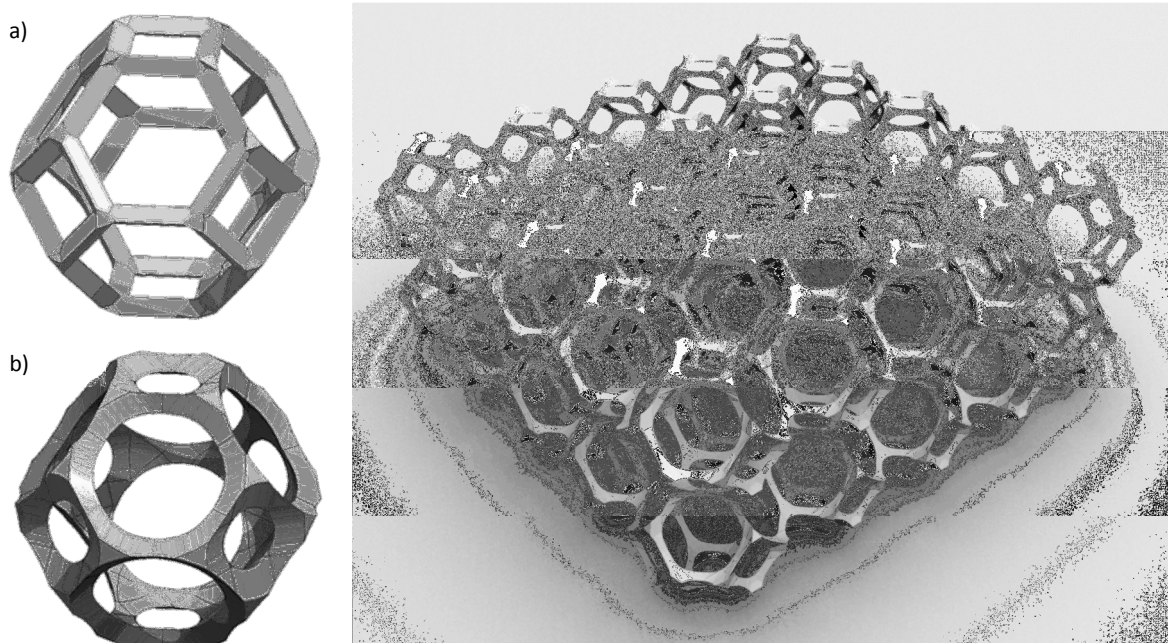


Figure 1-14: Straight strut structure design with a) no node fillet and b) large node fillet

The CAD model is converted into the STL format before fabrication by an AM machine, a process discussed in detail in Chapters 2 and 3. The STL format approximates curved geometry with flat, triangular polygons. To minimise the number of triangles required to represent each cell (and thus reduce file size and memory requirements) the struts were given triangular cross-sections. With no node fillet, each strut is fully represented by only six triangular polygons, as opposed to the dozens required to accurately represent a cylindrical strut, as shown in Figure 1-15. The manner in which struts join to form the Kelvin cell facilitates smooth connections between triangular struts. Four struts meet at each node of the structure, so for any strut in that group of four, its triangular cross section is in line with the other three struts, also shown in Figure 1-15.

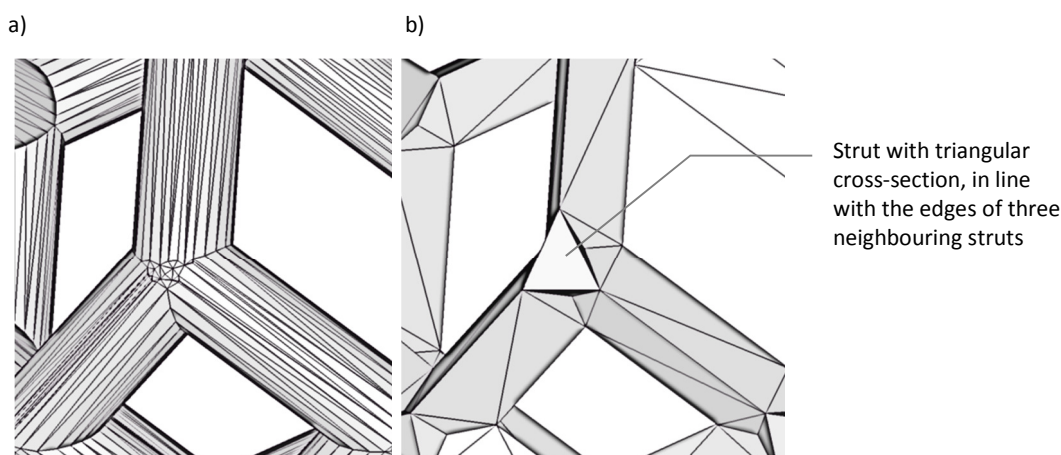


Figure 1-15: The number of triangles required to represent a) cylindrical and b) triangular struts is vast

When compressed, the samples exhibited the characteristic three stage compression seen in foams [48]. Taking advantage of the geometric freedom that AM provides, a more complex structure design was also tested. The Kelvin cell was augmented with helical struts, as shown in Figure 1-16. The helix is constrained to a 'law curve' that reduces the radius of the helix to zero at each end. This allows helical struts to connect as a structure without intersecting each other. The reasoning behind this helical design was that by increasing the overall length of each strut would allow more deformation within the structure (both for struts in tension and compression) and thus increase energy absorption.

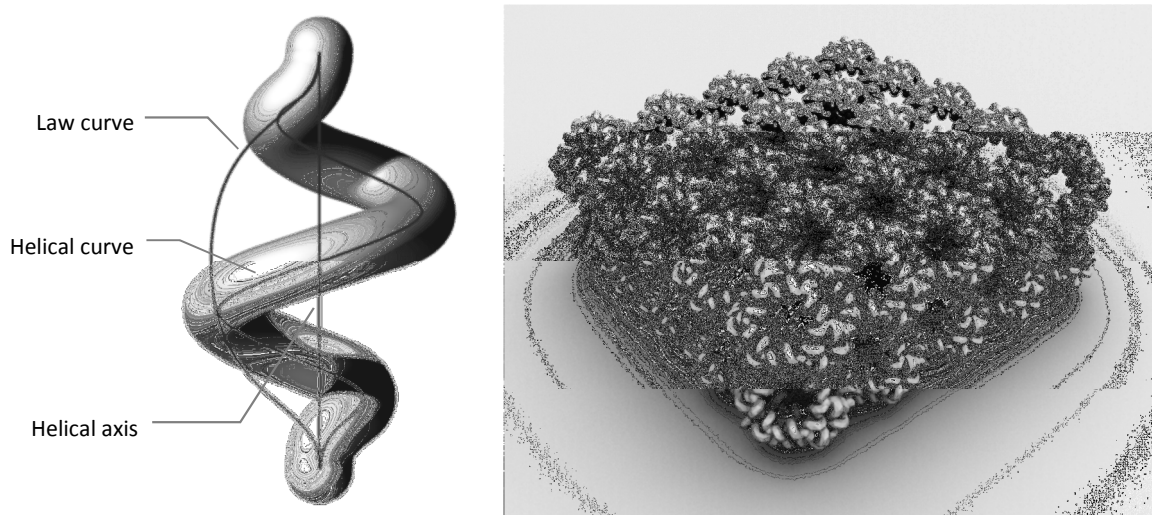


Figure 1-16: Helical strut structure design

This augmented design was also subjected to compression testing, the modifications giving the samples a two stage compressive profile quite unlike the response seen in standard foams [48]. Further to these preliminary tests, a whole series of compression tests were carried out to quantify how cell diameter, strut diameter and node fillet affected the compressive behaviour of the straight strut design. However, none of the samples were at a level that could be used as sports PPE. A combination of both limited material options for laser sintering and the absence of many of the mechanisms of energy absorption available in foams (as discussed in Section 1.5.1) meant that the energy absorption capabilities of the structures were limited.

## 1.6 The Greater Potential of Lattice Structures

Although energy absorption for sports PPE was shown to be out of the reach of current AM lattice structures, there are other applications that they are ideally suited to. Lattice structures provide a way to modify the properties of a part, or regions of it, with a wide range of potential applications. Lattice structures with high stiffness have applications in the aeronautical, automotive and space



industries, or any industry where weight reduction is a primary objective in component design [49,50]. Compliant structures could provide flexibility to regions of a component, as well as acting as energy absorption or impact protection [51,52].

There are even medical applications for lattice structures, such as the development of scaffolds designed to promote *in vivo* tissue growth [53-57]. The complexity achievable with AM is particularly apparent here as the organic form of the human body can be replicated. Dampening of noise or vibrations is another potential application, as well as heat exchangers [51,58]. Lattice structures can also have purely aesthetic applications. These can all be achieved in a single, consolidated part when designed for AM. Components with regions of structure that have different properties can be made without the need for assembly. While some research has gone into the construction of lattice structures through complex extrusions [58] or wire weaving [59], these are basic in comparison to what can be achieved through AM.

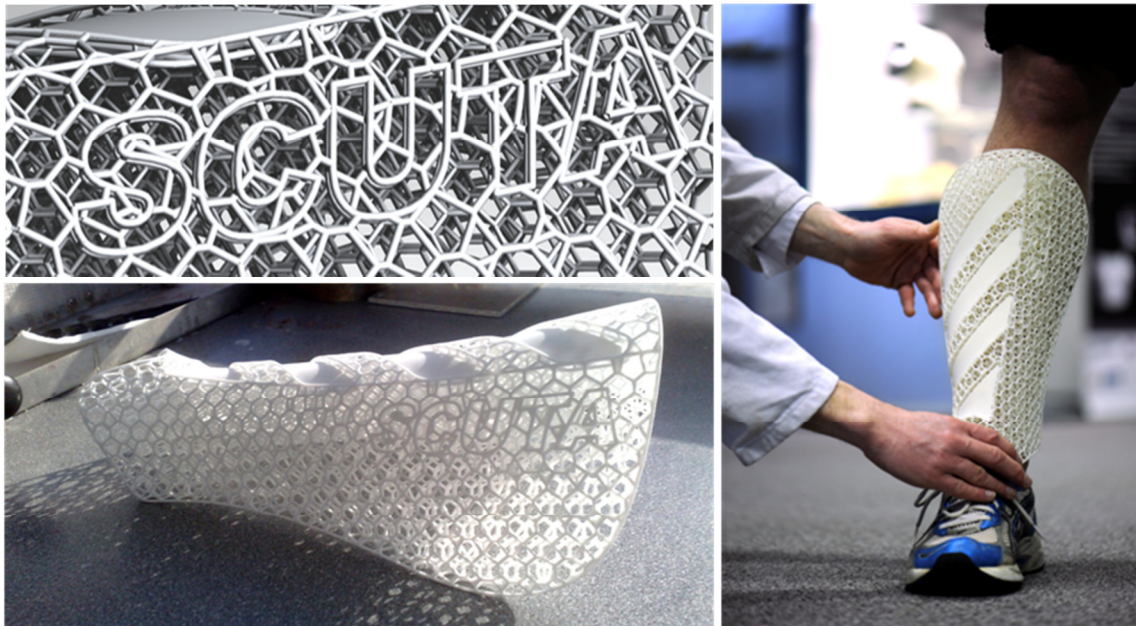
## 1.7 Conclusions from Background Work: Issues with Modelling Lattice Structures

The test samples, being simple 3D arrays of a unit cell, were fairly straightforward to model in conventional CAD software (NX, Siemens). A single cell was parametrically modelled and tessellated a number of times in x, y and z directions. However, even modelling the relatively small samples of the more geometrically complex lattice structures (e.g., the helical struts), the CAD software reached memory limits. For lattices of a few hundred struts, the CAD software became unstable and frequently crashed due to exceeding available computer memory. A workaround was discovered by modelling a single cell in CAD and converting this to an STL format, which could be tessellated a greater number of times before reaching limits. As discussed extensively in the following literature review, this is due to differences in the geometry representation between CAD formats and the STL format (in the latter models are faceted; surfaces are merely approximated with triangular polygons). However, this was not a suitable solution and memory limits were still reached manipulating the simplified STL geometry.

This issue was exacerbated when lattice structures were manipulated into a useful shape, i.e. for generating a conformal structure. The constraints that conventional CAD imposes on duplicating geometry are even more apparent when attempting to modify it. Figure 1-17 shows a generic 'shin protector' designed as a concept for the Scuta project. In conventional CAD the sole task of

generating geometry took three hours on five separate PCs. It was not possible to generate in one go without reaching memory limits.

It became clear through the course of this early work that the benefits of AM lattice structures cannot be fully realised until the tools exist to generate them. The ability to conform them to any required shape is also of paramount importance, as there is no use for a lattice structure with ideal properties that cannot be shaped to fit a working environment.



*Figure 1-17: Shin protector concept, designed as a demonstration part somewhere between football shin guard and cricket shin pad*

The following chapters investigate why the underlying geometry representation methods used in the creation of these lattice structures are not suitable, as well as the alternatives. The literature review also highlights currently available structure generation software that overcome these issues to some degree, but shows that there is still a gap: a requirement for a structure generation method that can efficiently generate structure geometry.

## 2 | The Generation of 3D Geometry and the Implications for Lattice Structure Design - the Conventional Route

There are two aims of this and the following chapter: to review the existing methods of generating three-dimensional, virtual geometry and to assess the suitability of these methods in the design of conformal lattice structures for additive manufacturing. To fully exploit additive manufacturing for structure design, the underlying method must be capable of both efficiently representing large, complex structures as well as facilitating the ability to manipulate these to fit a shape. The term 'complexity' can be a vague, general word with many meanings. Gibson, Rosen and Stucker have suggested four categories of complexity that additive manufacturing affords [2]:

- *Shape* complexity - it is possible to manufacture a product of virtually any shape, such as organic, freeform shapes with complex internal geometries that would prove too costly to machine or impossible to remove from moulds.
- *Hierarchical* complexity - the ability to manufacture lattice structures from micro- to macrostructure that have particular mechanical properties.
- *Material* complexity - because material is processed one layer at a time, the possibility exists that material properties can be graded across and between layers.
- *Functional* complexity - the ability to manufacture functional mechanisms that are built fully assembled.

For the purposes of this work, the ability to construct geometry with hierarchical complexity is important, and any references to complexity will refer to this particular category, unless otherwise stated.

Commercial computer-aided design (CAD) systems are typically used in the design of products that are to be manufactured by conventional processes, and do not fully exploit the design freedoms of additive manufacture [60,61]. Despite this, CAD software is still the primary environment to design AM products in due to the lack of readily available, commercialised alternatives.

There is a conventional route for designing a product for additive manufacture with CAD software, shown in Figure 2-1. A wide range of CAD software is commercially available, each utilising their own

file formats to store 3D data. Similarly, there are a wide number of AM processes that - in general - require data input in their own proprietary formats. Because this many-to-many arrangement would require each AM process vendor to provide conversion algorithms for every CAD system available (as well as continuous updates as needed), a single neutral format is used as an interface between the two. This is the STL format, named for the AM process that it was originally developed for: stereolithography [62]. An STL model is a faceted representation: the surfaces that comprise a CAD model are decomposed into a series of triangular polygons [63].

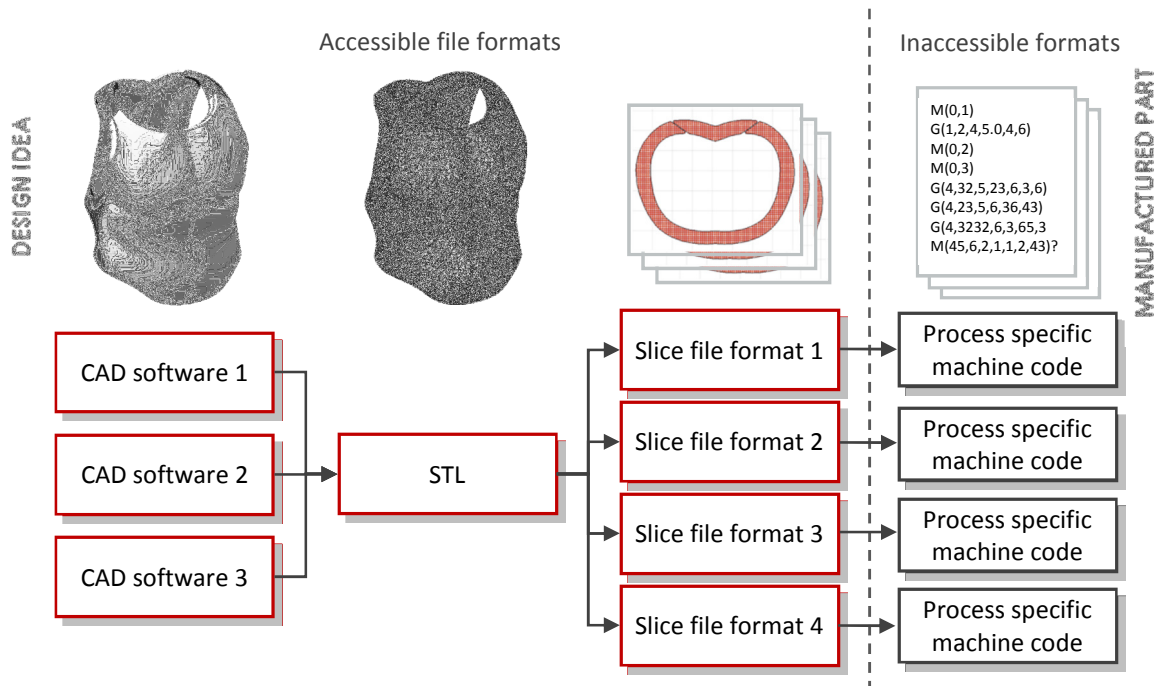


Figure 2-1: Conventional route of data flow from design to additive manufacture

A slicing algorithm then converts the STL model into a slice file: a series of cross sections that correspond to the layer-based fabrication of an additive manufacturing process [62]. This slicing is usually done by software specific to a particular additive manufacturing machine. The slice file is input into the machine, which then converts it to a machine code-like format that controls the mechanics directly. This last conversion step is completed automatically, the results of which cannot usually be observed by the user.

A number of groups have attempted lattice structure design and generation at each of the stages of this conventional route, while others have investigated implementing entirely different methods of geometry creation, detailed in the next chapter. Due to the confidentiality around data flow within most AM machines, the methods that avoid the conventional route for structure generation must

inevitably re-join it at some stage to format the data in a way that the machines can read and manufacture.

This chapter discusses the issues surrounding the manipulation of geometry at each of the stages of the conventional route. Alternative methods are discussed in the following chapter. This chapter is split into sections that describe and assess each of these methods: the method of geometry representation is first described, followed by any applications in structure generation and finally the method's suitability.

## 2.1 Conventional route - Generation of Lattice Structures through Conventional CAD

### 2.1.1 Overview of Conventional CAD

'Conventional CAD' is the first step in the conventional route from design to additive manufacture. Modern commercial CAD packages - such as NX or Pro ENGINEER - are powerful tools for the design, analysis and process planning of many conventionally manufactured products. This is because these feature-based modelling methods have been tailored to represent geometry in a way that is analogous to conventional manufacturing processes, describing geometric features of a part with functional meaning [64]. For example, 'draft' and 'hole' are two common features available to the user in CAD packages - the former a feature that relates directly to a design requirement for injection-moulded parts and the latter a common machining process, as shown in Figure 2-2.

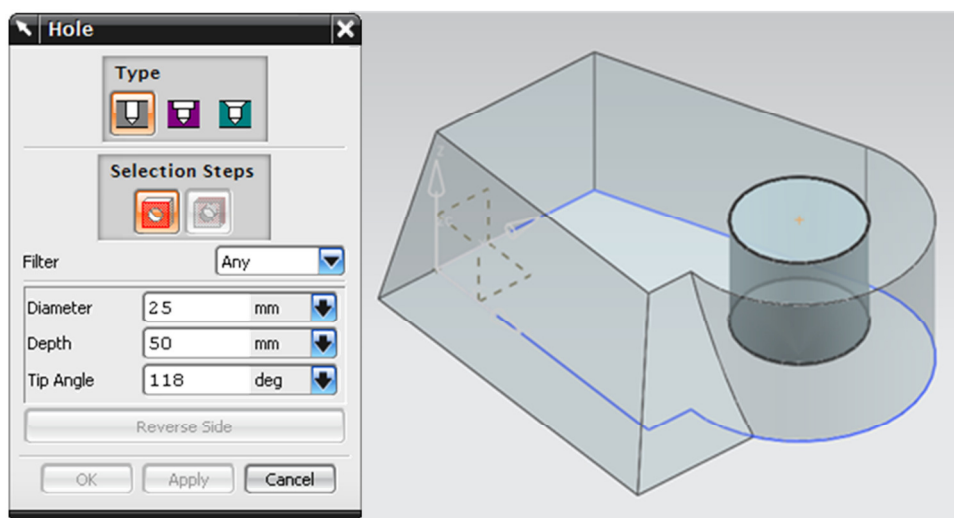


Figure 2-2: Screenshots from NX of hole feature with CNC-influenced parameters

Due to this design methodology, conventional CAD is better suited to designing traditional products than to fully realising the design freedoms of additive manufacture. The same can be claimed for the underlying methods that have been used to represent geometry in a CAD package. These methods are called 'constructive solid geometry' (CSG) and 'boundary representation' (B-rep) [2,60]. CSG is historically easier to implement than B-rep, although more limited in the geometry it can represent and thus is not common as the basis for modern CAD software.

### Constructive Solid Geometry

CSG models are based on the use of pre-defined 'primitives' that when combined by Boolean operations can form highly complex models [65,66]. Examples of primitives include cuboids, cylinders and spheres, which are implicitly defined functions in the form of  $f(x,y,z)=0$  [65,67]. For example, the implicit functions that define a cube consists of six inequalities, as shown in Figure 2-3a. Each of these inequalities corresponds to a 'half-space' - an infinite plane that splits 3D space into two. One side of the half-space is considered 'inside', the other 'outside', as shown in Figure 2-3b [67,68]. Six half spaces when arranged as shown in Figure 2-3c form a fully enclosed cube of finite dimensions.

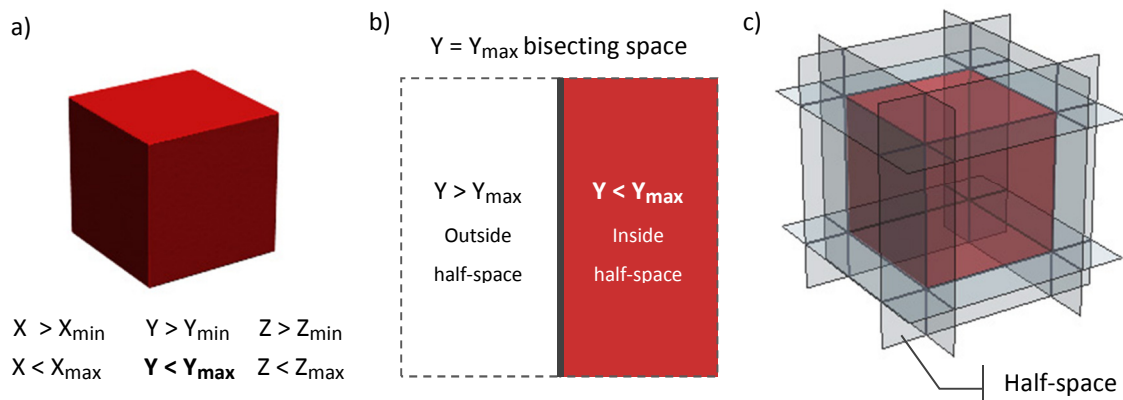


Figure 2-3: a) A cube and its implicit functions, b) the ' $Y < Y_{max}$ ' half space and c) the half-spaces that define a cube from infinite space

Because the half-space definition of solids makes it straightforward to determine what lies inside and outside of a particular primitive, CSG models are easily combined to form more complex shapes with Boolean operations. CSG utilises three Boolean operations: union, subtraction and intersection, also illustrated in Figure 2-4. A union operation combines two shapes into a single solid, whereas an intersection operation retains the space that both solids inhabit. A subtraction (also known as 'difference') removes a 'tool' solid from the 'target' solid, and is unique in that an order must be specified when selecting primitives as there are two possible solutions (i.e. A-B and B-A).

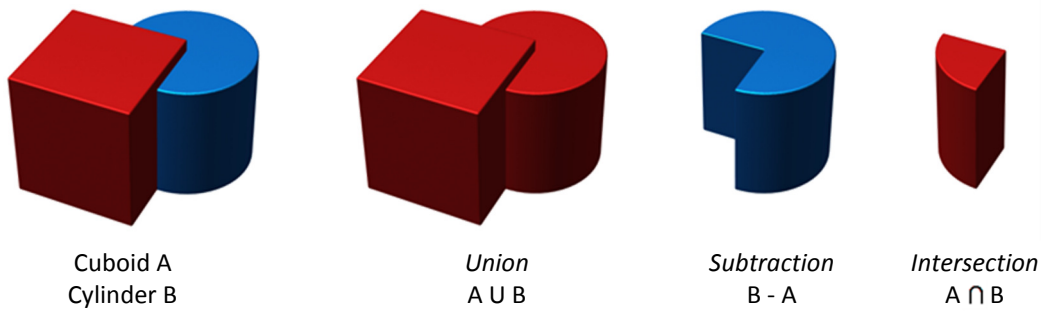


Figure 2-4: Boolean operations between two primitives

CSG models are described in a hierarchical tree, with each component primitive stored as a 'leaf'. Boolean operations, as well as general transformations such as translations and rotations, are stored as intermediate nodes in the branches of the tree [65,68]. This concept is illustrated in Figure 2-5. With an ordered combination of transformations and Boolean operations, complex composite models can be constructed. CSG trees are binary: each node can only have two branches [69]. This means that only two primitives can be combined per operation.

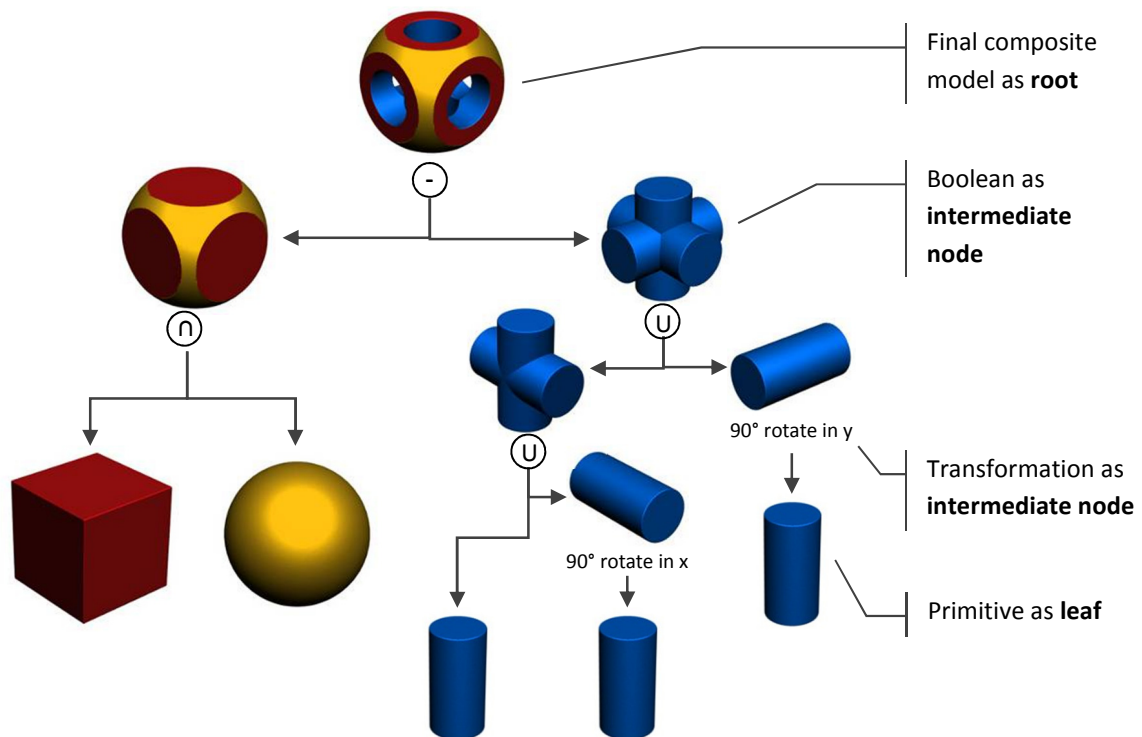


Figure 2-5: CSG hierarchy tree

## Boundary Representation

The complexity of CSG models are ultimately limited by the set of base primitives. Sweeping, freeform shapes such as the aerodynamic surfaces of an aircraft or the bodywork of a sports car cannot be achieved with CSG. Boundary representation indirectly represents a part through explicit definition of its boundary surfaces [70] and is capable of defining freeform complexity. B-rep definition of a model is split into two categories: topological information and geometric information [69].

Topological information describes the connectivity between faces and edges of a model whereas geometric information is the mathematical equations of these elements [69]. These equations can be linear or higher-order polynomial (e.g: quadratic), which determines what classification of boundary representation the model is. The STL file format is technically classified as faceted boundary representation: the triangular polygons that compose an STL model are derived from linear equations. Modern CAD systems utilise advanced boundary representation which in addition to planar surfaces also include quadratic, toroidal and spline surfaces such as NURBS (non-uniform rational B-splines) [70]. Examples of these surface types are shown in Figure 2-6.

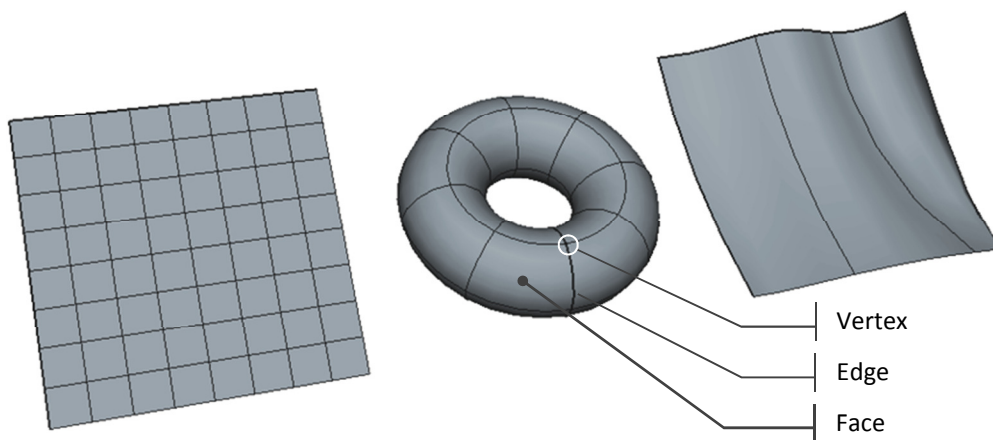


Figure 2-6: Planar, toroidal and quadratic B-rep surfaces

These surfaces have zero thickness - to define a solid shape, a set of surfaces are further characterised with topological information that define their connectivity. An edge is defined by vertices: when a set of edges form a closed loop, a face is defined. When a set of faces form a closed volume a solid body has been defined. [67,71]. Although intuitive, in practice this means that even simple shapes have lengthy and complex B-rep descriptions [72,73].

A B-rep model structure is essentially a set of lists - a vertex list (containing the vertex co-ordinates), an edge list (denoting vertex pairs that form edges), and a face list (that groups edges) [74]. The



order in which this topological information is listed determines what side of the boundary is classed as inside or outside of the body. Usually the vertices that compose a face are ordered counter-clockwise - a normal vector for each face is calculated using the 'right-hand rule' [69].

Unlike CSG, Boolean operations are not integral to boundary representation and as such are much more difficult to perform [74]. Consider the union of two cubes, as shown in Figure 2-7. In CSG, this is achieved by manipulating the simple set of implicit functions shown in Figure 2-3. The united cubes are represented by a function that is a combination of the two whole inputs. In B-rep however, depending on the positioning of the two cubes, some data must be deleted while new data is generated when the two B-rep cubes are combined into one structure. In the example in Figure 2-7, Vertices of one cube that are determined to be inside the other must be deleted (and vice versa), while new vertices are generated at intersections. Intersecting edges must be shortened by reassigning them to the newly generated vertices. Likewise, any intersecting faces must be replaced by referencing the newly modified edges. Euler's characteristic can be used to check the result is valid (e.g. to ensure that no new vertices have been missed) [73,74].

$$no. vertices - no. edges + no. faces = 2$$

Equation 2-1: Euler's characteristic

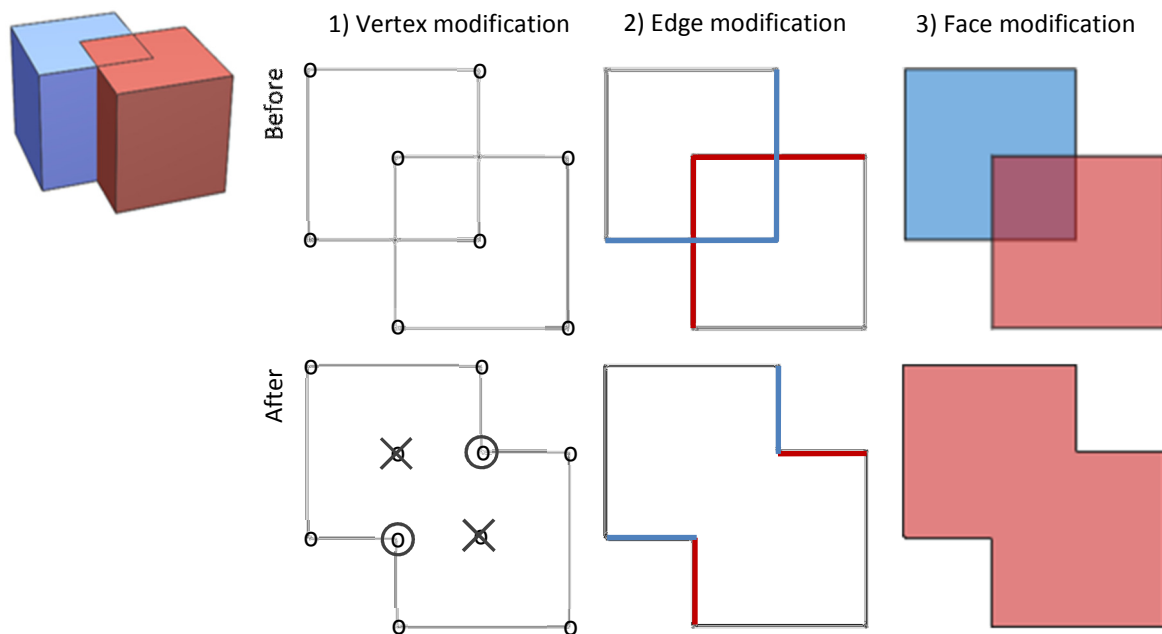


Figure 2-7: Union of two B-rep cubes - vertices are first modified which cascades down into edge and face modification

A B-rep CAD systems allow geometry - once created - to be controlled with variables or equations, a function termed 'parametric modelling' [60]. The ability to define relationships between dimensions

gives the user greater control over a model [75], an example shown in Figure 2-8. By changing the variable assigned as the rib thickness in this example, the model can be modified as required to suit a particular situation. The cylinder and hole diameters are related to each other: the equation controlling the diameter of the cylinder references that of the hole, which itself is controlled by another variable.

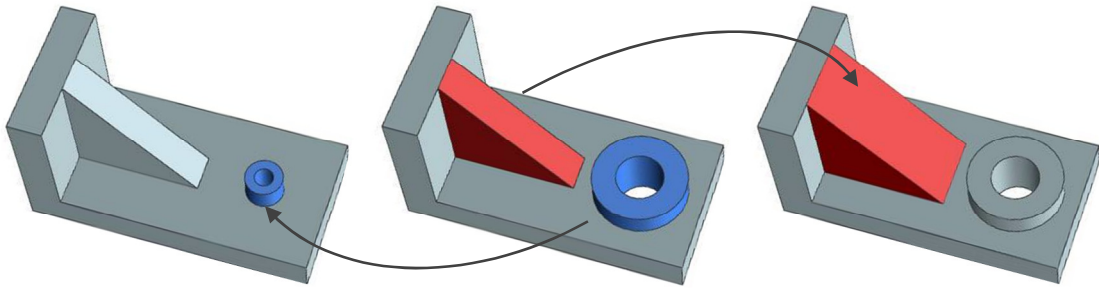


Figure 2-8: Simple CAD model with two parametric controls

### Hybrid Modelling

B-rep modelling is much more flexible than CSG. However, CSG still retains some advantages, such as the integration (and thus more efficient use) of Boolean operations. The two methods are complimentary in the sense that they each address the disadvantages of the other. For this reason, some work went into developing a hybrid modelling approach that exploits the advantages of both B-rep and CSG [76]. These CAD systems combined the hierarchy tree of CSG with the modelling ability of both methods. B-rep modelling was implemented on 'low level' geometric entities, such as individual faces of a shape. CSG integration allowed for robust Boolean operations between 'high level' primitives, i.e. the shape as a whole [76,77].

To utilise both CSG and B-rep in the modelling of the same design, a hybrid modeller must be able to convert between both methods. It is fairly straightforward to convert from CSG to B-rep - this is common practice in CSG modellers to generate a model that can be rendered for visualisation [72]. However, because the majority of freeform, sculpted shapes possible by B-rep are impossible in CSG, no all-encompassing B-rep to CSG conversions exist [72]. Hybrid modellers couldn't overcome the limitations of CSG modelling and are no longer a focus of research.

## Summary

Most modern CAD systems utilise boundary representation as a base because it is much more flexible than CSG. However, CSG still influences modern CAD systems in the implementation of model history trees [78]. Like CSG hierarchy trees, a model history tree describes a model by the individual features that compose it. This gives context to a model and allows for relationships between features. Edges can be placed in parallel, circles can be placed concentrically, *et cetera*.

The next section details how various groups have utilised the inherent advantages of conventional CAD software to design and construct lattice structures.

### 2.1.2 Applications - Structure Generation using Conventional CAD

Due to their relative ease of use, several groups have attempted to design and generate lattice structures with commercial CAD systems [54-57,79,80]. Tissue engineering has been a common application in much of this research, for example the generation of rigid, porous scaffolds to promote bone growth in prosthetic joints [57,81]. The majority of these methods trim a structure to a shape, discussed in Chapter 4. Naing *et al.* developed a system dubbed the 'computer aided system for tissue scaffolds' (CASTS) which links together several commercial systems to generate tissue scaffolds from MRI data [57]. The actual geometry construction module of the CASTS system runs Pro/ENGINEER, utilising B-rep modelling [57,60] A cell type is first selected from a built-in library within CASTS, examples shown in Figure 2-9. The parametric aspect of the software is exploited; a unit cell is scaled to desired dimensions with cell length, breath and height parameters [57].

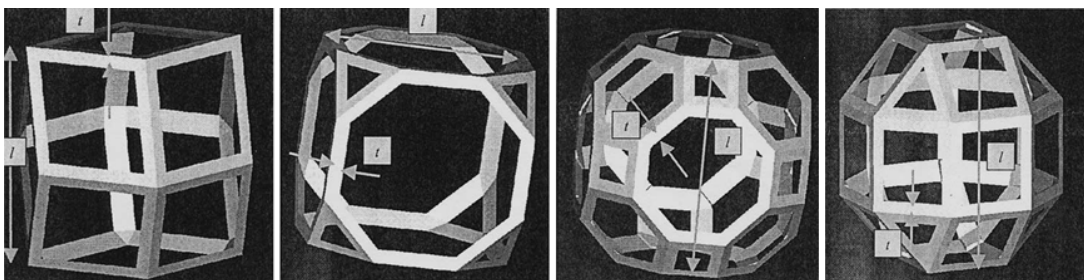


Figure 2-9: A selection of cell types available in CASTS [81]

To generate a scaffold that is conformal to the required shape (for example, a bone segment), a regular, cubic array is trimmed to fit it, an example shown in Figure 2-10. The bone segment that is to be populated with structure is imported into the Pro/ENGINEER environment (a) and the cell design

is orthogonally arrayed to fully enclose it (b). The segment is then Boolean intersected with the scaffold to generate a trimmed structure, conformal to the bone.

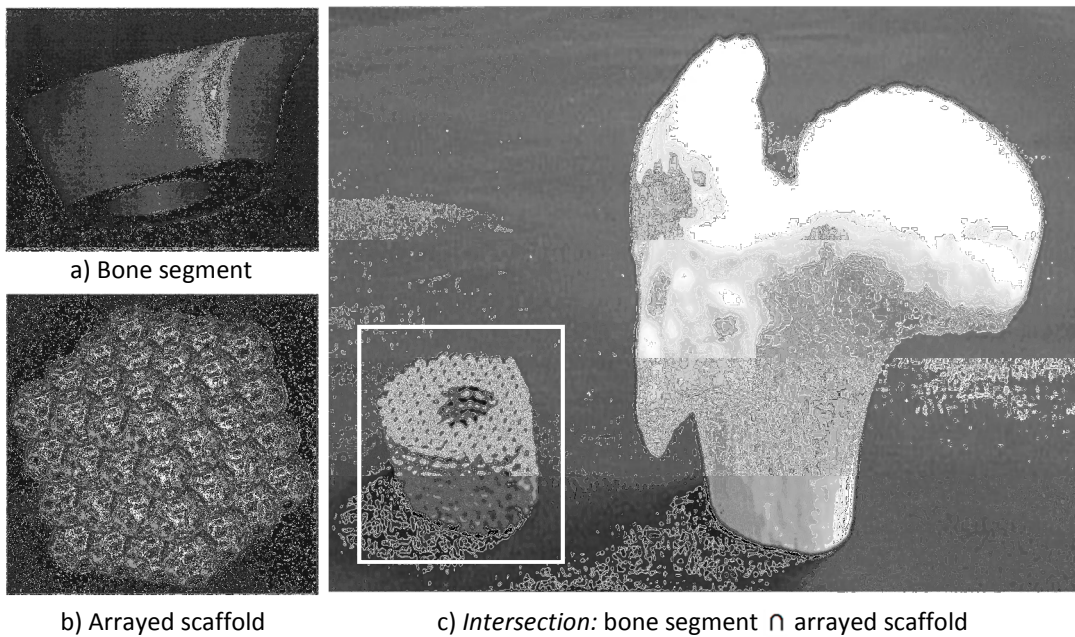


Figure 2-10: a) Input shape, b) tessellated structure & c) section of trimmed structure with hip joint model [57]

This trimmed structure is converted from a CAD model into an STL, through Pro/ENGINEER's built-in conversion tool. The data conversion process follows the conventional route described in Section **Error! Reference source not found.**; the STL is converted to a slice file and input into an additive manufacturing process (laser sintering in this case) for fabrication. A similar approach was taken by Lam *et al.* to conformal structure generation [56]. Tissue engineering scaffolds were generated and trimmed in the CAD package Unigraphics (now called NX), which then needed conversion to STL and then slice file format for manufacture on a 3D printer. Examples are shown in Figure 2-11.

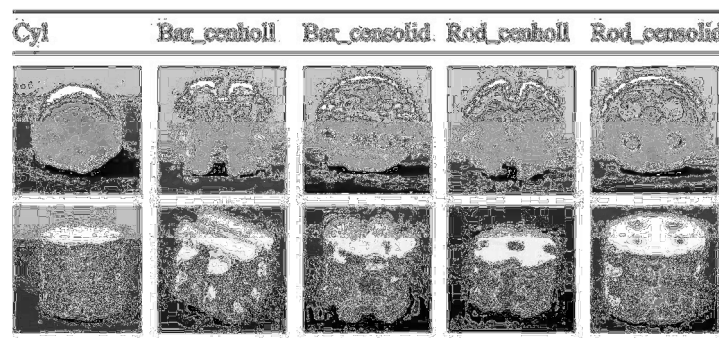


Figure 2-11: 3D printed tissue scaffolds [56]

Wettergreen *et al.* have also utilised conventional CAD to design a range of cells that can be combined to form functionally graded structures [55]. A range of cell designs have been developed that tessellate orthogonally, termed 'building blocks' across an example human vertebral body. To ensure that any particular building block can connect with any other in the range, each design includes a torus-shaped common interface, as shown in Figure 4-4.

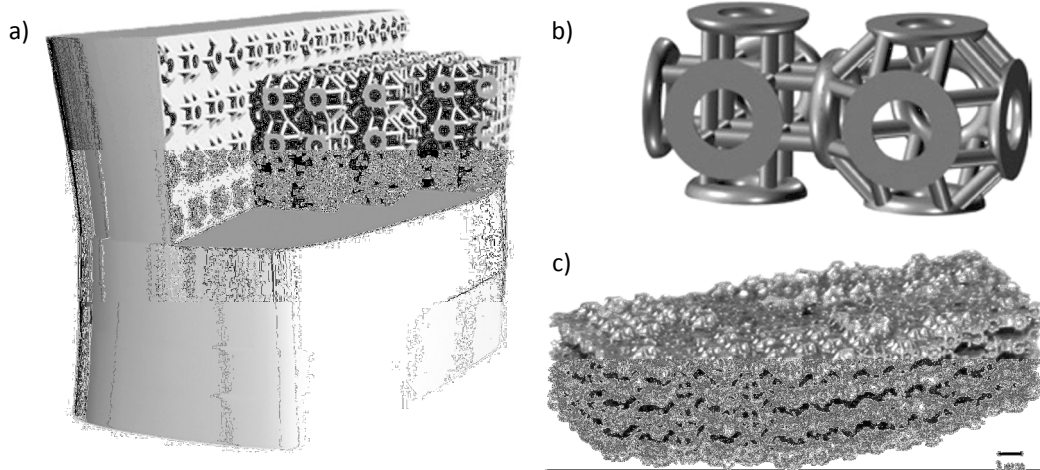


Figure 2-12: a) Cell placement, b) cell designs with torus-shaped common interface visible & c) photograph of fabricated part [55]

An approach to modelling complex structures with CSG was investigated by Schroeder *et al.* [65]. A mathematical framework was developed to model the heterogeneous structure of natural porous materials such as bone. The method utilised CSG due to its efficient application of Boolean operations, as the structures are generated by subtracting spheres from a part, as shown in Figure 2-13. While capable of constructing these complex, randomised structures, the method is limited in the overall conformal shape that encloses it. Because CSG can only construct models from the combination of simple primitives, this method can approximate the porous structure of bone, but it cannot approximate a bone's overall shape.

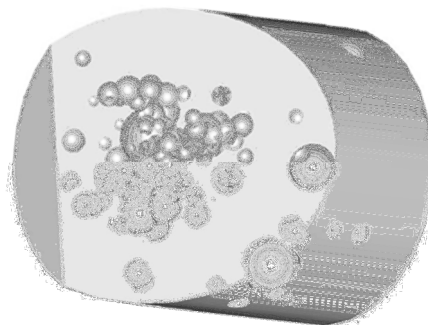


Figure 2-13: Porous structure modelled in CSG [65]

### 2.1.3 Suitability of Conventional CAD for Structure Design

#### *Geometric Complexity*

While the underlying methods of B-rep and CSG modelling allow for the generation of complex shapes through freeform surfaces or primitive combinations, conventional CAD is unsuitable for the generation of large, complex structures. Naing *et al.* showed that the parametric nature of commercial CAD software was well suited to the construction of relatively simple structures [57,81]. This was further exploited by Gervasi and Stahl and Wettergreen *et al.* in the construction of functionally graded structures with varying geometry throughout [55,82]. However, the relatively simple structure designs produced by these are as a consequence of conventional CAD systems being tailored towards the constraints of traditional manufacturing systems [83]. They do not exploit the geometric freedom afforded by additive manufacture [1].

Additionally, the overall sizes of the generated structures were relatively small: on the scale of bone segments with only a few hundred cells per part. Wettergreen *et al.* found, when constructing the vertebral body structure, that anything larger could not be attained with current CAD software due to limited computing resources [55]. When attempting to generate structure arrays on a larger scale, many other groups have found conventional CAD unsuitable for the task: limited on the number of surfaces that can be constructed [2].

The Boolean functions inherent in CAD systems were utilised to unite arrays of cells together and, in some cases, to intersect with a bounding shape to generate a trimmed, conformal structure [56,57]. In a pure CSG environment, as stated in Section 2.1.1, Boolean operations are integral to the method, however they are the most demanding component of a B-rep environment [84]. This is the case when a conventional CAD modeller is used to generate large arrays of structure because these Boolean operations are required for each and every strut in the structure [50,85]. While computational resources of modern PCs and CAD software continually improve, the requirement to generate large structures is a current issue.

#### *Model Structure*

This parameterisation of geometry by conventional CAD modellers is tied into a construction history that not only describes the model but the order in which it was created. While useful for assigning relationships between features of a model (as well as giving context to individual topographic features), this structure also hinders the modifications to a model that weren't considered by the

original designer. Changes to a model must cascade down through the construction history tree to check for dependant relationships between features. For this reason, modifying large models with complex history trees is an prohibitively slow process, as the entire branches of the model tree must be updated for every minor modification [78]. This is particularly noticeable when modelling large, tessellating structures as the branches of the tree become extremely long as individual cells are constructed and duplicated.

### Model Validity

For a CAD model to be suitable for manufacture, it must be unambiguous as to what lies inside and outside of the model; the model must be solid. While CSG defines solids as standard, B-rep does not. A single surface defined by boundary representation could not be manufactured because it has no thickness. A solid model is only defined in B-rep by a group of surfaces that perfectly match up at their boundaries [1]. As a B-rep model gets more complex – such as when tessellating a cell to form a structure – the likelihood increases that slight errors will occur that mean surfaces do not match up exactly [60]. Similarly, it is possible to create invalid geometries that, in the virtual environment of B-rep modelling obey checks (such as Euler's characteristic), but could not exist in the real world [73]. An example of a model that would be impossible to manufacture is shown in Figure 2-14.

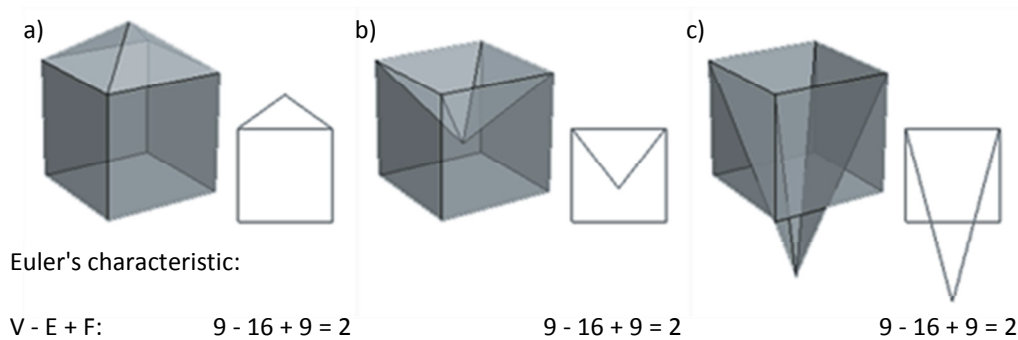


Figure 2-14: a & b) valid models & c) invalid model, but each with a Euler's characteristic of 2 (based on a figure from [73])

In summary, the limitations of conventional CAD software are particularly notable when modelling lattice structures. CSG cannot represent all shapes that may be required when designing lattices. Although B-rep is much more flexible in terms of geometry construction, the Boolean operations required to combine many thousands of individual structural elements are so computationally demanding that it makes the use of the method unfeasible. Additionally, there are issues with model validity with B-rep.

The next section details geometry creation methods that avoid conventional CAD, the first step of the conventional route from design to additive manufacture. Rather than generating conformal structures in CAD software, which must then be converted to STL and then a slice format, geometry is directly manipulated at the STL stage.

## 2.2 Stage 2 of Conventional Route - Generating Lattice Structures at STL

### 2.2.1 The STL File Format

The STL format is the second stage of the conventional route of data flow, a stage that exists as a neutral format that any CAD software can convert to, and any slice format can be produced from. It has been argued that including slicing algorithms in every CAD system would be cumbersome, as a range of algorithms would need to be included to cover every format in use [2].

At present, the STL format is widely accepted as the *de facto* standard for the translation of data for all AM processes [1]. Most CAD packages include export functions that convert a CAD model into an STL file. An STL model is a faceted boundary representation of a model: a surface model composed of triangular polygons and as such can only ever approximate curvature. An example of an STL file is shown in Figure 2-15. There are a number of other neutral file formats that exist to exchange data between CAD systems, such as IGES (initial graphics exchange system) and STEP (standard for the exchange of product model data) [72,86]. The STL format, however, was developed specifically for the translation of CAD model data to AM systems [87].

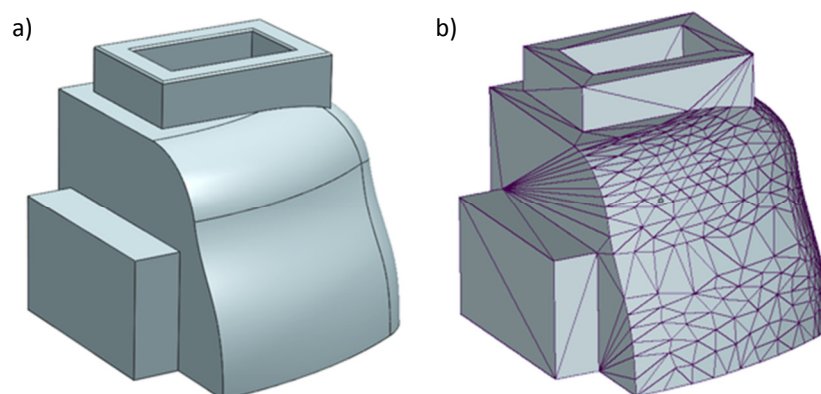


Figure 2-15: Example geometry as a) a CAD model & b) an STL model

A solid model is represented as a closed surface, with the inside and outside distinguished by the triangle normals. This is often referred to as a 'watertight' model [88-90]. Each polygon in an STL file



is described as a set of three vertex co-ordinates and a normal vector, although the vector is superfluous as the polygon's normal can be interpreted by the order in which vertex co-ordinates are listed. There are several types of STL format. The ASCII format is often used in prototype software as it has a simple code structure, whereas the binary format is more compact in its representation of geometry and thus most suitable for normal use [2,63]. A colour STL format allows colour information to be stored, for use with processes such as 3D Printing [91]. A portion of STL ASCII code is shown in Figure 2-16.

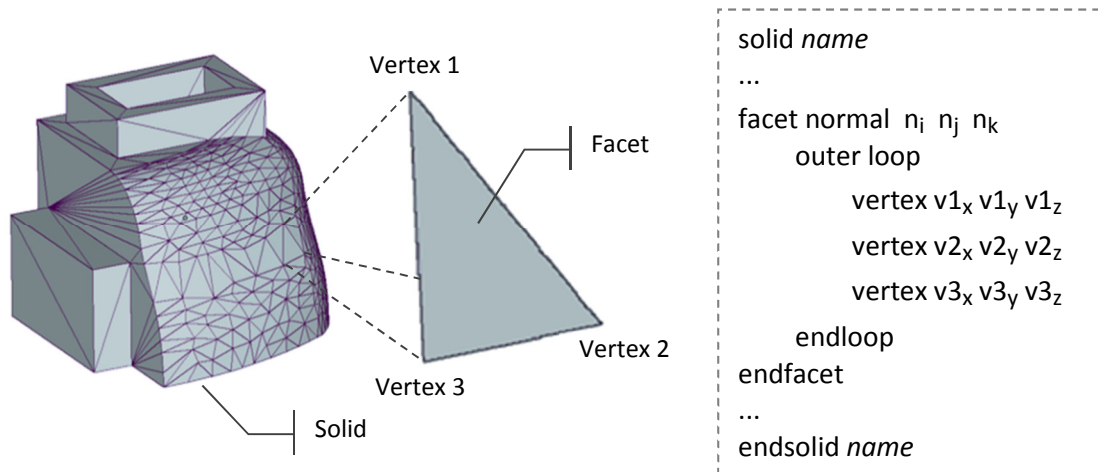
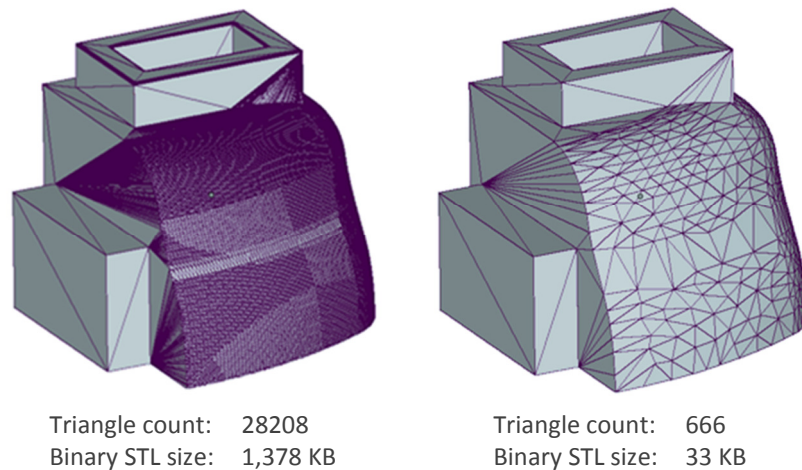


Figure 2-16: The ASCII STL file format

Because of the manner in which surfaces are triangulated during the conversion process, certain geometric features require a large number of triangles compared to their complexity. Polygon count is dependent on surface curvature. Any curved surface must be decomposed into a number of flat surfaces in a manner analogous to representing a circle with an n-sided polygon. At a fine tolerance, a circle may be represented with a polygon of hundreds of edges, at a coarse tolerance, a hexagon. A flat, square surface is efficiently represented with two triangles; however as soon as some complexity is added to an edge of this surface, the number of triangles increases dramatically.

The degree of faceting can be controlled in the conversion functions of most CAD software, two examples shown in Figure 2-17. A fine tolerance will produce a surface that more closely represents the original model, but at the cost of a larger file size, due to the increased number of polygons required. A compromise between model accuracy and file size is often an issue that must be considered with STL models. The level of faceting must be fine enough that it is not visible on a manufactured part, while not too fine to make file sizes too large for practical handling.



*Figure 2-17: The relationship between triangle count and size of a binary STL file*

Tolerances can be set so high that generated STL models can be significantly larger in file size than the CAD model that it has been exported from. Conversely, for a particular geometry it is usually possible to define a set of tolerances that constructs an STL of suitable accuracy that is significantly smaller than the original CAD model. Manipulating these STL models will demand less computer memory which - in the case of generating repeating structures - implies that a higher hierarchical complexity can be achieved before a computer's memory limit is reached. The next section details structure generation processes that have been developed to capitalise on this potentially reduced memory requirement.

### 2.2.2 Applications - Processes that use STL

Several commercial software packages are available that are able to generate lattice structures at the STL level. As previously stated, Materialise Magics is an STL checking and manipulation program, however it also includes an optional structure generation module [92]. A watertight model can be populated with one of a small library of structure types and the structure is trimmed to fit it. The option to add a solid skin around the structure is also possible. Another STL checking program that includes a structure generation module is from AutoFab [93]. Like Magics, an STL structure is constructed to fit a model and a solid skin can be applied as shown in Figure 2-18. In addition to a small library of included structure types, both Autofabb and Magics structure generation modules allow user-designed cells to be added. These can be modelled through any method, as long as it is finally converted to an STL model.

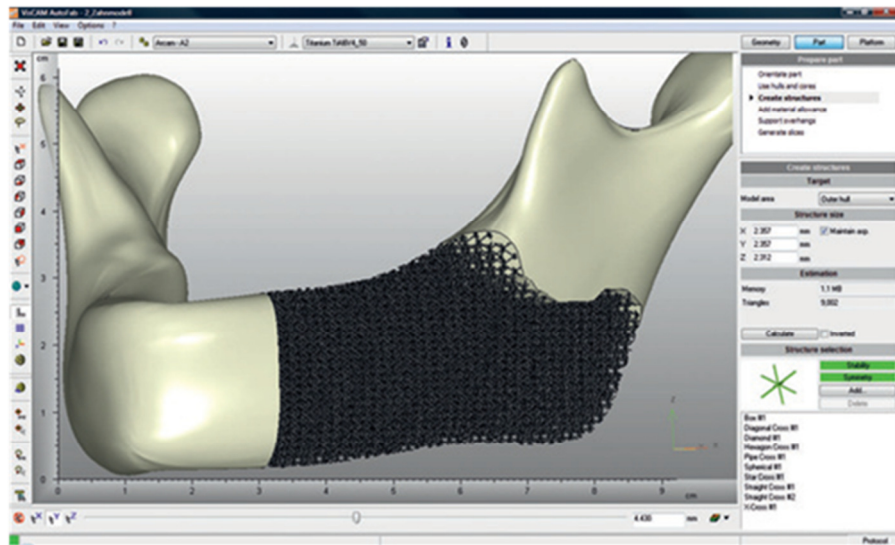


Figure 2-18: AutoFab structure generation software - showing lattice structure and solid skin option [93]

A method has been developed by Gibson *et al.* that constructs conformal structures with STL shells, which takes advantage of the reduced memory requirements of the format [50,71]. This 'swept structure' method is discussed in detail in Chapter 4. A conventional CAD module (ACIS) is used to generate individual cells which are then converted to STL and positioned as such to form a conformal structure [50,94]. This method constructs conformal structures by deforming each cell to fit a curved surface - an example shown in Figure 4-8.

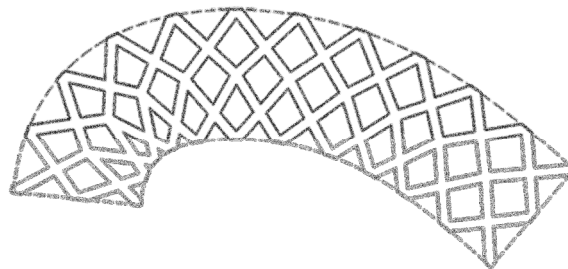


Figure 2-19: Conforming a structure to a shape through sweeping

Essentially, a B-rep surface model of each cell is individually generated in ACIS - deformed to fit a specific location within the warped structure - which is then converted to STL. These models are not watertight - the flat faces at the boundary of each cell are removed, as shown in the inset of Figure 2-20. The conversion algorithm is configured in a way that ensures the vertices of the STL model at the edges of the cell are coincident to the edges of neighbouring cells when stacked to form the conformal structure [50,94], as shown in Figure 2-20. Because the vertices (and by extension, polygon edges) of neighbouring STL cells match exactly, a watertight structure is generated without

the need for further Booleans [50]. As discussed in the previous section, Boolean operations are the most demanding component of any B-rep modelling environment.

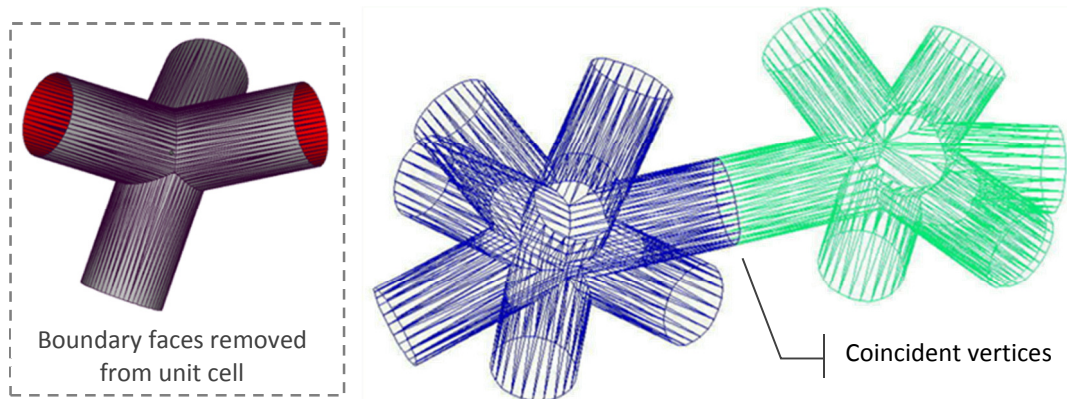


Figure 2-20: Watertight connection between two unit cells [94] & inset: surface model of a unit cell

This method combines an advantage of a conventional CAD modeller (the ability to parameterise the cell geometry) with the reduced memory requirements of the STL format in the construction of the conformal structure. An example of the complexity achievable with the method is shown in Figure 2-21. Cell geometry can be constructed with varying dimensions in the first stage of the method, which has facilitated investigation into the generation of functionally graded structures [71,88]. However once a cell has been constructed and stacked into the structure, it can no longer be modified.

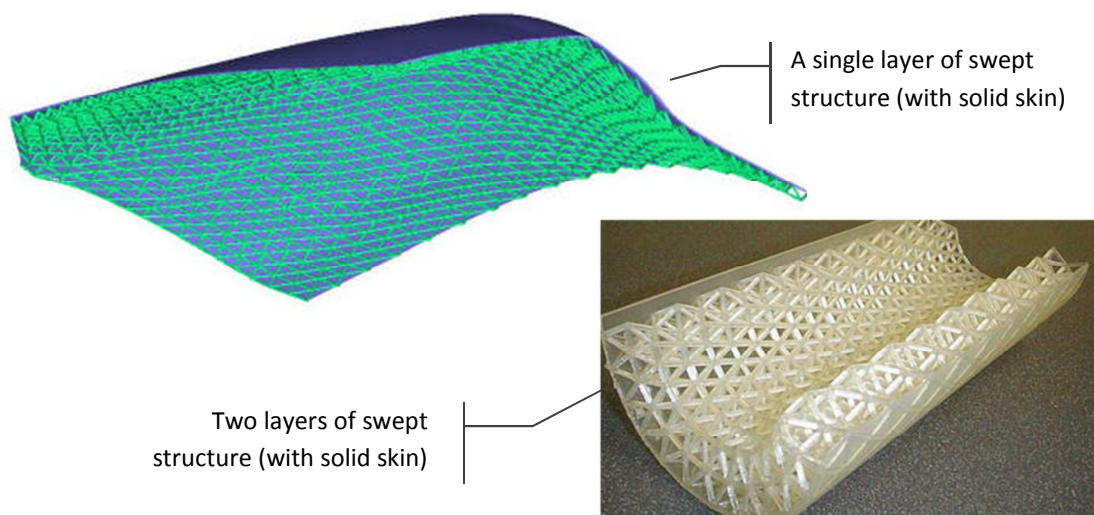


Figure 2-21: Examples of the complexity achievable with the Gibson et al. conformal structure method [71]

The avoidance of Booleans is the major advantage of this method, a technique that capitalises on the faceted topology of an STL surface. However, this is not transferrable to all methods of generating conformal geometry; it is specific to this structure sweeping method. STL cells are lined up perfectly

to construct an already conformal structure that is essentially watertight without further modification. With this structure sweeping method, the final structure topology is determined before any actual structure is constructed. If this geometry construction technique was applied to the trimming method, this advantage would be lost. A regular, bulk structure could be efficiently constructed, but to be made conformal it must be trimmed or intersected with a shape; significant Boolean operations would then still be required to trim this regular structure.

As stated in the previous section, STL file size is dependent on number of triangles, which in turn is dependent on both the coarseness of faceting and the actual shape and surface curvature of the model. A flat surface requires significantly fewer triangles than a curved surface. In the context of structure design, a structure comprised of triangular struts will be represented by a significantly smaller file than a similar structure of cylindrical struts, as shown in Figure 2-22. Several STL structure generation techniques capitalise on this simplification of geometry to facilitate the generation of many-celled structures.

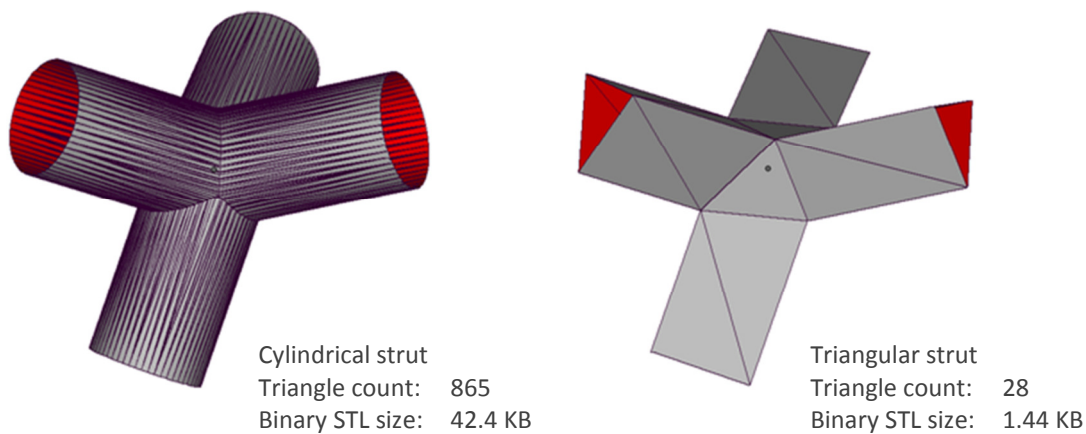


Figure 2-22: Efficient geometry design in STL modelling

The general-purpose 'TetraLattice' developed by Gervasi *et al.* has many proposed applications from expendable patterns for investment casting to conformal filters [95,96]. The structure design facilitates large tessellations at the STL level because only six triangles are required to form each strut, as shown in Figure 2-23. This structure design highlights how the reduced memory requirements of the STL format can be exploited at the design stage.

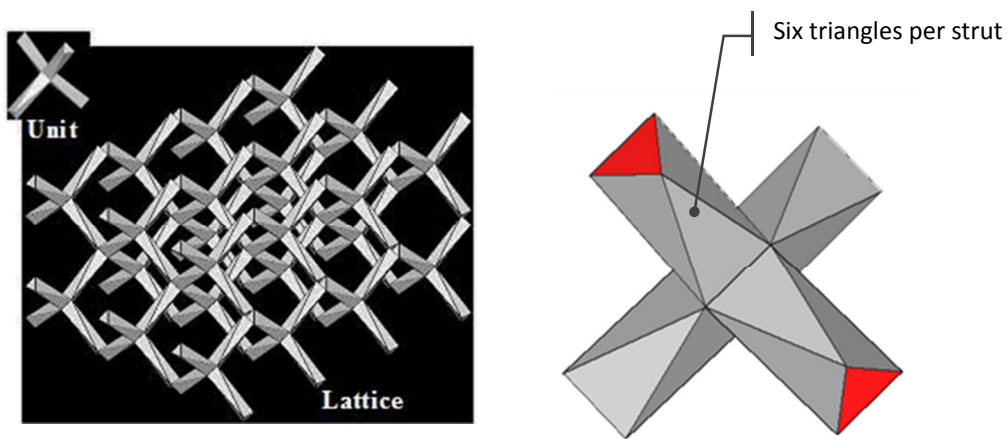


Figure 2-23: TetraLattice structure [96]

### 2.2.3 Advantages and Disadvantages of STL manipulation

#### *Design Environment*

Generally, methods based around STL manipulation are capable of generating larger structural arrays when compared to conventional CAD methods. This is essentially because certain geometries can be constructed which require less memory to manipulate. However, conventional CAD is still often used at some point to design the structure's unit cell, as no robust STL design environment exists.

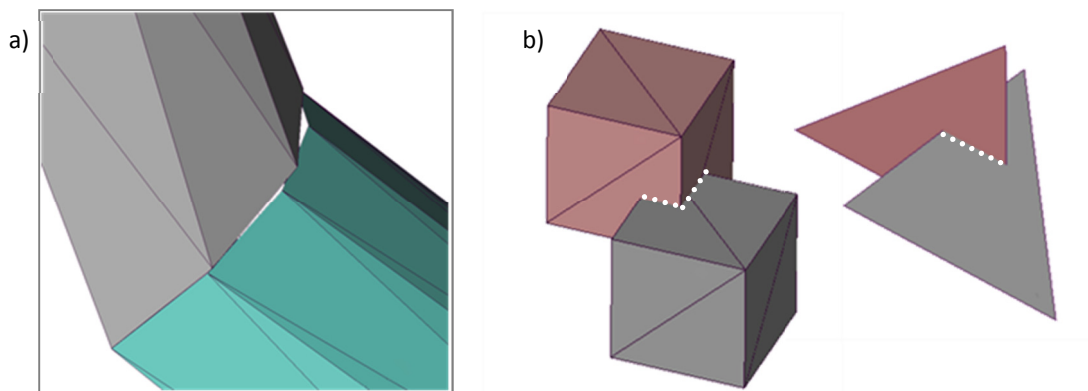
The lack of an ability to parameterise geometry (for future modification) is a significant disadvantage which removes flexibility from the design process. There is no built-in knowledge within the STL format that defines particular features of a design. Whereas higher-order B-rep models define surfaces according to their topology (such as flat faces or cylindrical faces) an STL model decomposes every surface into a set of flat polygons. The original knowledge of the surface type is lost. This is significant because designs are usually parameterised around these surface types. For example, a structure's 'strut diameter' may be a common structure dimension to vary, which in advanced B-rep would directly correspond to a set of cylindrical features that could be located and subsequently modified. The triangles that represent the faceted version of those cylindrical struts do not stand out in the STL model and so cannot be easily isolated for modification.

Radical changes to geometry that are useful are not permitted as the topological information of a design is lost in conversion. Software such as Magics can attempt to re-distinguish such topological features as flat faces or curved surfaces or a part, as well as allowing manipulation of individual

polygons. Certain transformations can be applied to these selection groups, however the usefulness of this is limited.

### *Manifold Errors*

As a boundary representation, the STL format is prone to certain errors without robust checking, much like conventional CAD [60,82,97]. An edge of a solid is defined where two faces meet: depending on the CAD software used for modelling, these two faces may be two separate surfaces. If this information isn't preserved, the two surfaces may be triangulated in such a way that their faceted representations do not match, as shown in Figure 2-24. This is because it is possible for different triangulation patterns to represent the same geometry [2]. Another potential error can occur when surfaces overlap, for example, two surfaces from different bodies that haven't been combined with Booleans. Triangles will intersect where these surfaces overlap which may cause ambiguity in shell watertight checks [98].



*Figure 2-24: Manifold errors: a) mismatching connecting surfaces and b) intersecting triangles*

Errors are most likely to occur because a conversion algorithm doesn't include robust checking, however most modern algorithms do not allow such errors to be generated [2] and STL checking and pre-processing software such as Materialise Magics offer an extra level of checking [92]. A simple calculation, derived from Euler's characteristic, to determine if a particular model is watertight is shown in Equation 2-2 [2,99]. Additionally, there is ongoing research into slicing STL files with errors: an STL is sliced and any defects are repaired in 2D across the slices [98].

$$\frac{\text{no. triangles}}{2} + \text{no. vertices} = 2 \times (\text{no. bodies} - \text{no. passages})$$

*Equation 2-2: Watertight STL calculations ('passages' refers to holes going all the way through a body) [2]*

In summary, the STL file format allows the construction of larger structure arrays compared to conventional CAD, due to reduced memory requirements for faceted geometries. Structure geometry can be optimised to take advantage of triangular polygon surface topology by minimising the number of triangles required to represent geometry. However, these optimised geometries aren't particularly complex - the complexity afforded by AM processes isn't fully realised. Inevitably, direct STL structure generation still reaches a memory limit that prevents the representation of larger, more complex structures. The STL modelling environment is inflexible and lacking - as the second stage of the conventional route, the majority of design work is expected to occur at the CAD stage. STL geometry must be converted to a slice format before manufacture, so any gains in modelling at the STL stage must be compared with the further operations required to translate the data into a slice format. The next section discusses several methods developed that bypass both CAD and STL stages of conventional route by generating conformal structures at slice level.

## 2.3 Stage 3 of Conventional Route - Generating Structures at Slice Level

### 2.3.1 Slice formats

The final stage of the conventional route of data flow that can be accessed and manipulated is the slice file. A slice file represents a model in a layer-by-layer format, corresponding to the layer manufacturing technique of AM processes, as shown in Figure 2-25. Conventionally, a slice file is derived from an STL model. Direct slicing of a CAD model has been investigated, however due to the sheer range of CAD formats in use, it is currently more feasible to use the neutral STL format as a starting point [100,101].

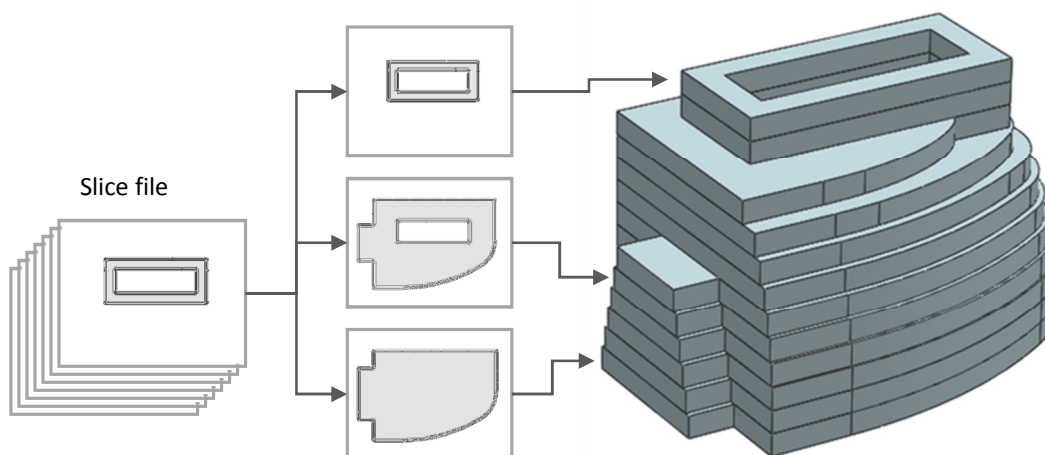


Figure 2-25: Layers of a slice file corresponding to AM process' layer manufacture



The format of a slice file varies between AM process, and even between manufacturers. Broadly, slice formats can be split into two categories: Vector formats and raster formats [102], illustrated in Figure 2-26. Vector formats are used by AM processes such as laser sintering, stereolithography and FDM. Vector slice files define geometry with 2D splines and polygons, comprised of vertices and edges. These are analogous to the scan lines of the AM processes, such as the laser path in laser sintering. Many of these formats are proprietary, such as 3D System's SLC format and EOS' SLI format for their own laser sintering processes. The common layer interface format (CLI) is an open source alternative [103]. Raster formats are essentially an ordered series of bitmap images. The term 'raster' comes from the field of computer graphics, where it refers to both the pixel array that comprises bitmap images and that of monitors and televisions. In a raster format slice file, pixels of a particular value define where material is deposited. 3D printers build from raster slice formats, much like a standard printer prints bitmap images.

There are several methods to convert an STL into a slice format. For vector formats, a plane can be intersected with the model, each polygon edge at a time. From this, individual intersection points are generated which can then be connected to form an outline of the part at the height of that plane. The plane can be moved to the height of each layer to completely slice the model [104].

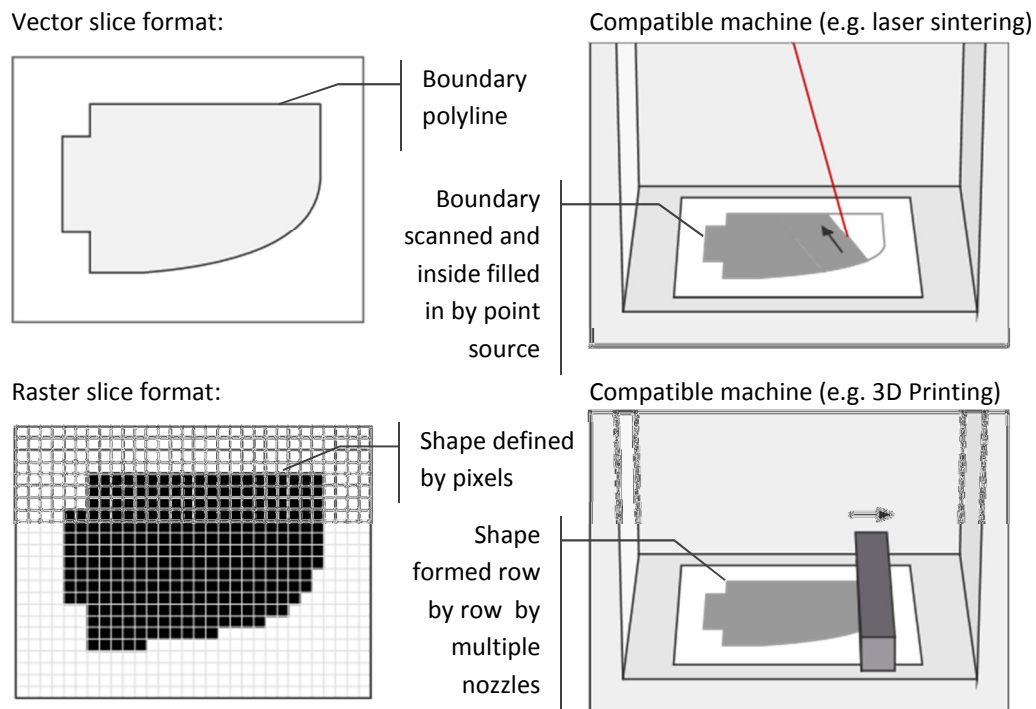


Figure 2-26: Relationship between slice format and AM machine type

A method to generate raster slices involves performing a Boolean intersection between the model and a single layer-thick cuboid, for each layer. The result is then rendered from a top-down

perspective and saved as an image file. This method could be employed in, for example, the rendering software POV-Ray [105]. While conversion from vector to raster formats is trivial, the reverse is not. As all computer graphics are displayed on a monitor comprised of pixels, even a vector output, when drawn on the screen is represented by pixels. The location of the pixels can be captured to generate raster files.

Rather than generating slice files from any other geometry representation, some groups have investigated actual geometry construction at the slice level. These are discussed in the following section.

### 2.3.2 Applications - Processes that use Slice Files

‘Selective Space Structures’, the commercially available software from netfabb, generates conformal structures in a slice by slice approach [106,107]. Like most of the methods detailed in this chapter, the conformal structures are constructed by trimming a regular structure to an input shape. An input shape (in STL format) is first converted to a vector slice format by the software, and a structure type is selected by the user. The structure type is stored within a library also in a vector slice format and is overlaid over the input shape, as shown in Figure 2-27. To trim the structure in this format, the operation is a simple series of 2D Boolean intersections. A series of discrete, less computationally intensive operations, this method is a more robust alternative to a single 3D operation, such as when trimming CAD or STL models.

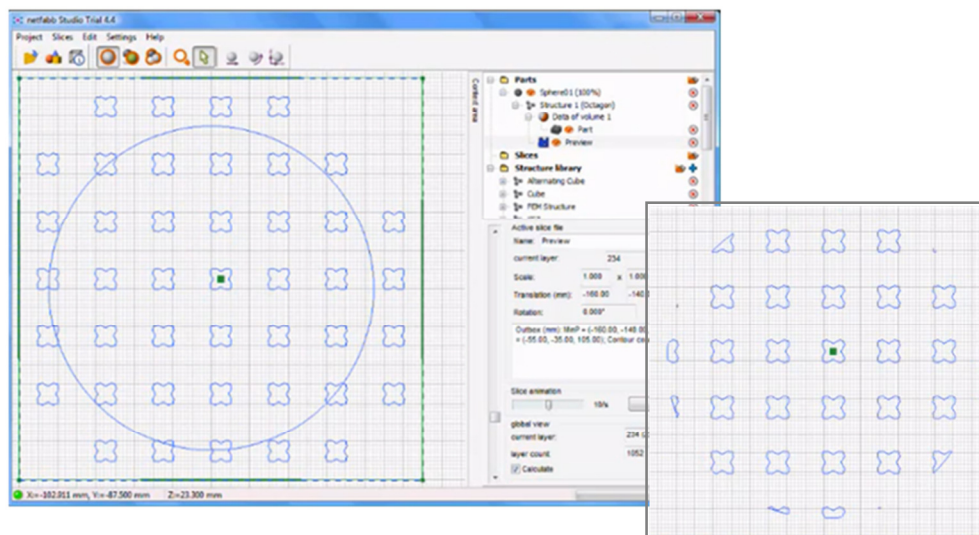


Figure 2-27: Trimming a structure to a shape at the slice level with netFabb Studio software [107]

By constructing a part at the slice stage, it becomes difficult to visualise it, thus reducing the level of confidence a user has in the design. Viewing the part as a series of cross-sections is no alternative to

viewing it as one 3D model that can be explored and verified. In an effort to address this, netfabb provide an approximate representation of the trimmed structure before the Boolean intersection operations are completed, as shown in Figure 2-28.

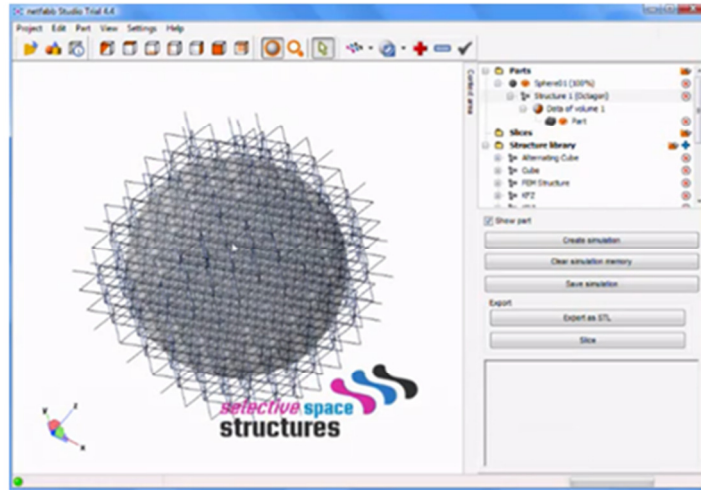


Figure 2-28: Approximate trimmed structure visualisation in netFabb Studio [107]

Rather than actual, solid geometry, the visualisation is a wireframe model of the structure. The wireframe model is not actually trimmed to fit the shape, rather the structural cells that would intersect with the shape are displayed as an approximation of the conformal structure. In addition to the Selective Space Structures software, netfabb also sell the 'Structure Generator' add-on that allows users to design or import new structure types, rather than being constrained to the library of structure types built into the original software.

Although a more efficient approach to conformal structure generation than both standard CAD and STL techniques, its speed is still dependent on structure and shape complexity. Any more complex structure designs imported through the software's add-on will take longer to process. In a similar manner to how a complex shape must be represented with a larger number of triangles in the STL format, slices of a complex shape must contain a greater number of polylines to represent complex cross-sections with this method.

A unique method of generating trimmed structures in a layer by layer approach has been developed by Brooks *et al.* [108]. The software, 'Manipulator' was developed in conjunction with an actual Selective Laser Melting process (SLM) at MCP (now Renishaw AMPD) and the University of Liverpool. This structure trimming method is unique in its control over how the laser sweeps across the powder bed, giving unprecedented control over the construction of fine metal structures [52,53].

A series of cubic cells that are comprised of combinations of 'pillar', 'diagonal' and 'octahedral' struts (shown in Figure 2-29) are available for selection. The SLM process requires support structures to build anything at an angle less than 30° to the build layer. It is for this reason not possible to incorporate such 'low angle' struts or horizontal struts into the cell designs [52,53].

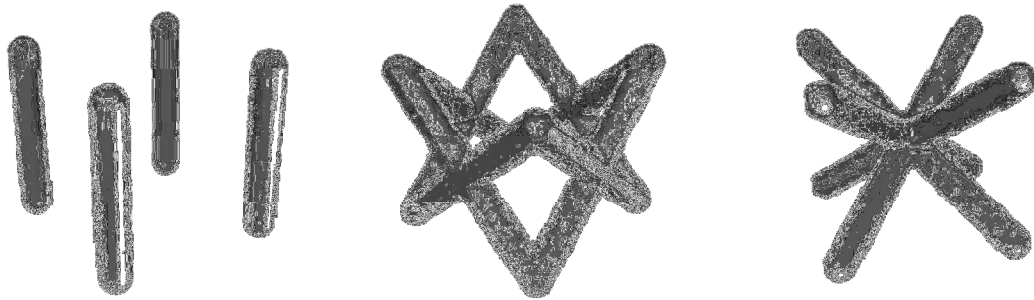


Figure 2-29: Pillar, diagonal and octahedral struts

The method is capable of generating extremely fine metal lattice structures, examples shown in Figure 2-30. For each layer of a standard part manufactured in SLM (or indeed any laser based AM process), the outline and fill are separately traced onto the powder layer to melt material together. For very fine geometries, this will overexpose the area to laser energy and melt a wider pool of metal powder together. The end result will be thicker geometry than required. To overcome this, Manipulator utilises a single laser point exposure for each strut on each layer to achieve fine geometries, equal to the melt pool of the single laser point [53].

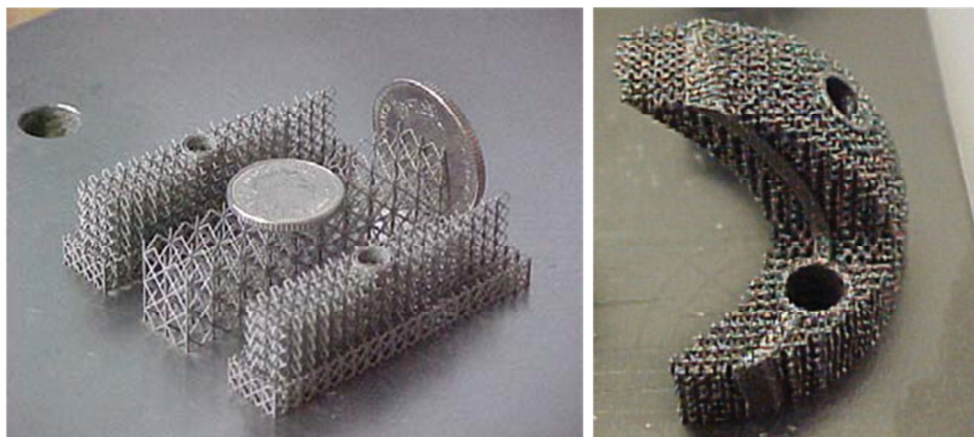


Figure 2-30: Fine metal structures produced with Manipulator [53,109]

Rather than inputting a standard slice file to the machine, a series of scan vectors are calculated for each of the pillar, diagonal and octahedral strut types shown in Figure 2-29. These scan vectors are calculated from the corners of each cell. For example, the laser points for each layer required for a particular diagonal strut are interpolated from corner co-ordinates [109], as shown in Figure 2-31.

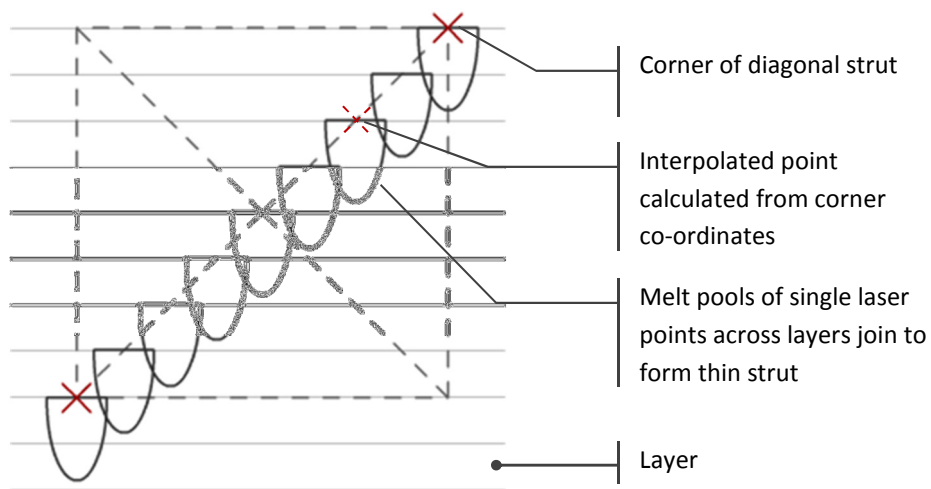


Figure 2-31: Side view of interpolated laser profiles of a diagonal strut in Manipulator software

The method has also been modified to produce pseudo-randomised structures that more closely approximate the structure of trabecular bone for use in orthopaedic applications [109].

A novel layer-based approach to the construction of structure geometry has been investigated by Chow *et al.* Structures are generated with a series of 2D Voronoi diagrams - a means of subdividing space determined by a set of points (as discussed in Chapter 2) [97]. Voronoi diagrams are often used for modelling the random structure of foam, however through judicious placement of seeding points, Chow *et al.* have demonstrated specific, regular structure designs can be constructed, examples shown in Figure 2-32. The inside and outside of the structure are defined by assigning either a solid or void value to corresponding seeding points [97].

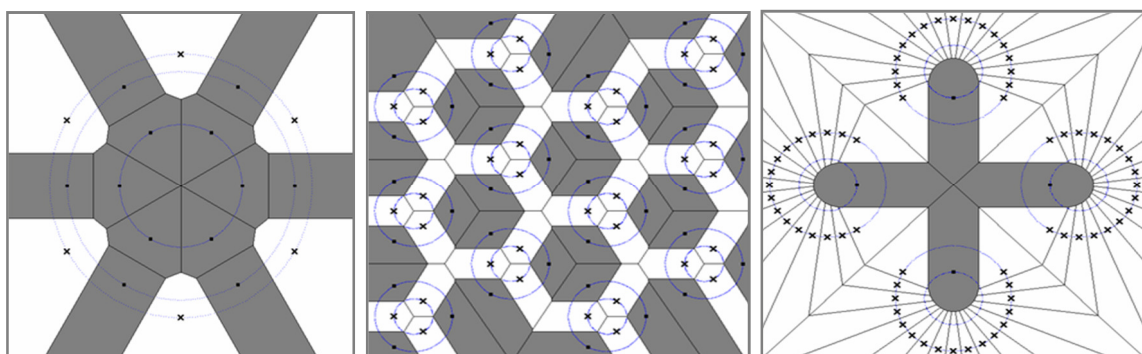


Figure 2-32: Constructing regular tessellations with structured placement of seeding points [97]

Each layer of the structure is described with a different Voronoi diagram. To generate successive layers of the structure, the seeding points are animated; each point has a velocity specified to it which warps the Voronoi diagram accordingly. Discrete steps of this motion correspond to each slice

of the structure. This is illustrated in Figure 2-33: points travel in the direction of their arrows to join two nearby struts together. By grouping the seeding points in rings (shown in Figure 2-32), the structure is parameterised - changing the ring diameter modifies the structure in a controlled and specific manner [97].

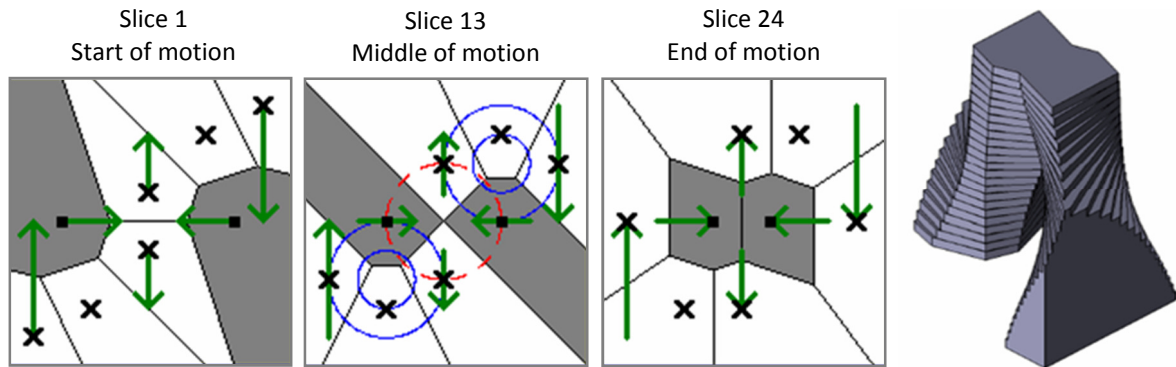


Figure 2-33: Generating multiple slices of structure by 'animating' seeding points [97]

Due to the nature of Voronoi diagrams, complete, watertight boundaries are always formed. This is a robust method of boundary representation when compared to conventional CAD methods and the STL format. However, the user must be familiar with the relatively abstract technique of manipulating Voronoi diagrams to fully exploit this method. Although constructing geometry in a layer by layer approach, this method still currently uses a conventional CAD modeller (ACIS) to generate the geometry [97]. Thus it benefits from one advantage of manipulating slice files (breaking down potentially complex 3D operations into a series of simpler 2D operations), although still suffers from the resource-expense issues of conventional CAD. However, it would be a relatively trivial task to format the output for direct slice file writing.

Structure generation methods have been developed specifically to utilise the material deposition process of AM processes. The FDM process constructs a part by laying down an extruded bead of near-molten polymer [19]. The usual process is to lay down hatches of parallel beads to construct a solid part, however various groups have investigated methods to construct lattice structures directly by spacing out these beads [110-112]. Structural elements are build out of single beads, examples shown in Figure 2-34.9

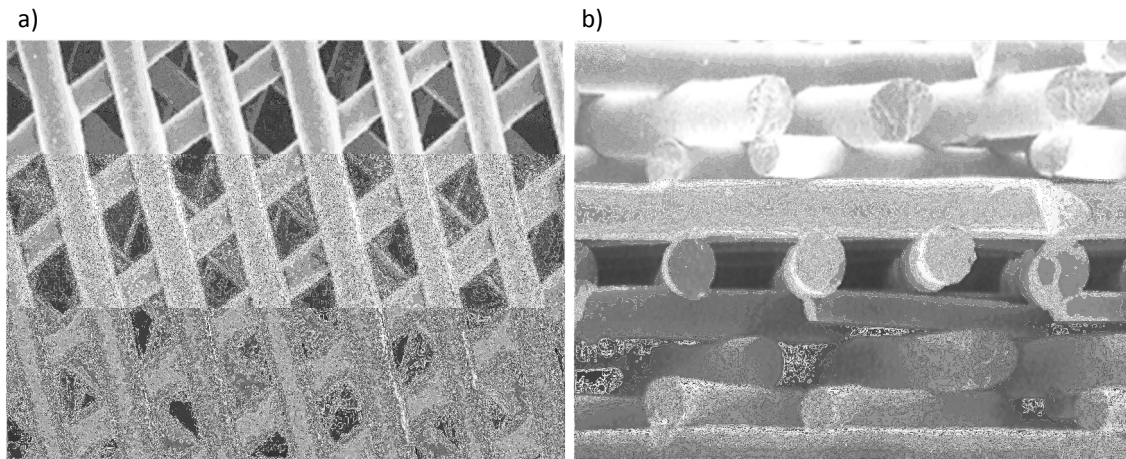


Figure 2-34: SEM images of FDM bead lattice structures [110]

Process parameters of FDM include setting the bead diameter and spacing. Bead orientation is varied between layers as standard in FDM to ensure manufactured parts are not weak in a particular direction [112]. Normally, bead spacing is related to bead diameter to ensure solid parts; by purposefully increasing the spacing, the beads are laid down with gaps between them. Successive layers lay beads at different angles, forming simple lattice designs. A similar technique has been implemented by Stamp *et al.* for the SLM process [113]. The hatch spacing between laser scans is widened to generate the individual structural elements shown in Figure 2-35.

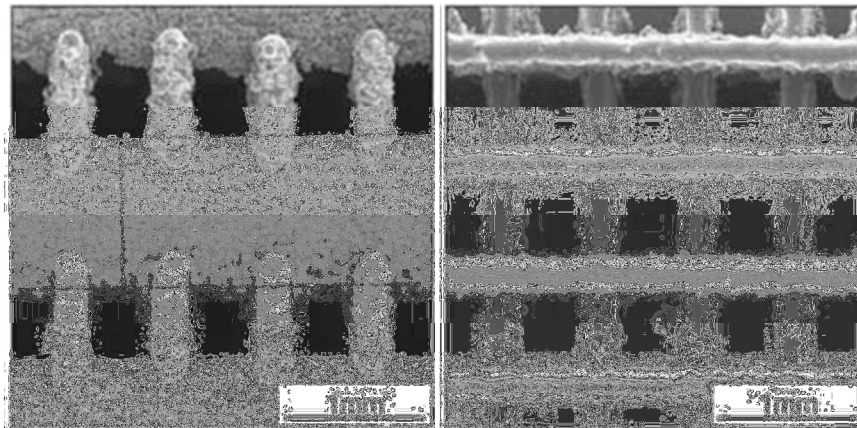


Figure 2-35: SEM images of spaced scan patterns [113]

This method of structure generation is unusual because it utilises the manufacturing process rather than any particular geometry construction method to generate geometric complexity. For this reason, the design information required is merely the conformal shape for structure to populate, rather than a complex 3D structure. Although a much more efficient manner of structure manufacture, this method is limited by the types of structure design that are possible. Only simple

hatch-like structures are possible, where the 'struts' that compose the structure are oriented in the layer plane. The geometric freedom facilitated by AM is not fully exploited.

### 2.3.3 Suitability

The immediate advantage of working at the slice level is that once geometry is constructed, there is no need for further conversions into other formats. With the majority of commercially available AM machines, the slice file is the last step with public knowledge of the format. However, slice formats are specific to their machine - there is no standard format that is independent of technology [114].

#### *Design Visualisation*

There are certain disadvantages with generating geometry at the slice level. When observing 3D geometry as a series of 2D slices, the ability to clearly visualise the design is lost. The ability to clearly visualise a model is an important one – to be able to check manually for otherwise unapparent mistakes, as well as for aesthetic considerations. It could be argued that layer by layer observation is a superior alternative to viewing the model in 3D, as internal geometry isn't hidden, however this benefit does not outweigh the overall loss of clarity. Indeed many conventional CAD packages allow for section views of models, diminishing this perceived advantage of working at the slice level. Although, viewing a model that has been discretised into slices is a more realistic representation of a part manufactured through additive manufacturing, which will be stepped in the same way.

#### *Flexibility of Process Planning*

Working at the slice level also limits flexibility in terms of positioning models in the build volume of an AM process. Parts may be translated relatively easily. Vector slices can be easily rotated in Z-axis (without altering geometry), but not X- and Y-axes, raster slice geometry will be altered with any rotations. Similarly it is relatively difficult to scale parts; if a part is to be increased in size, new layers must be constructed through the interpolation of existing layers. While not impossible, the transformation of parts at the slice level is not ideal. As such, extra care must be taken to ensure correct dimensioning from the start of the design process.



## 2.4 Summary of the Conventional Route

Regardless of lattice structure design or its application, a considerable level of hierarchical complexity is required of the geometry creation method. This chapter has shown that of the three broad methods of the conventional route of design for additive manufacture, manipulating slice formats is most suitable for the construction of lattice structures. A complex 3D problem can be decomposed into a series of simpler 2D tasks. Working at the final stage of the conventional route also has the significant advantage of reducing the number of data conversion steps into a format suitable for a particular additive manufacturing process.

However, what manipulating slice formats gains from higher achievable complexity, it is limited in terms of useability. Although conventional CAD software reaches complexity limits relatively early, a modelling environment exists that facilitates relatively straightforward structure design. The STL format allows the construction of larger structures, however only basic STL modelling software exists. Additionally, the STL format does not lend itself well to parametric or feature-based modelling due to the low-level way in which geometry is stored. This is also the case with most slice formats - vector formats may decompose part cross-sections into 2D primitives such as polylines, but these only have relevance to a particular cross-section, rather than a structural design feature. On top of this, slice formats (being two-dimensional cross-sections of 3D models) do not allow for straightforward visualisation of designs. Any confidence in design decisions is undermined if the user cannot clearly visualise the design.

Although conventional CAD software is the easiest to use, the fact remains that it has been designed to model conventionally manufactured parts. Conventional CAD does not exploit the geometric freedom afforded by additive manufacturing methods. At best, CAD software gives some control over complex freeform shapes such as aerodynamic panels or aesthetically styled devices, however random and organic shapes such as foam or bone structure are difficult to approximate.

It is entirely possible that - with a powerful enough computer, enough time and a particularly skilled (and patient) user - the large, complex structures that the work discussed in the chapter aims towards could be accomplished in any of the stages of the conventional route. To do so however would be an inefficient task when there are other methods to construct geometry that are potentially more suitable for this application. These methods are the focus of the next chapter of this literature review.

### 3 | The Generation of 3D Geometry and the Implications for Lattice Structure Design - Alternate Methods

This chapter details methods of creating virtual 3D geometry that are not part of the conventional route from design to additive manufacture, as defined in the previous chapter. Each method is assessed in terms of suitability for constructing complex lattice structures and any existing attempts are documented. To be able to manufacture designs modelled in one of these methods, it must be converted into one of the stages of the conventional route, as shown in Figure 3-1. Because conversion steps contribute to the overall time taken from design to manufacture, the point at which each method enters the conventional route is taken into consideration.

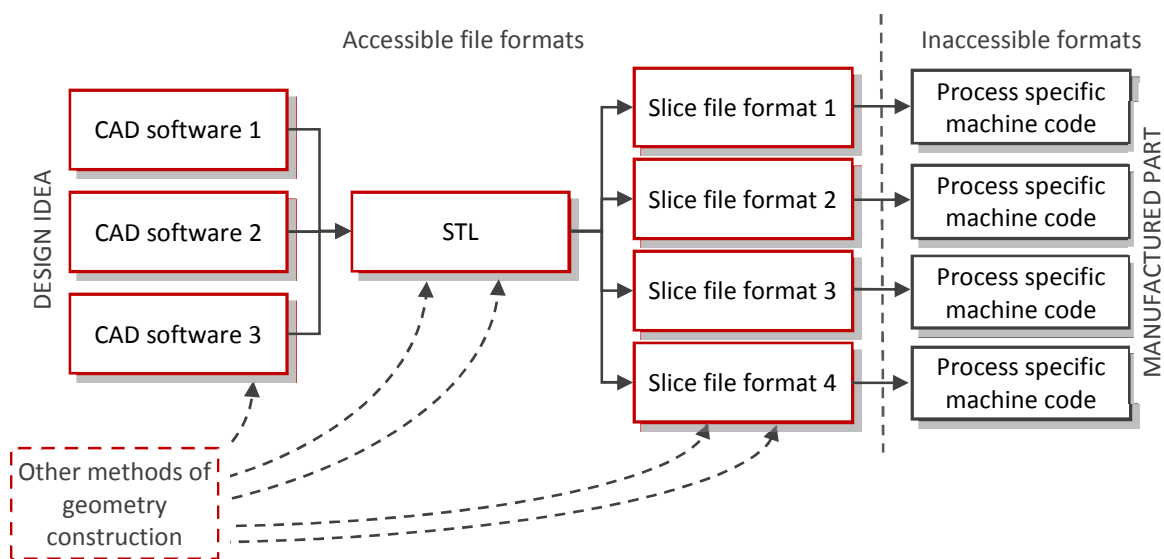


Figure 3-1: Integrating into the conventional route from design to additive manufacturing

The first new geometry construction methods to be considered are commonly referred to as "STL 2.0". The intention is to replace the STL format with a more capable and detailed representation of an original CAD model as the second, neutral step of the conventional route.

### 3.1 STL 2.0

Although the STL format is the *de facto* standard for exchanging data between CAD systems and additive manufacture, it is not without its limitations, as discussed in the previous chapter. Being a form of boundary representation, manifold errors can occur where surfaces don't perfectly meet, producing invalid geometries that cannot be manufactured. Additionally, the format contains redundant data: Each triangular polygon (a facet) is described by three vertices (x,y,z coordinates) and a normal vector. The facet normal is superfluous as it can be calculated by using the right-hand rule on the facet's vertices [115]. Additionally, in the current format a vertex shared between facets is duplicated for each facet that shares it [116]. This is shown in Figure 2-16. This unnecessarily inflates an STL file size and thus the time it takes to read and write.

The STL format is also limited in the information about a model that is translated [115,116]. Surface texture, material properties and the grading of such properties are all areas of potential in additive manufacture that the STL format cannot represent [116]. Some groups have attempted to extend the format to include some of these features, as well as eliminate the redundancy [115,117]. In short, however, the STL format is not future-proof. As AM processes improve, CAD software can be upgraded to incorporate new features - the STL format in its current state cannot.

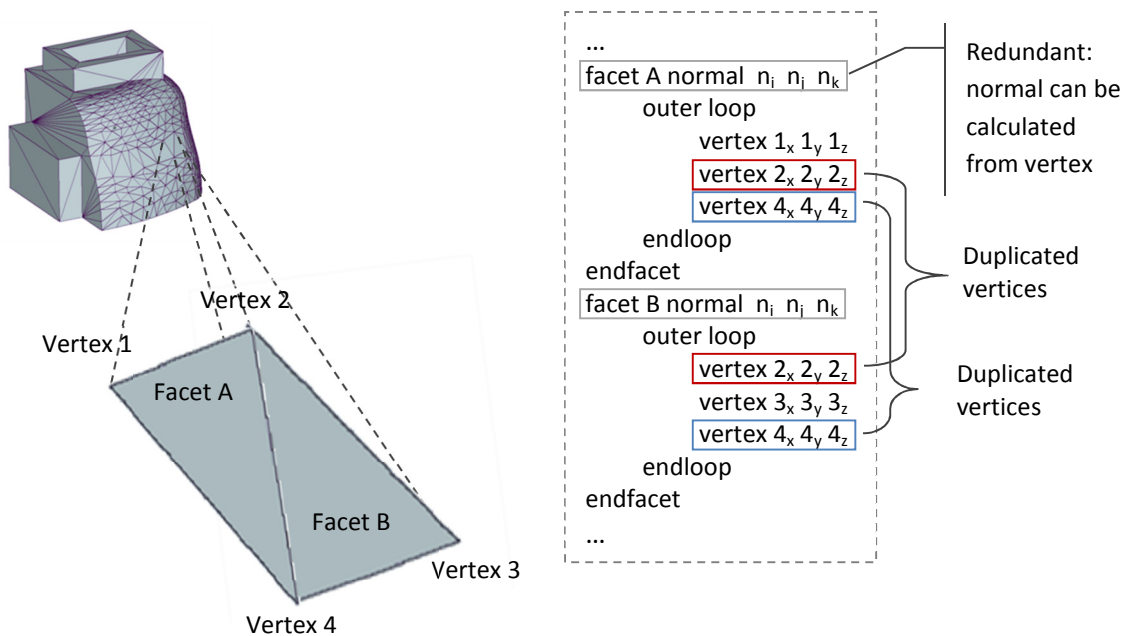


Figure 3-2: Redundant code in the ASCII STL file format (facet A and facet B are neighbours)

Over the last two decades, a number of groups have developed entirely new formats to replace the STL format. Notable examples include: the RPI (Rapid Prototyping Interface) and the LMI (Layer

Manufacturing Interface) [101,118]. The RPI format supports advanced B-rep and CSG modelling as well as the original faceted representation to ensure backwards compatibility with STL [118]. The LMI format builds on the faceted representation while implementing a topological hierarchy and again, eliminates redundancy [101]. Neither of these formats have gained particular peer approval and the STL format has remained dominant.

A more recent attempt to succeed the STL format has been proposed as the AMF (additive manufacturing file) [116]. With future-proofing in mind, the ability to assign materials as well as the ability to define smooth grading between material types are built into the format (examples shown in Figure 3-3). Material categories are initially assigned at the start of the file and each body (or 'region') of the model is assigned one of these materials [116]. Materials can be graded within a region with the use of coordinate dependant equations [116,119]. This works on the assumption that whatever software reads the AMF file already has a library of material properties stored, and that the AM process is capable of depositing this material range.

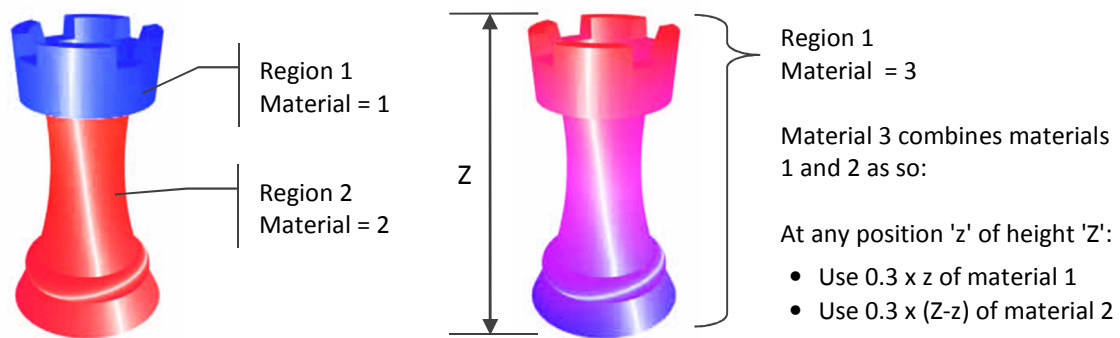


Figure 3-3: Defining multiple and graded materials in the AMF format [116]

Rather than grading materials smoothly across the model, the AMF format allows the definition of lattice structures by tiling materials in discrete portions - a simple example shown in Figure 3-4. Structures may be defined by function representation, a standard STL mesh or with a voxel matrix [116]. Voxels are discussed in detail in Section 3.2 while function representation is discussed in Section 3.3. When using equations or functions to design graded materials or structures, post processing software must be utilised to generate physical 3D geometry to visualise and manufacture.

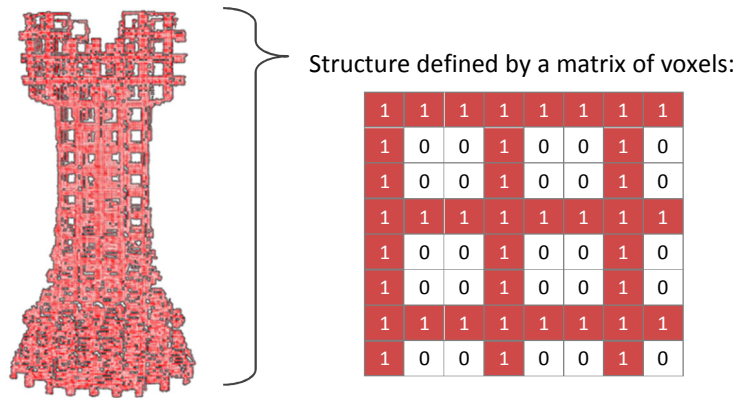


Figure 3-4: Defining a structure in the AMF format [116]

The redundancy of STL files is eliminated by structuring the AMF file more like a standard B-rep model. As stated in the previous chapter, B-rep models can be considered as a structure of lists. A vertex list stores co-ordinates of vertices, an edge list groups vertices into pairs and a face list groups bounding edges into faces. Rather than individually defining each face with three vertices (as the STL format shown in Figure 2-16 does), every vertex is first defined, followed by a facet list (termed 'triangle list') that is defined with three vertex IDs [119]. These IDs point to a single set of vertex coordinates that is only listed once, rather than listing that set every time a triangle that shares it is defined. An example of this structure is shown in Figure 3-5.

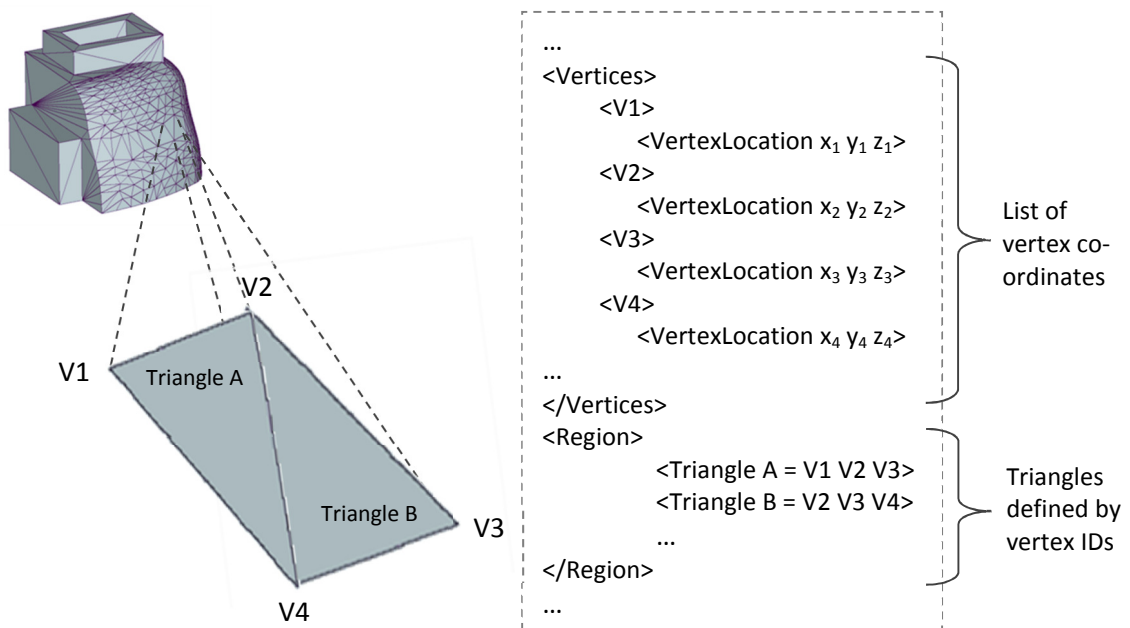


Figure 3-5: AMF pseudo-code, based on [119]

Not only does this structured method of representing triangles eliminate redundancy, but it guarantees that neighbouring triangles align perfectly. If a vertex is shared three times, and written separately three times into a file, there is a possibility that small rounding errors may modify the coordinates slightly between duplicates. This creates holes in the model, making it invalid and in need of fixing. The chance of this occurring with the AMF format is avoided because the vertex is only written once [116].

An STL file requires the calculation of a single vector normal per triangular facet, to determine what is inside and outside of the model. The AMF format allows the inclusion of different vector normals for each vertex of a triangular facet, which can be used to represent curved surfaces, as shown in Figure 3-6 [119]. The triangle to be curved may be subdivided into four new triangles that more closely represent the curved surface. This may be repeated until the desired accuracy of curvature is achieved. All the information required to define curved surfaces can be stored in the AMF format with a single triangle with three vertices [119]. This is merely an efficient manner to translate the data; the actual subdividing to construct a smooth surface must be completed in post processing software to be visualised.

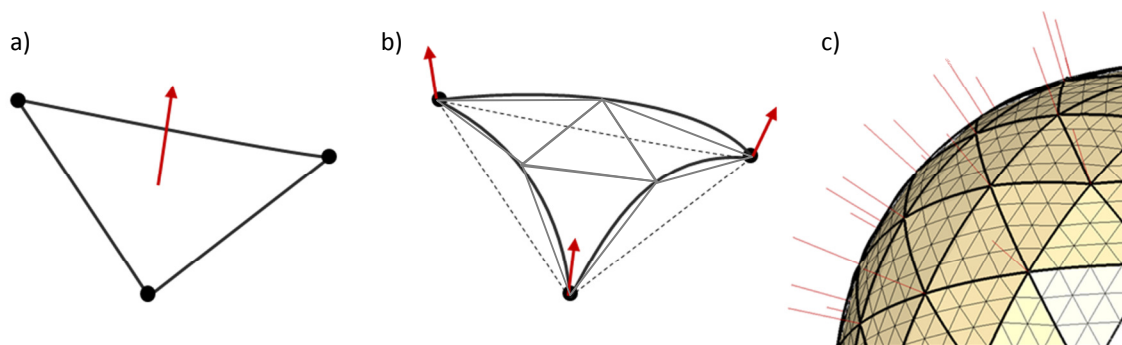


Figure 3-6: a) Planar face with 1 normal, b) curved face with three and c) subdivided further to more accurately represent a curved surface [119]

In summary, the proposed AMF format is a significant improvement over the STL format, although it still suffers from some of the same limitations. Despite being a boundary representation, the AMF format has some control over volume properties. However, volume properties cannot be controlled as explicitly as boundary geometry. Certain manifold errors have been eliminated, however invalid geometries may still be generated in CAD and passed through the AMF format unchecked (such as the 'impossible geometries' described in the previous chapter).

The ability to curve facets with normal manipulation potentially allows a reduction in file size of complex freeform shapes when compared to the STL format. Rather than generating a high tolerance

STL file to capture smooth variations in surfaces, a relatively low number of triangles with varying normals may be written, to be post-processed at a later step. In terms of structure design, the AMF format provides syntax to efficiently store the description of a structure, rather than the explicit dimensions of the structure itself. However it is not a solution in its own right; the AMF is merely a translation language, neither a design tool or post processor. Additional software must currently be used to convert the structure design information into an explicit representation. The choice of this method, be it B-rep or other, will ultimately define the complexity limits of this particular route.

### 3.2 Voxels

Voxels (an abbreviation of 'volume elements') were originally introduced as a means of 3D geometric modelling as an alternative to conventional surface modelling [120,121]. Rather than representing a shape only by its surface (as B-rep methods do), a voxel model defines it as a volume. Analogous to the rectangular pixels that represent a 2D image such as a bitmap, voxels are discrete blocks and as such voxel models are 'stepped' in nature [83]. This analogy is illustrated in Figure 3-7.

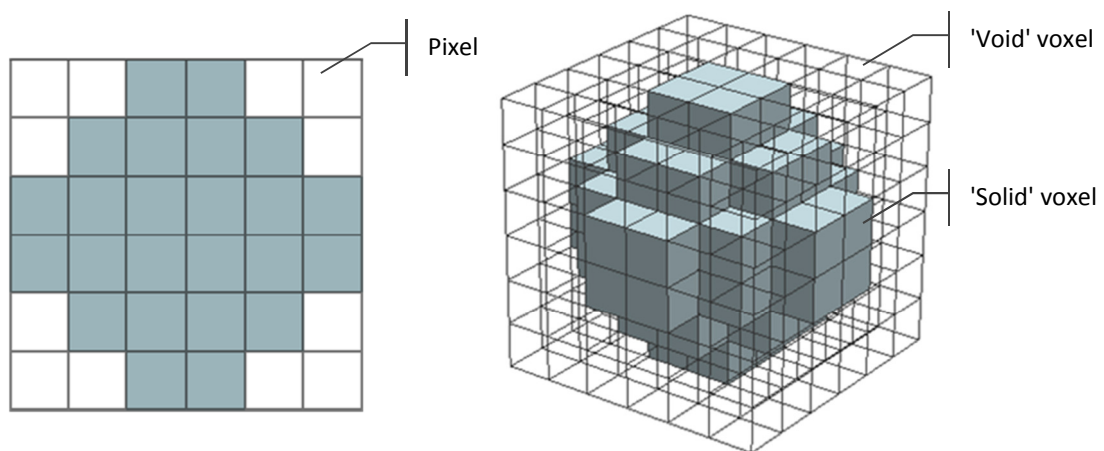


Figure 3-7: Low resolution pixel image of a circle and voxel model of a sphere

A voxel model is essentially a 3D matrix, with each element of the matrix a voxel. At its most simple, these voxels can have a value of one or zero, with one indicating the solid model and zero indicating void space [121]. Because a voxel model represents shape through volume rather than through a series of boundaries, there is no associated risk of generating manifold errors or invalid geometries. Anything created in a voxel model is valid for manufacture, as any physical thing that exists in the real world is composed of volumes rather than boundaries.

A key advantage of voxel models which set them apart from boundary representation is the ease at which additional information can be represented across a solid [120,122,123]. Density variation can be represented by assigning voxels through a range of values [1,124], rather than just values of zero and one representing void and a homogenous solid. However, this level of detail requires significant computer storage [124]. For this reason, voxel modelling has only become a feasible proposition over the last decade, with the developments in computer performance [125].

There are two methods to visualise voxel models. The voxels that comprise a model can be directly rendered or may be converted to a surface model and rendered conventionally. One method to do this is to construct an isosurface - discussed in more detail in the following section.

### 3.2.1 Voxel Methods for Structure Generation

Hollister *et al.* essentially used a voxel method to generate tissue scaffolds for reconstructive surgery [126], an example shown in Figure 3-8. The method generates lattice structures that are trimmed to fit a region of damage highlighted by an MRI or CT scan. MRI and CT scans are methods used in medical applications to acquire 3D models of patients' soft tissue [125], although MRI machines also have industrial applications. The output from both medical scanners are a series of 2D bitmap slices: images that represent successive cross-sections of the patient [126]. When stacked on top of each other, these 2D images (with a thickness corresponding to the distance between scans) can be considered a voxel model.

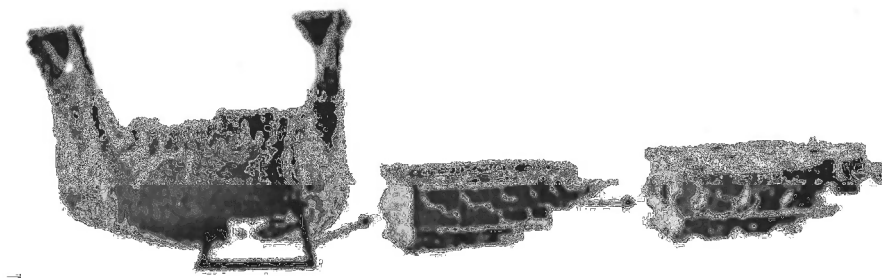
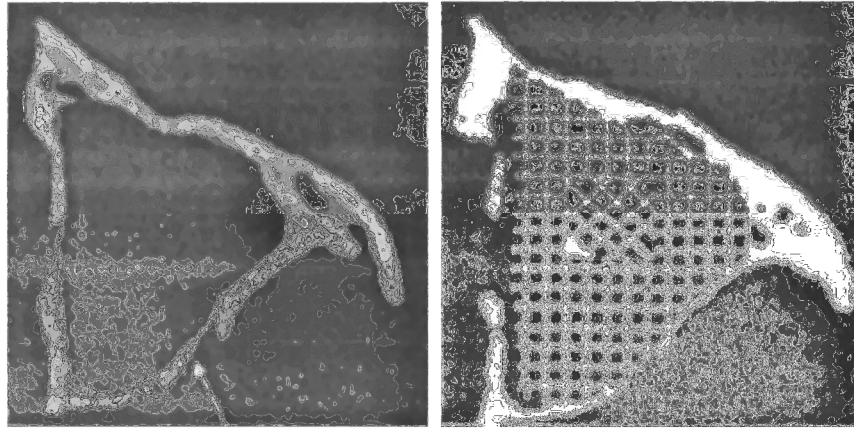


Figure 3-8: Tissue scaffold modelled with voxels [126]

The method developed by Hollister *et al.* isolates a region within the voxel volume as a shape to be tessellated with a tissue scaffold design. A simple voxel model of a lattice structure is generated using mathematical formulas to construct and combine voxelised primitives (such as spheres, cuboids and cylinders) [126]. The structure is tessellated orthogonally to dimensions that would fully enclose the isolated region. The voxels that comprise the void space of both the region and structure have a value of zero, whereas the structure voxels that represent solid have a value of one. The structure

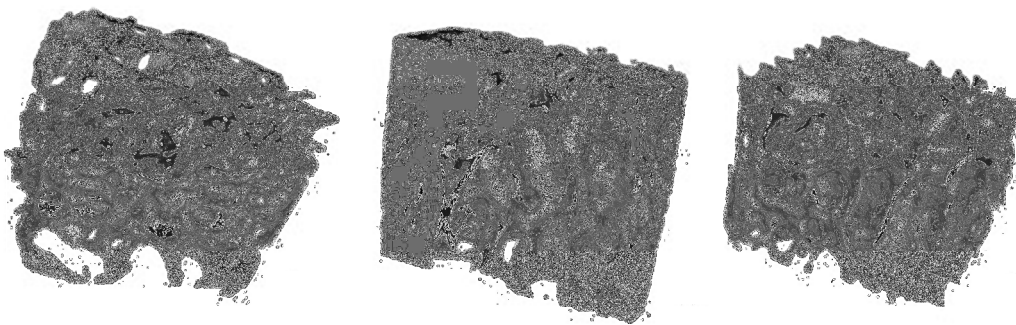


voxels are overlaid onto the region voxels, slice by slice. These voxel slices are multiplied together, element by element. Any structure voxels that are multiplied with the region slice void voxels also become void voxels, as any value multiplied by zero becomes zero. This effectively trims the structure voxels to the region, an example shown in Figure 3-9.



*Figure 3-9: Filling a slice of voxels with structure [126]*

An advantage of voxel models is the ease at which shape complexity can be represented, an interesting application of this being the ability to represent natural structures such as bone or foams. It is a straightforward process to extract the relevant tissue from an MRI or CT scan, the bitmap slices easily converted to voxels. This has been achieved by several groups for the purpose of analysis [127-129]. The capture of this geometry is a potential starting point for the characterisation and optimisation of organic, freeform, additively manufactured lattice structures. Examples of these random porous structures are shown in Figure 3-10.



*Figure 3-10: Random porous structures generated from voxel models [128]*

A voxel model is compatible with the data flow of the conventional route, through several routes depending on the AM process. As discussed in Chapter 2, slice files can be categorised into two types: raster and vector formats. Raster formats are essentially a sequence of bitmap slices. A voxel model can quite easily be converted to a raster slice format in the reverse manner that MRI and CT

scans are converted to voxels [83,126]. Each raster slice will correspond to a one voxel thick layer of the voxel model; single layers of voxels become 2D images comprised of pixels [83]. In some ways, this conversion from voxel model to raster slices could be considered an advantage in its own right over the conventional route, as a voxel model of a part becomes a WYSIWYG ('what you see is what you get' [130]) representation. The stepping inherent in a voxel model is, in the vertical axis, equivalent to the stepping between layers associated with additive manufacturing processes [83]. Unlike the majority of AM processes (i.e. those that utilise vector-based slice files, such as laser sintering), voxel models are stepped in all three axes, so this claim is not applicable to all AM processes, as demonstrated in Figure 3-11.

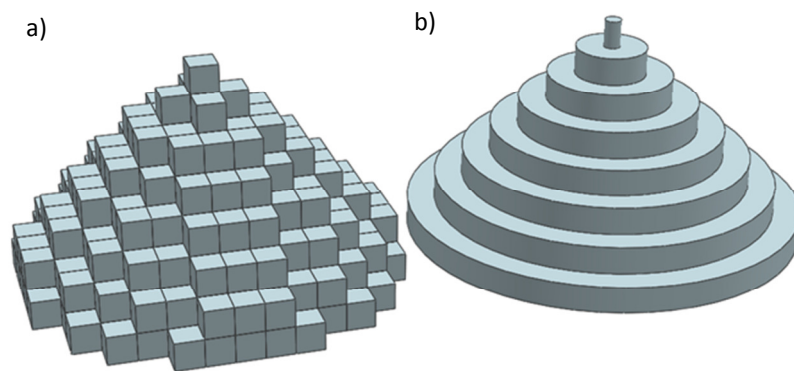


Figure 3-11: a) Voxel model - stepping in three axes. b) approximation of a laser sintered model - stepping in one axis.

Laser sintering - like most modern AM processes - require slice files in a vector format, so the approach described for converting to raster slices is not applicable. A voxel model must be converted into a format that is compatible with the conventional route for laser sintering, and the approach employed by Hollister *et al.* is an established route to convert a voxel model to a surface model: isosurfaces [126,127].

An isosurface is a boundary representation method that generates surfaces comprised of triangular polygons from a voxel volume [127,131], an example shown in Figure 3-12. Isosurfaces are commonly used to convert from voxel models to allow for straightforward visualisation and manipulation in more common CAD software. One such method of generating an isosurface is by using the 'marching cubes' algorithm [131].

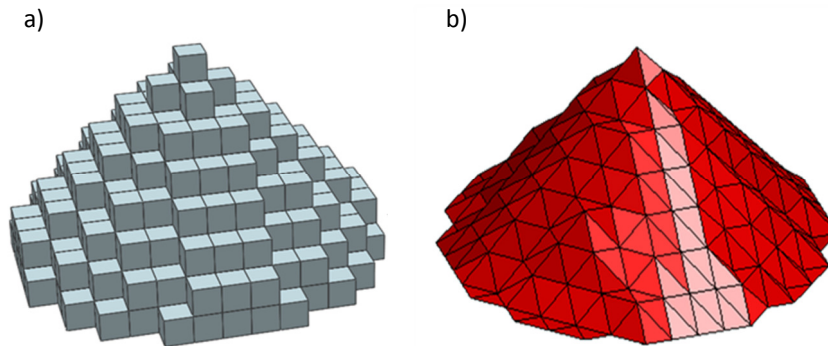


Figure 3-12: a) A voxel model and b) the isosurface constructed from it

The voxel model is interrogated in eight-voxel groups at a time. The group consists of eight neighbouring voxels arranged in a larger cube, as shown with two examples in Figure 3-13 (a). An isosurface is generated between a threshold - in this simple case where the voxel is in a binary state of zero (void) or one (solid), the threshold is set between the two values. The eight voxels within a voxel group are connected to form a cube. If an edge of that cube lies between one voxel that is below the threshold and one that is above, a node for one of the triangular polygons that compose the isosurface is placed at its midpoint (b). These nodes are then connected to form triangular polygons (c) [131].

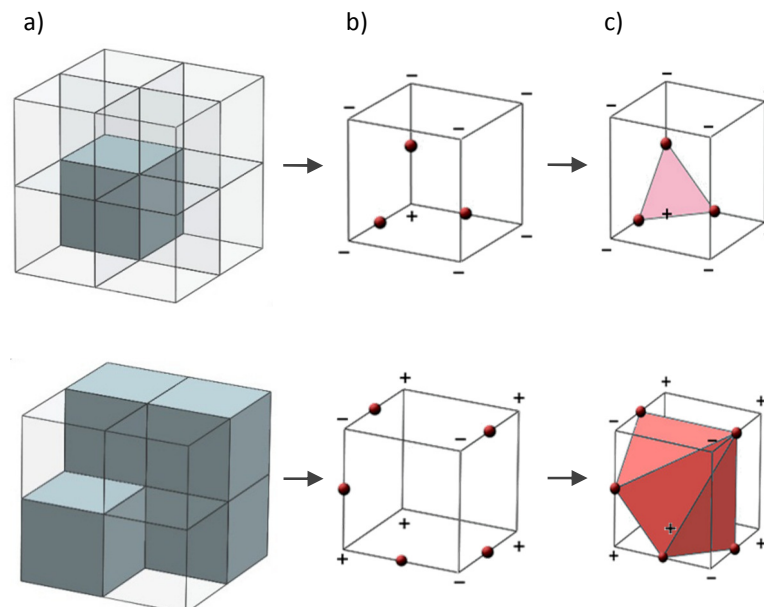


Figure 3-13: a) Eight voxel group, b) midpoint selection and c) midpoints connected to form triangular faces

Because an isosurface is composed of triangular polygons, it is a simple operation to reformat the information into an STL file format. This provides an alternative means for a voxel model to be

inserted into the conventional route for additive manufacture, and one compatible with laser sintering.

This is a far from satisfactory route, however, due to the size of the STL files generated. To avoid obvious stepping on a voxel model, the resolution must be high enough to exceed the resolution of the AM process. Because each polygon of the isosurface is generated on the scale of the eight-voxel groups, the polygons will be on the same scale as the voxels. Where a standard triangulation from a B-rep CAD surface to STL would construct few polygons whose size is dependent on the curvature of the surfaces, an isosurface is always constructed of small polygons, dependant on the original voxel resolution. This is illustrated with the worst case example: triangulation of a flat surface, in Figure 3-14.

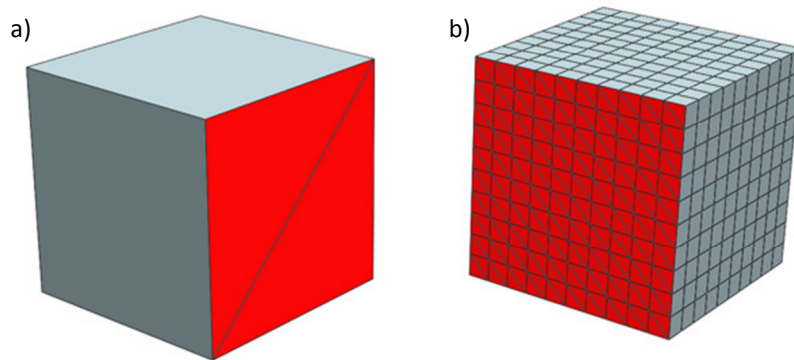


Figure 3-14: a) Triangulation of a flat face of a B-rep model (2 triangles), b) triangulation through isosurface construction of a voxel representation of a flat face (200 triangles)

This inefficient means of boundary representation ensures that any large voxel model converted to an STL will inevitably require significant computational resources to manipulate. However, in STL manipulation software such as Magics, functions exist for triangle count reduction.

### 3.2.2 Suitability of Voxels for Structure Generation

#### *Shape Complexity*

Voxel modelling shows great potential for the generation of large, complex lattice structures, due to the ability to efficiently generate complex geometry. Because voxel models represent volume rather than surfaces, a highly complex geometry requires the same memory as a completely empty space of the same size. A voxel volume of a particular size is composed of a set number of voxels. Whether the voxels are solid or void in nature has a huge impact on the geometric complexity of the model, but no impact on the overall file size of it, as the memory requirements to store a solid voxel are the same as a void voxel. This essentially means a voxel model's file size is independent of geometric

complexity [83]. The issue with this memory structure, however, is that for even a simple or empty model, the memory requirements are considerable [83]. However, as the memory capabilities of modern computers improve, voxel modelling has become more viable [132].

Both the accuracy of a model and its memory requirements are tied to the model's resolution - the size of the voxels relative to the model. No matter how high the resolution, the voxel model will always represent angled and curved surfaces with a stepped boundary: a result of the discrete block nature of voxels. A low resolution voxel model will require less memory, but the stepping will be on a relatively larger scale. However, methods have been developed that reduce the memory requirements of a voxel model. 'Octree decomposition', is such a method, where regions of a model of the same value are grouped into single, larger voxels [133]. For clarity, the 2D version (quadtree decomposition) is shown in Figure 3-15. As with PCs, however, AM processes are always improving: as surface quality of AM produced parts improves, the voxel resolution of models will have to increase to compensate, further driving up memory requirements.

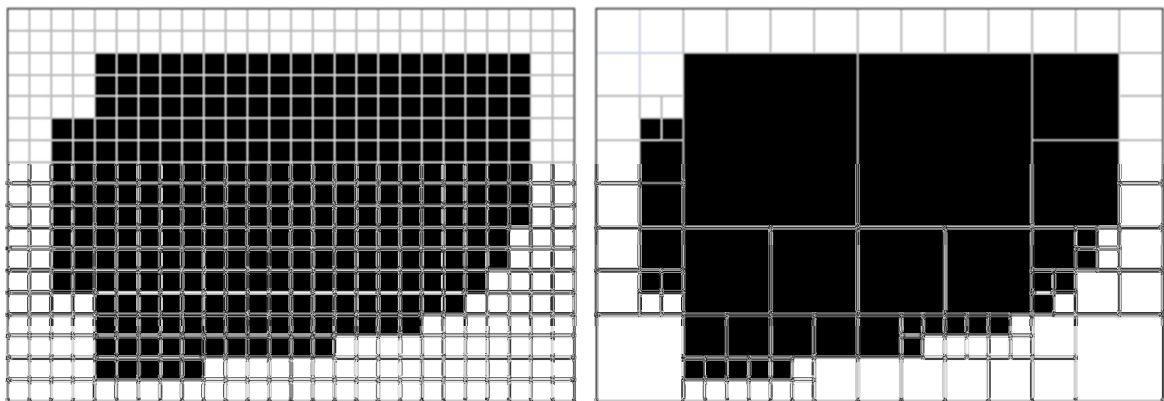


Figure 3-15: 2D quadtree decomposition

Voxel models allow shape, material and hierarchical complexity that conventional CAD cannot achieve. For example, voxel modelling allows for surface textures to be easily defined, using algorithms analogous to image processing techniques (like adding noise) that can generate texture like fur or stone (as shown in Figure 3-16). Each voxel can also contain additional information such as material properties [122,134].

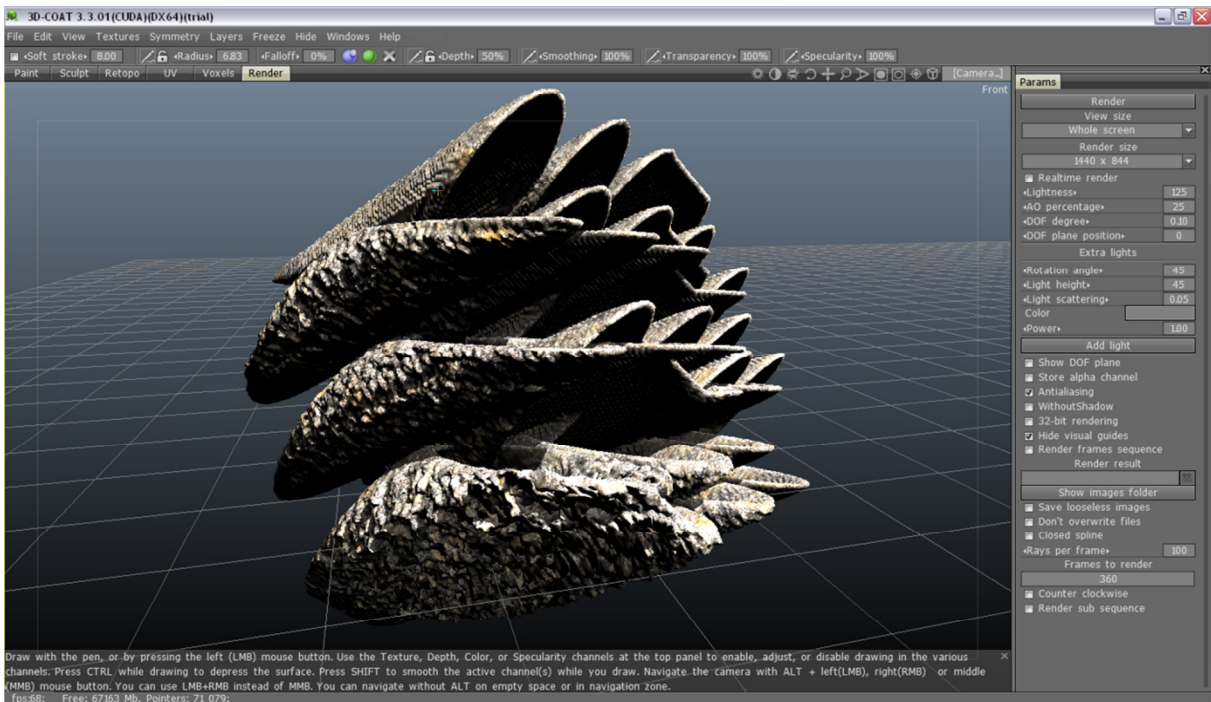


Figure 3-16: 3D-Coat voxel modeller: capable of complex and organic geometry, including surface texture [135]

Additionally, because it's a volume representation rather than a boundary representation, any voxel model generated is of a valid geometry. There is no possibility of manifold errors - gaps in surfaces or overlapping boundaries - which could prevent a model from being fabricated. Volume models are more appropriate representations of real-world parts, which makes a voxel-based approach more robust in terms of constructing manufactureable models.

However, there is no topological information stored in a voxel model that gives meaning to its features. For example, conventional CAD methods can define individual struts of a lattice structure, along with topological constraints that ensure correct connectivity with neighbouring structural elements. A voxel model cannot provide this connectivity - a voxel is known by its location, not its purpose.

### Useability

As well as the equation-based construction of geometric primitives that Hollister *et al.* demonstrated [126], commercial voxel modellers exist that supply a familiar, standard CAD-like interface for the construction of voxel models. 3D Coat is such an example, its user interface shown in Figure 3-16. However, conventional CAD is arguably better at controlling part dimensions: many voxel modellers are content with providing tools for artists rather than engineers and as such don't focus on accurate placement of features [135,136]. Also, if a feature happens to need to be placed between two

discrete voxels, it is not possible to represent without either locally or globally increasing voxel resolution. Alternatively, any geometry that can be constructed in CAD can be easily converted to voxels through a process called voxelisation. In the same way that it is trivial to evaluate a mathematical function to generate discrete points that can be plotted, voxelisation is a relatively straightforward process that samples a 3D model to generate discrete voxels [102]. The reverse is significantly more arduous, much like taking a set of points and attempting to fit a function to them. Trying to convert a voxel model to a boundary representation (as discussed with isosurfaces) is not a particularly efficient process.

With the ability to import CAD models, utilising any combination of these geometry construction methods allows a freedom to create lattice structures of any design in a voxel environment. However, the topological information of a model is lost in the voxelisation process. When decomposed into a series of discrete voxels, all connectivity information that defined a surface is discarded. This is similar to the effect when converting a CAD model to an STL, as described in the previous chapter. Similarly, it is difficult to assign specific dimensional information to regions of a voxel model when built from scratch. Where in a B-rep model it is known that a specific face of a model is, for example, cylindrical, a diameter can be assigned to parameterise it. The voxels that approximate a cylindrical face of a voxel model do not contain this information. They cannot be automatically grouped as such and thus this cylindrical face cannot be modified in a meaningful way.

From a technical standpoint, voxel models can take full advantage of modern, multiple-processor computer [132]. Due to the discrete nature of voxel models, the model can be split and manipulated in parallel between multiple processors. For a conventional CAD model this is more difficult, as a particular surface can have implications across the whole model. However, even with modern computers, the size of a high-resolution voxel model can still make manipulation a slow process.

### *Data flow*

A notable disadvantage from modelling with voxels is that it is not part of the conventional route of information flow for the majority of AM processes. Isosurfaces have been shown as a possible but unsatisfactory conversion route due to memory requirements. The raster-based slice files that can be easily generated from a voxel model are incompatible with most AM processes that require vector-based slice files. The advantages gained from voxel modelling are nullified if the geometric information cannot be efficiently transferred to an AM machine for manufacture. If an efficient

manner can be developed, then voxel modelling has considerable potential for the design of lattice structures.

### 3.3 Function Representation

Function representation (F-rep) defines geometric objects implicitly, that is, with functions that determine whether a point is inside, outside or on the surface of the object [137]. F-rep can be considered an evolution of CSG. Whereas CSG is only capable of determining if a point is inside or outside of a solid, F-rep is capable of defining heterogeneous solids [1,138]. Like CSG, F-rep can construct geometry with Booleans and primitives, but has been expanded to allow operations to further modify geometry such as parametric models, skeleton-based models and complex sweeps [1,137,139]. 'HyperFun' is a freely available programming language that models parts through function representation [140].

An F-rep model only exists as a set of highly computationally efficient functions during modelling, allowing the construction of complex geometry [1,141]. Consider the rectangular 'slabs' shown in Figure 3-17. Each slab is represented by a single function that defines its size and position, rather than the multiple lists of vertex co-ordinates, edge and face connectivity required by boundary representation.

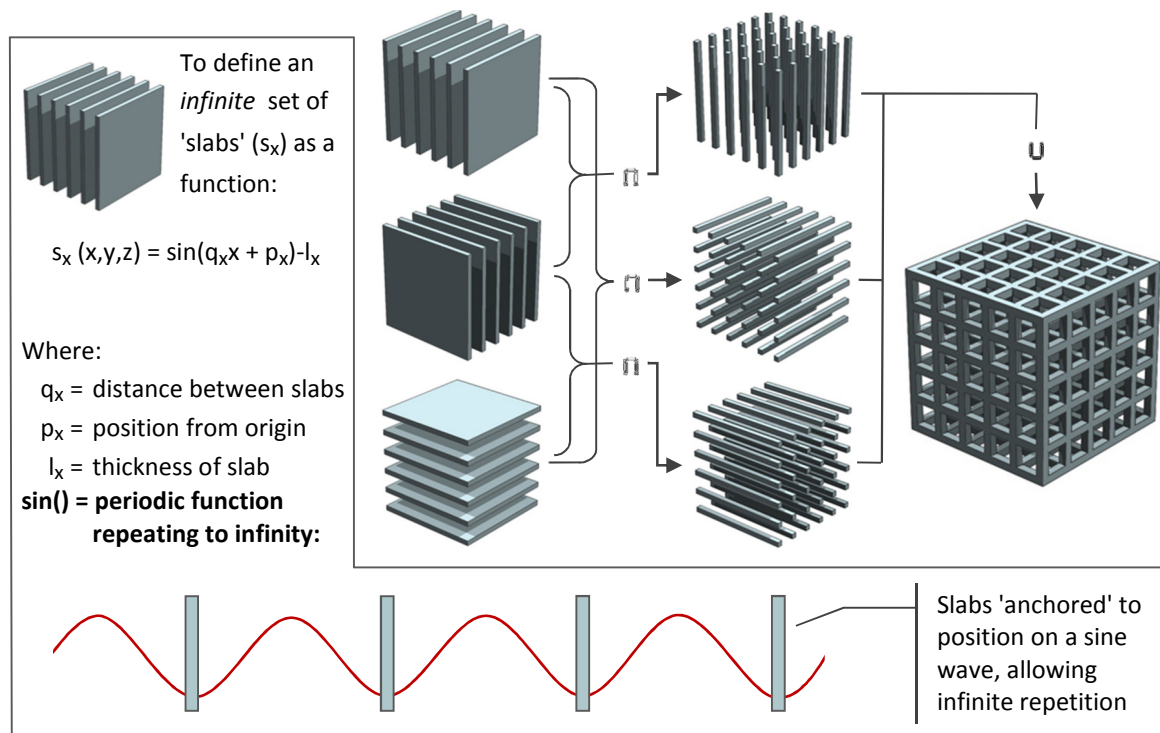


Figure 3-17: The implicit functions and the Booleans to generate the cubic lattice structure



These slabs are, by nature of the implicit functions that represent them, repeated into infinity (as shown in Figure 3-17), which facilitates the representation of lattice structures that are also infinite in size [60]. These slabs are combined with Boolean operations to form the basic structure and finally trimmed to a boundary shape of finite size. To vary the distance between struts, for example, the frequency value of the structure's function need only be changed [60]. This compares favourably with B-rep modelling, where to generate a structure with more struts, each new strut would need to be duplicated and repositioned. Examples of an F-rep structure are shown in Figure 3-18, the only difference between the lattice structure's F-rep description are the frequency values of the function.



Figure 3-18: Modifying structure geometry through manipulation of a function [60]

As well as functions that define primitives, additional functions have been developed that augment or warp geometry. Examples of warping are shown in Figure 3-19: a) the struts of the structure increase in thickness linearly along one axis; b) the distance between struts is dependent on the distance to the external surface; c) a pseudo-random structure is generated by adding a noise function to a regular tessellation [142,143]. A blending function has also been applied to example b) in Figure 3-19 to generate a smoother, more organic structure[137]. Like voxel modelling, F-rep is also capable of using noise functions to define surface textures, which further expands the capabilities of this method [138].

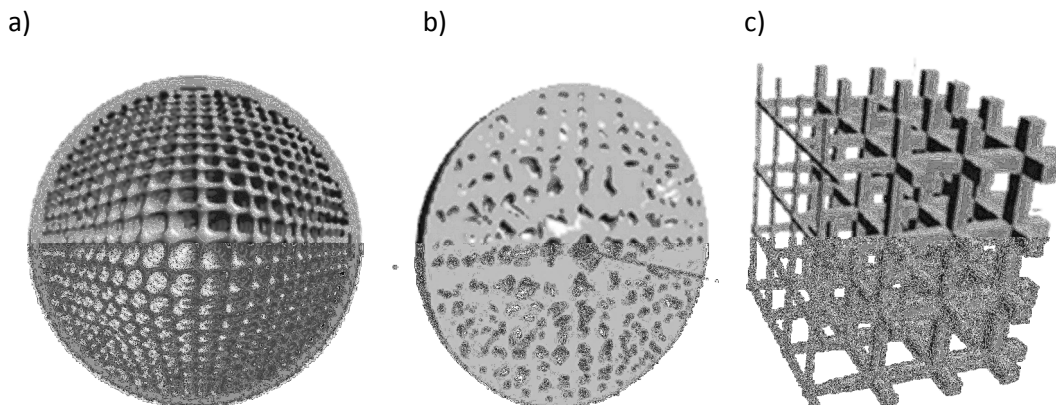


Figure 3-19: F-rep structure dimensions grading as a function of a) distance along x-axis, b) distance from surface of sphere and c) random noise [60]

A key development in F-rep modelling is the method in which models are visualised, which was until recently a major disadvantage of the method [2]. Visualisation is crucial in allowing real-time feedback during the design process - ensuring confidence in a design - especially as complexity increases. In conventional visualisation of geometry, an explicitly defined surface is required for a rendering engine to produce a shaded model. The most basic rendering techniques (e.g., Gourard and Phong shading) polygonise a surface and shade each polygon depending on its angle relative to a light source [133]. This is readily compatible with B-rep models, as they are already polygon surfaces, but an extra conversion step is required when using these methods to render F-rep models.

An F-rep model can be converted to a B-rep model for visualisation [144], but the advantage of modelling in F-rep over B-rep (the ability to efficiently represent complex structures) is lost if this is necessary. This was a necessary conversion that limited F-rep until a method was devised by Fryazinov and Pasko to directly ray trace a model at real-time rates on modern PCs, without the need to generate explicit surfaces [60,139]. Ray tracing is a more advanced rendering method that projects virtual light beams that - when reflected from a surface - bounce back with an intensity value that determines the model's shading [145]. Fryazinov and Pasko have utilised a computer's graphics card to accelerate the process, to allow for interactive viewing and modification of an F-rep model of a greater complexity than possible with standard rendering of conventional B-rep models [139].

One inherent issue with function representation, however, is that designs aren't modelled directly. To model a part the functions that best fit it must be determined. In some cases it is difficult to model exactly what is required, although in these cases where a part cannot be represented with functions, it can be used as a starting point and then modified with any other modelling method as required.

### 3.4 Conclusions

Several alternate methods of geometry creation have been discussed in this chapter. The various STL 2.0 formats - while showing significant advantages over the standard STL format - don't show particular promise in the design of lattice structures. The AMF format contains a function for the storage of structure design information but no conformal structure is explicitly created. Function representation is a promising method and has been shown to efficiently generate large arrays of particular lattice structures.

Voxel models have the potential to allow the design of geometrically complex lattice structures. Structure design is limited by overall part size rather than geometric complexity. Being a volume

modelling technique rather than a boundary representation, any modelled geometry is valid for manufacture. Where AM processes require raster-based slice files, voxel models can be easily converted and for AM processes that require vector-based slice files, isosurfaces may be used to convert voxel models into STLs. This particular route however will result in particularly large files that may be difficult to handle.

The STL 2.0 formats have been specifically designed to replace STL and as such integrate into the conventional route of design to manufacture well. F-rep and voxel modelling are integrated into conventional route to some extent. It has been stated that F-rep can be easily converted into STL and slice formats [1] however the efficiency of these conversions is not clarified. Voxel models can be quite easily converted to the raster slice formats required by 3D printing, however for the majority of AM processes (that require vector formats) this conversion is not so straightforward.

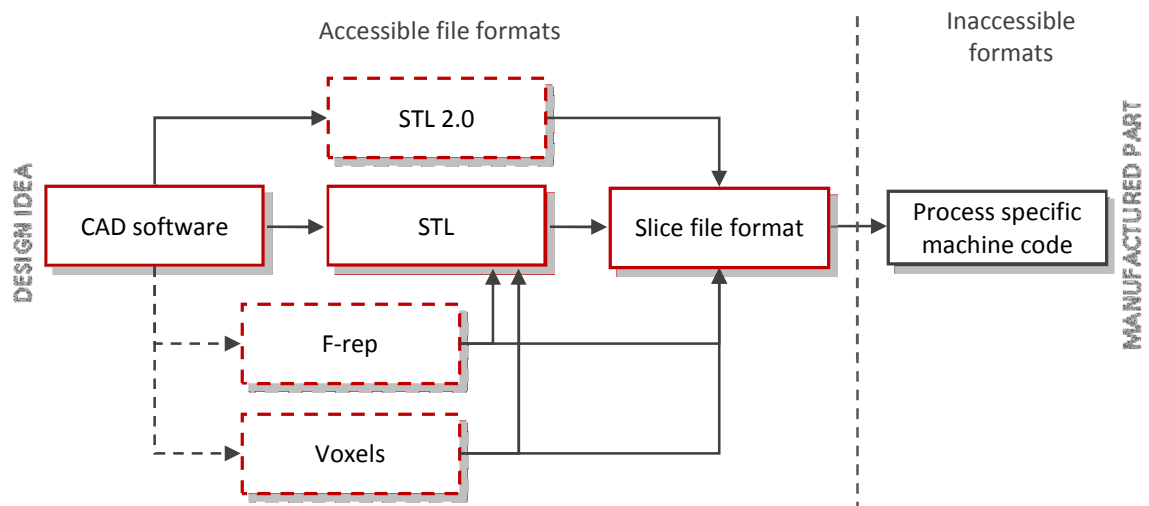


Figure 3-20: Integrating alternate methods into the conventional route. F-rep and voxel methods can generate geometry independently or import from CAD (hence dashed arrows), whereas STL 2.0 (as primarily a translation language) requires a CAD model

The incorporation of a modelling method into the conventional route is important due to drivers at both ends. At the beginning of the process (the design idea), users are generally comfortable and familiar with current design tools (such as CAD software). At end (manufacture of an AM part), the majority of AM machines have narrow allowances for what can be read. For any realistic uptake of a new structure generation method, it should be able to integrate between these two fixed points. In summary, voxels and function representation have potential for the generation of conformal, complex lattice structures but must be formatted for or incorporated into conventional route in a way that satisfies both users and existing infrastructure.

The following chapter discusses methods for conforming a structure to a shape.

## 4 | The Design of Conformal Lattice Structures

The previous chapters discussed applications of geometry representation methods in the design of lattice structures. In terms of designing conformal lattice structures, there are a range of approaches. The purpose of this chapter is to review these methods. A number of groups have investigated methods to generate large, complex lattice structures and are also noted. How this geometric complexity is achieved is the topic of the following two chapters. This chapter is solely concerned with presenting general methods of conforming a structure to a shape.

### 4.1 Overview of General Lattice Structure Design

Throughout this thesis, the topology of a lattice structure is categorised as follows. A lattice structure can be considered as a hierarchy of different structural elements - depending on the situation it may be useful to consider the structure at different levels of this hierarchy. At the lowest level, a structure is comprised as a series of 'struts' (labelled in Figure 4-1). While the design of the struts may be identical throughout the structure, they will be oriented as such that they can join to form a lattice. The next level of the hierarchy is a 'cell'. The cell is this arrangement of struts and defines their connectivity.

The term 'structure' is used to describe the entire lattice of struts or cells. Because the shape of a cell can vary between different structure types the generic term 'cell diameter' is used in this work as a measure of the size of a cell. The 'conformal shape' is the boundary to which a conformal lattice structure must fit. A conformal structure may also have a 'skin', a solid sheet of material across all or a portion of boundary of the conformal shape that can serve a number of roles, from aesthetics to reinforcement.

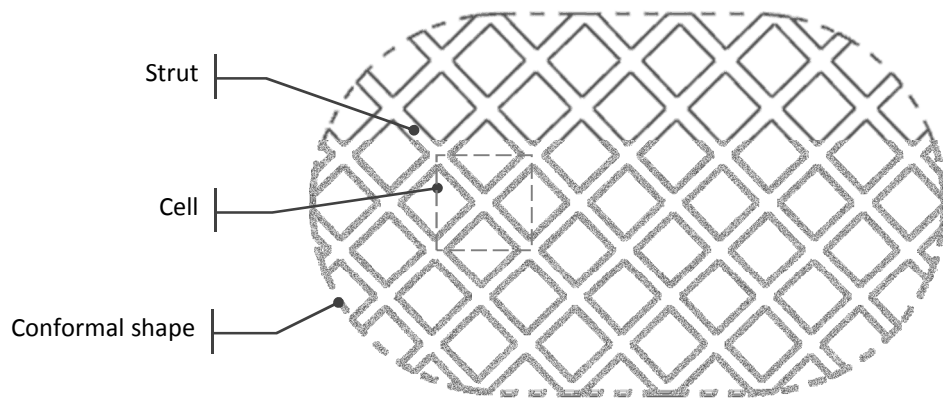


Figure 4-1: Lattice structure naming convention

As discussed previously in Section 1.5, it can be assumed that both the strut geometry and cell geometry will affect the properties of a lattice structure. For example, regardless of the cell type, increasing the thickness of the struts will tend to stiffen a structure. Similarly, reducing a cell's diameter while retaining the strut thickness will also stiffen a structure. Generally, the higher the ratio of cell diameter to strut thickness, the stiffer a structure will be. It can be assumed that any modification to the size or shape of a cell will modify its properties. This simple assumption is an important consideration to note in the following section that details the methods by which a structure can be manipulated to conform to a shape.

## 4.2 Methods to Generate Conformal Lattice Structures

### 4.2.1 Trimmed Structures

The majority of conformal structure methods reviewed trim a regular tessellation of a cell to the required shape. This method of conforming structures to a shape is often employed because it is a relatively straightforward method to implement that in general consists of two steps. First, a structural cell is arrayed to encompass a shape and then second, regions of the structure that are determined to be outside of the required shape are removed. A simple 2D example of a trimmed structure is shown in Figure 4-2.

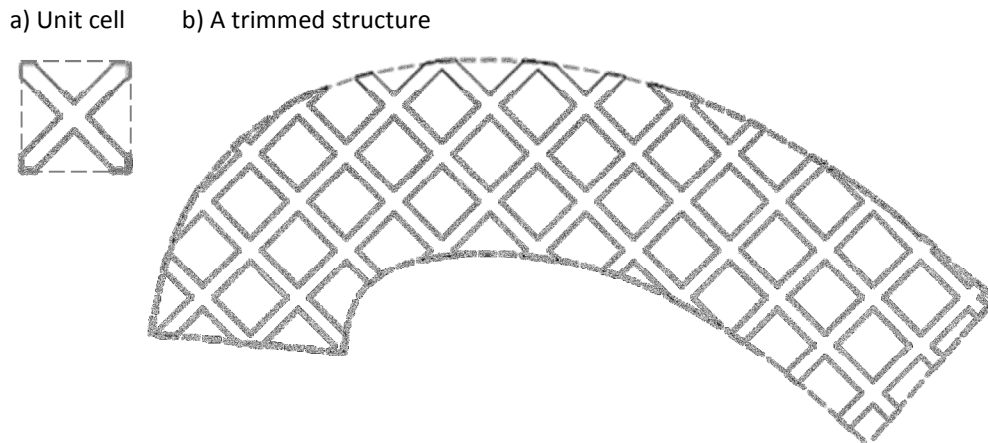


Figure 4-2: Trimming a structure to fit a shape

Medical applications are common between many attempts at generating trimmed structures. For example *in vivo* tissue scaffolds have been designed to promote new cell growth, as shown in Figure 4-3 [55-57]. For these applications the porosity of the structure is the primary consideration (to permit the movement and growth of biological cells or for modelling purposes) rather than strength or rigidity [54]. Several software packages also generate trimmed conformal structures, such as Materialise Magics, Selective Space Structures from netfabb and Marcam Engineering's AutoFab [92,93,107]. Magics and Autofab generate structures at the STL stage, whereas Selective Space Structures uses a vector-based slice format, as discussed in Chapter 2.

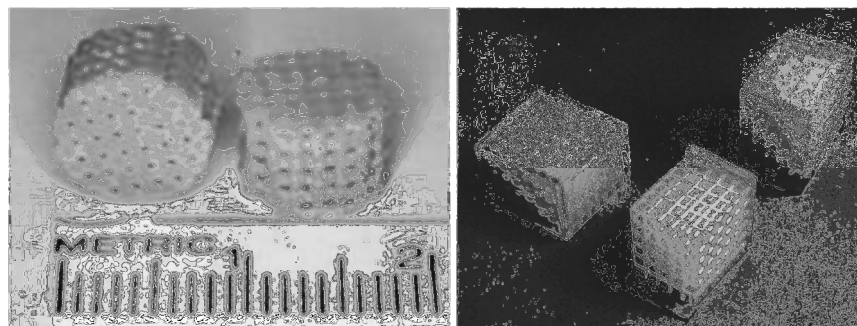


Figure 4-3: Trimmed structures investigated for medical applications [79][80]

Certain groups have taken the trimmed method further by implementing techniques that vary geometry across the structure, commonly termed 'functional grading' [55][82]. Wettergreen *et al.* designed a series of different cell geometries that incorporate a common mating point [55]. The structure was designed to resemble a human vertebrae. To functionally grade the structure, finite element analysis was conducted on a model of the vertebrae. This resulted in a 'modulus map' used as a key for the selection and location of suitable building blocks.

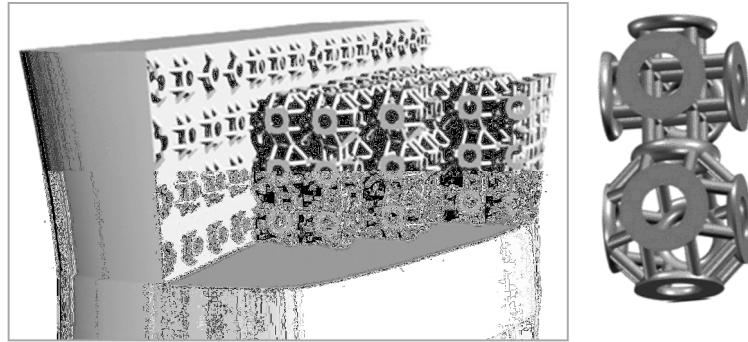


Figure 4-4: Trimmed structure generation by Wettergreen et al. [55]

A modulus map is a two-dimension graphical representation of a particular mechanical property, such as stress [146]. Different colours of a modulus map correspond to different levels of stress that a region of the part is enduring, which was used to determine an appropriate structure for that region [55].

A similar method of generating functionally graded structures was investigated by Gervasi and Stahl [82]. An optimisation procedure following the Solid Isotropic Material with Penalisation method (SIMP) was implemented on a two-dimensional design domain with initial conditions that described loadings and constraints, as shown in Figure 4-5. After several iterations, the SIMP method generated a density map, a two-dimensional grid of different density values. This grid was used as a basis to map a simple structure that was manipulated to match the required density at each point.

The method is currently only capable of generating single-layer structures; the structure was arbitrarily repeated four times to thicken the part. Unlike the other trimmed structure methods discussed in this section, this method does not actually trim the structure to the input conformal shape, it approximates the boundary to the nearest whole cell.

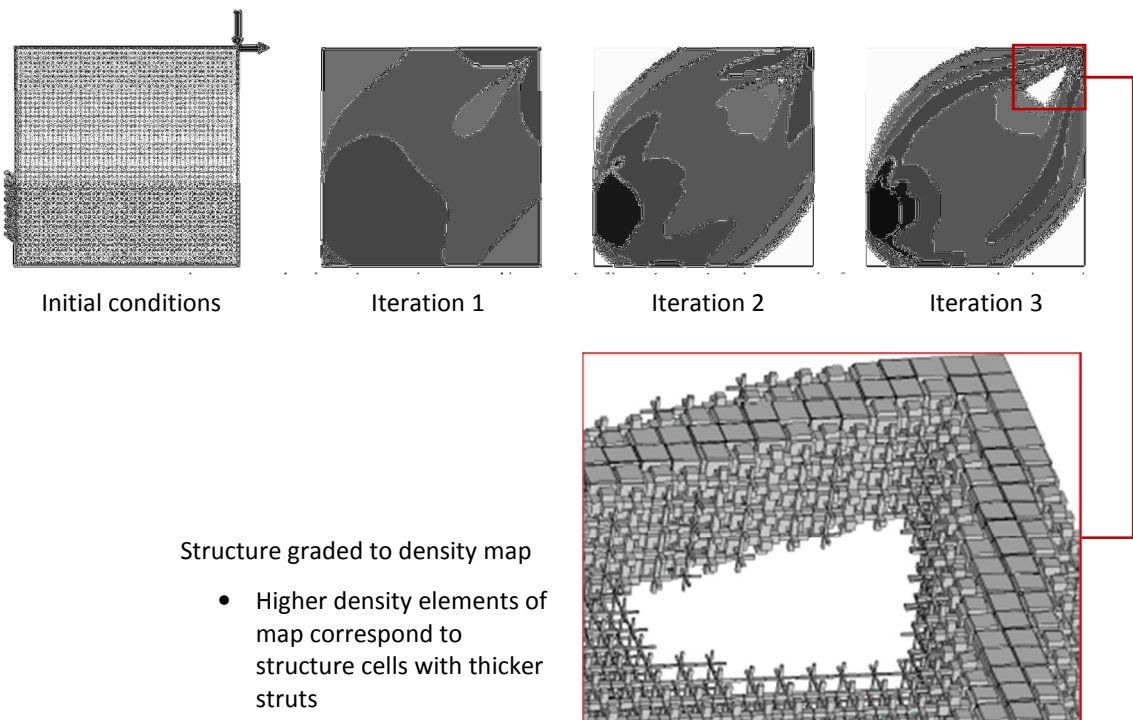


Figure 4-5: Topology optimisation - selection of structure through a density map [82]

The main advantage of the trimming method of conformal structure generation is that it is robust. An outcome can be generated given any unit cell and any input conformal shape. This is a significant advantage as the method is not limited to particular forms of a conformal shape. Whether or not the outcome is useful is down to the choice of cell and conformal shape rather than the method. For example, trimming a structure to a thin-walled conformal shape is liable to separate a structure into multiple parts that will be unconnected when manufactured. In general, a conformal shape with geometry that is thinner than the diameter of the cell risks this issue, as shown in Figure 4-6.

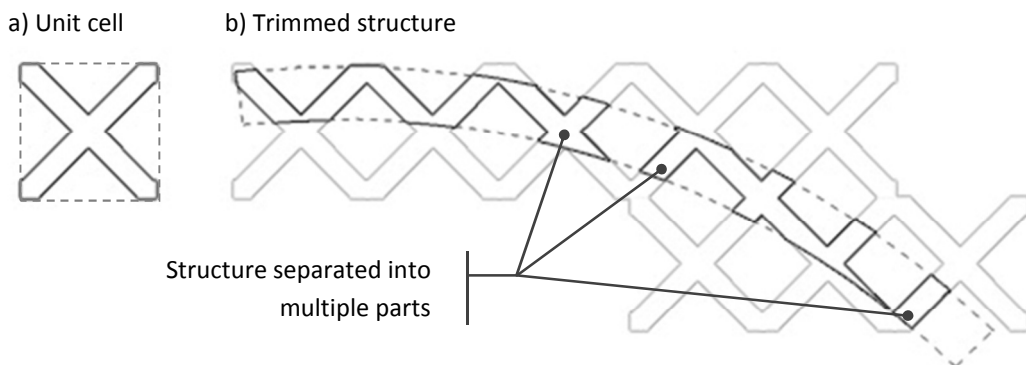
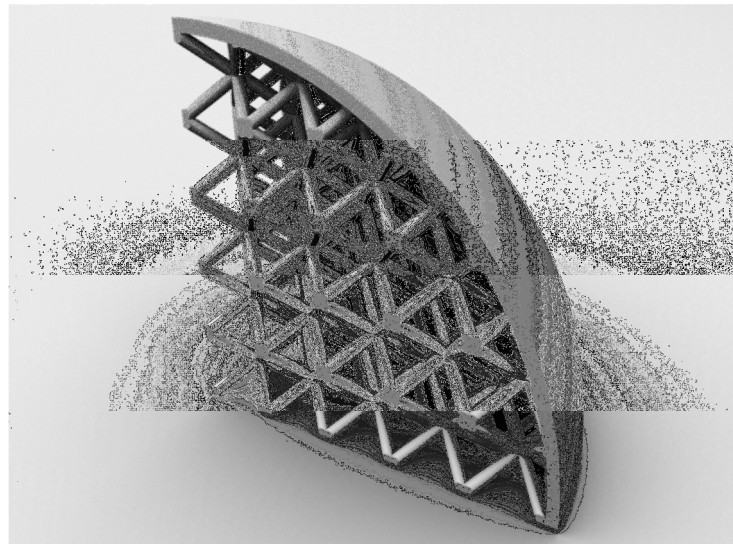


Figure 4-6: Trimming a structure to a thin-walled conformal shape



A disadvantage of this conformal method is that trimming a structure inevitably weakens it at the boundary. The strength of any structure is dependent on the interconnections between neighbouring struts. At the boundary the connectivity between cut struts is lost when regions of the structure outside of the conformal shape are removed. Structure trimming software such as Magics and AutoFab provides the option to add a skin to the structure in an attempt to address this issue. An example of a skin is shown in Figure 4-7.



*Figure 4-7: A lattice structure partially covered by a skin*

Another disadvantage of this method is that there is usually an obvious structural orientation relative to the conformal shape. Fully isotropic behaviour is an unlikely property in any structure designed for the macro scale. Orthotropic behaviour (the same properties in x, y and z axes) are potentially obtainable in a cubically-arrayed lattice structure, although considering the anisotropy inherent in parts fabricated by additive manufacturing even this is not straightforward. For structure designs that maximise rigidity this is less of a concern as the structure can be designed in a way that even the weakest orientation is rigid enough. For compliant structure designs where a small range of low rigidity values is acceptable this could become a significant issue.

The trimmed structure method of generating conformal structures is capable of yielding a result for any conformal shape or cell design supplied. However, the orientation of a trimmed structure does not consider the curvature of the conformal shape's surfaces. Essentially a trimmed structure is capable of populating any volume with structure, but the quality of the boundary is not considered.

#### 4.2.2 Swept Structures

The swept structure method deforms a unit cell to fit a particular surface, as shown in Figure 4-8. When a structure is swept in this manner, it is always oriented in the same direction relative to the surface. This is in contrast to the trimming method of conforming a structure, where the surface of a conformal shape has no impact on the structure's orientation.

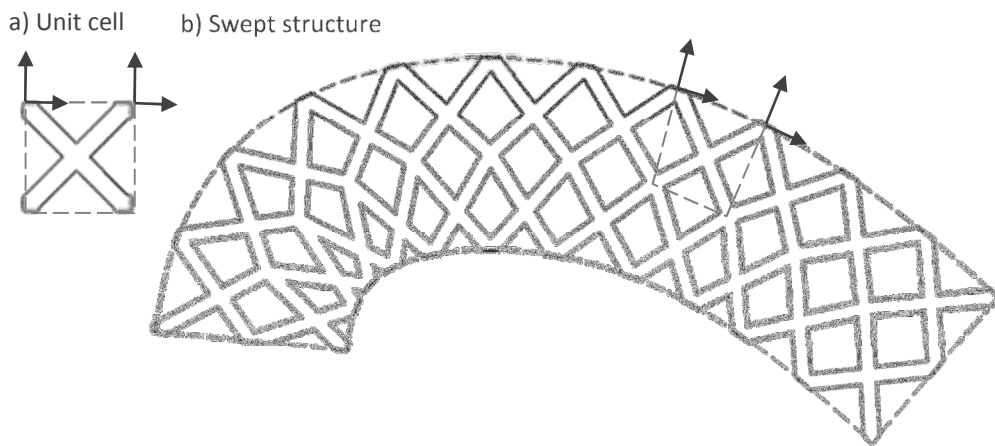


Figure 4-8: Sweeping a structure around a shape

Gibson *et al.* have extensively investigated sweeping structures to follow a surface [50,71,85,147]. Normal and tangent information is calculated for points on the surface, and a unit cell deformed to fit it.

Improved strength to weight ratio and stiffness are cited as applications for these particular conformal structures [50]. As such, the structure design most often employed is derived from a 3D tessellation of octahedra and tetrahedra (called the 'octet truss structure') - polyhedra composed of triangular faces that exhibit high rigidity as shown in Figure 4-9a [50,147]. At the most basic, the method has been employed to map a few rows of structure to thin-walled sheet-like parts, such as the 'skin' concept also shown in Figure 4-9.

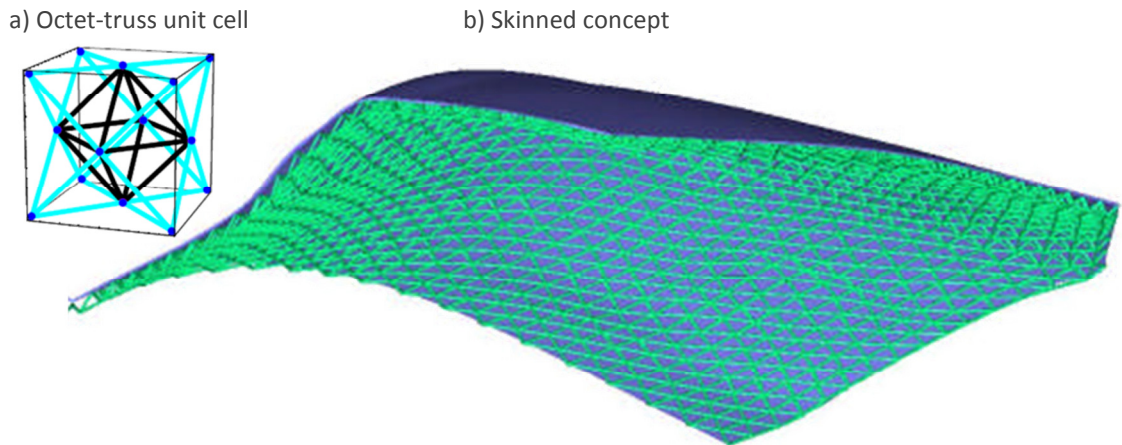


Figure 4-9: Swept structure mapped to thin-walled geometries [50,71]

The method has been developed to warp a structure between two different surfaces. Figure 4-10 shows a simplified 2D representation of a structure swept between two surfaces with different curvature. Wang and Rosen have developed a method which generates a number of intermediate surfaces between the original external surfaces. These intermediate surfaces are generated by linearly interpolating between discrete points on each of the original surfaces [85].

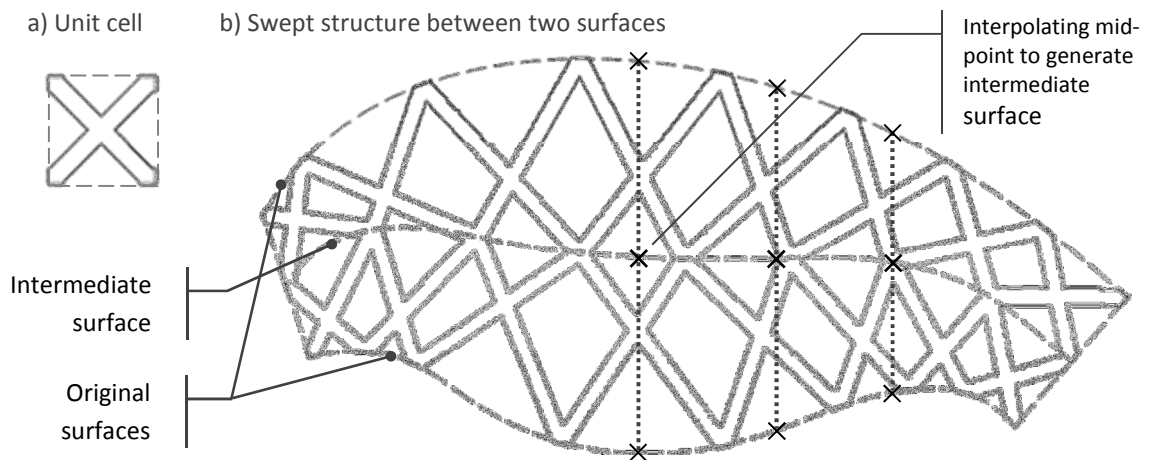


Figure 4-10: Sweeping a structure between two surfaces

As with the trimmed structure method, by varying strut geometry, a swept structure can be functionally graded. However, the sweeping itself adds a layer of complication to structure design. By deforming a cell to fit a surface, any properties specific to the original dimensions are changed. As discussed in Section 4.1, a simple way to control a structure's properties is to vary strut diameter or cell diameter. The diameter of swept cells can vary dramatically depending on the surfaces that the structure must conform to. The fundamental issue with the sweeping structure method is that the properties of the structure will inevitably become a function of the curvature of the surface.

From a technical standpoint, the system developed to generate a swept structure employed by Gibson *et al.* is complex. Sweeping a structure is a mathematically difficult task; for any point on the conformal surface, normal and tangent information must be calculated to construct a region to place a swept cell. For this reason, the chosen conformal shape is approximated by a series of Bézier surfaces [85]. A Bézier surface is one that is defined by a series of equations which are manipulated by a set of control points [148]. This allows a complex freeform surface to be approximated by a series of more simple Bézier equations, which simplifies the determination of normal and tangent information in the construction of structural cells.

Regardless of the technical system developed to generate swept structures, the actual conformal method is limited in terms of conformal shape complexity. As structure is deformed to fit a surface curvature, a surface can bend so tightly that it is impossible to fit the structure around it. An example is shown in Figure 4-11. Additionally, even with the modifications that allow sweeping between two different surfaces, the conformal shape must have clearly defined 'top' and 'bottom' surfaces to map structure between, also shown in Figure 4-11.

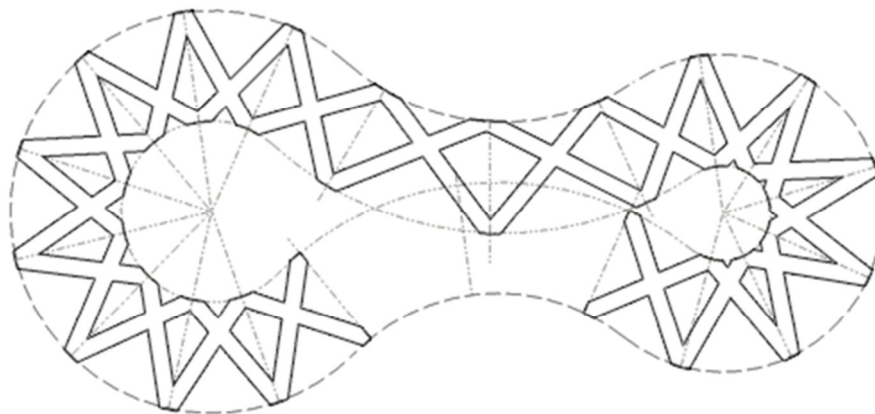


Figure 4-11: Difficulty in sweeping a structure around a shape with tight bends and no clear 'top' and 'bottom'

In general, the sweeping method is only suitable for conforming structures to certain geometries - with a high length to thickness ratio that generally follow a single surface. The swept method is severely limited in this respect.

#### 4.2.3 Meshed Structures

Finite element analysis (FEA) simulates the physical properties of an object and how it reacts under loads and constraints. FEA can be used to solve 2D and 3D problems and represents an otherwise complex model as a mesh of simpler discrete elements [148]. When meshing a 3D object, a volumetric mesh is usually generated - a mesh of 3D elements that discretises the whole volume

rather than just the surface. The field of FEA is well developed and by extension so too are the meshing algorithms used to discretise the object in question [148]. One relatively straightforward method of constructing a conformal structure is to take such a volumetric mesh and use it as a base to map structural geometry to.

An FE mesh is constructed at an early stage of finite element analysis [149]. It is an approximation of a part's geometry - a volumetric mesh is usually generated directly from a CAD model to certain accuracy and resolution conditions [149]. This volumetric mesh is comprised of many elements, as such a single complex problem is broken down into a series of simpler ones [150]. FEA can be conducted on 2D and 3D models, but the latter are of interest in the design of structures. Because the primary aim of FE mesh construction is to match a mesh to a particular shape, the basis for a conformal structure is generated without need for further modification.

There are a wide range of different volumetric meshing techniques that generate 3D polyhedral meshes for use in FEA, and some groups have investigated their use as the basis for conformal structures. A simple, 2D quadrilateral mesh is shown in Figure 4-12.

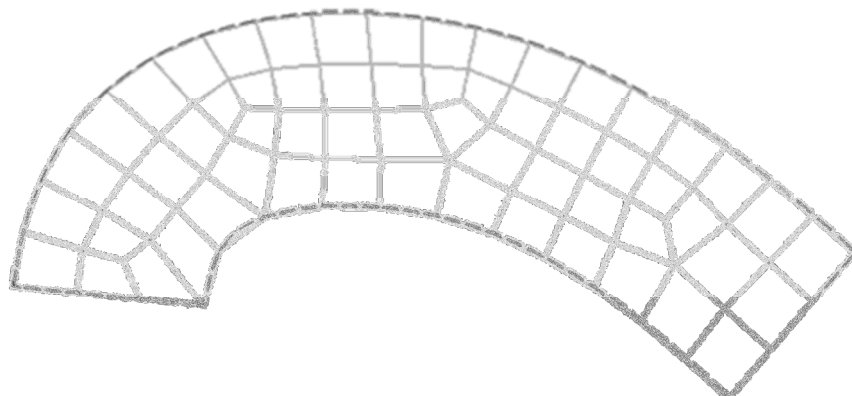


Figure 4-12: A finite element mesh

One method to generate a structure from a mesh is to extract the edges to use as a wireframe skeleton to map strut geometry to. Alternatively, a unit cell of structure can be warped to fit the elements of the mesh, as investigated by Stahl *et al* [82] (Figure 4-13). Although the distorted structure fits the conformal shape, the level of distortion of the structure raises into question the suitability of the method - the shape and size changes between distorted cells modifying its properties. However, it could be argued that the conformal shape shown in Figure 4-13 could have been meshed to a higher quality, in terms of minimising shape and size differences between elements.

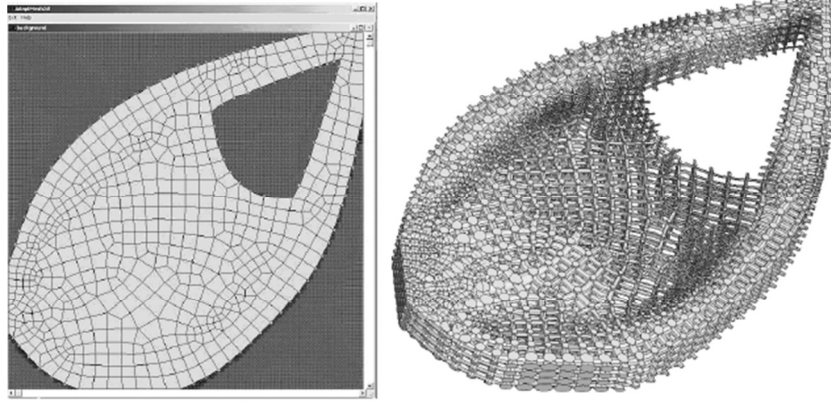


Figure 4-13: Mesh and structure [82]

The advantage of constructing conformal structures based on FE meshes is that it is a long established field and a number of software packages already exist that specialise in generating FE meshes [148,150]. This can then be used as a starting point - the basis to map structural geometry to. Like the swept method of generating conformal structures as discussed in the previous section, elements of an FE mesh are deformed to fit the conformal shape. Unlike the swept method, the FE meshes are much more capable in terms of shape complexity that can be meshed - they are not limited to the thin wall-like shapes that swept methods are. There are different types of meshes available for different purposes (such as four-sided tetrahedra, five-sided 'pentas' or six-sided 'hex' elements [151]) which would influence the properties of the types of structure designed. These examples are shown in Figure 4-14. These mesh types can even be combined, which has interesting implications for the generation of functionally graded conformal structures. However, only the tetrahedral mesh is particularly flexible in terms of shape complexity achievable [151].

A user has advanced control over the shape of an FE mesh, such as the level of deviation or deformation allowable between elements during the meshing process [151]. Despite this control, the final mesh will still exhibit a level of variation in element shape.

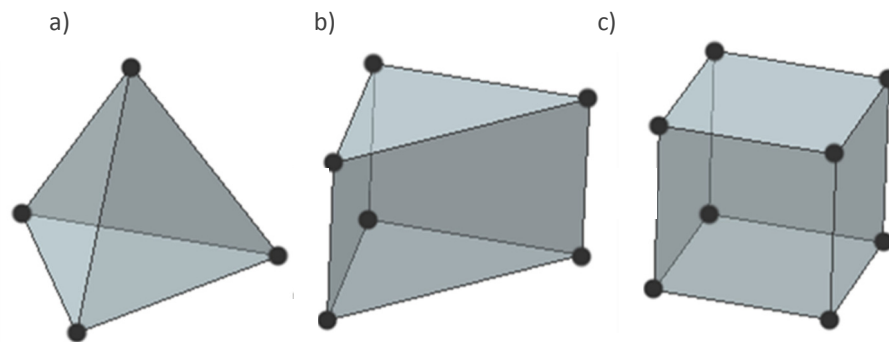


Figure 4-14: a) tet, b) penta and c) hex finite elements

#### 4.2.4 Voronoi Tessellations

Voronoi tessellations are techniques to subdivide space, commonly used to model the structure of materials and in computer graphics [152]. The method can be applied in two and three dimensions. A 2D Voronoi tessellation is the dual of a Delaunay triangulation - itself a method of constructing a triangular mesh from a set of points. A Voronoi tessellation is generated by constructing perpendicular bisectors from each edge of the Delaunay triangulation [97,153], as shown in Figure 4-15. Another way to consider the construction of a 2D Voronoi tessellation is that each vertex of the Delaunay tessellation acts as a seeding point for a cell, termed a 'Voronoi region' [154]. If a circular wave is imagined to expand at the same speed from each seeding point, straight boundaries are formed where waves from neighbouring points meet [154,155], also shown in Figure 4-15.

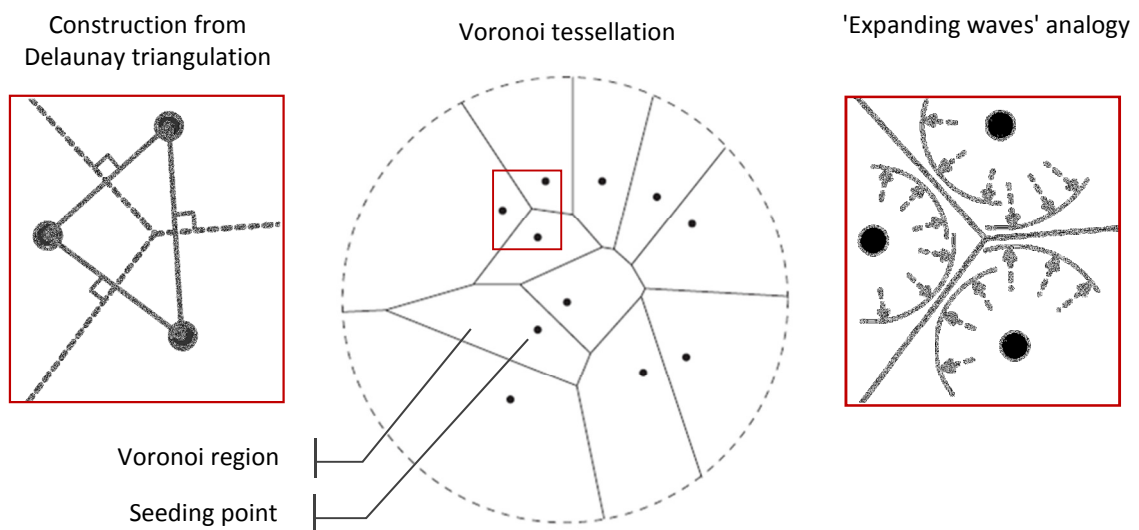


Figure 4-15: Construction of a 2D Voronoi diagram (based on a figure from [154])

A 3D Voronoi tessellation can be derived from a Delaunay tetrahedralisation in a similar manner. Rather than constructing perpendicular edges that bisect each edge of the triangulation, a surface is constructed normal to the midpoint of each edge of the tetrahedralisation. An example is shown in Figure 4-16.

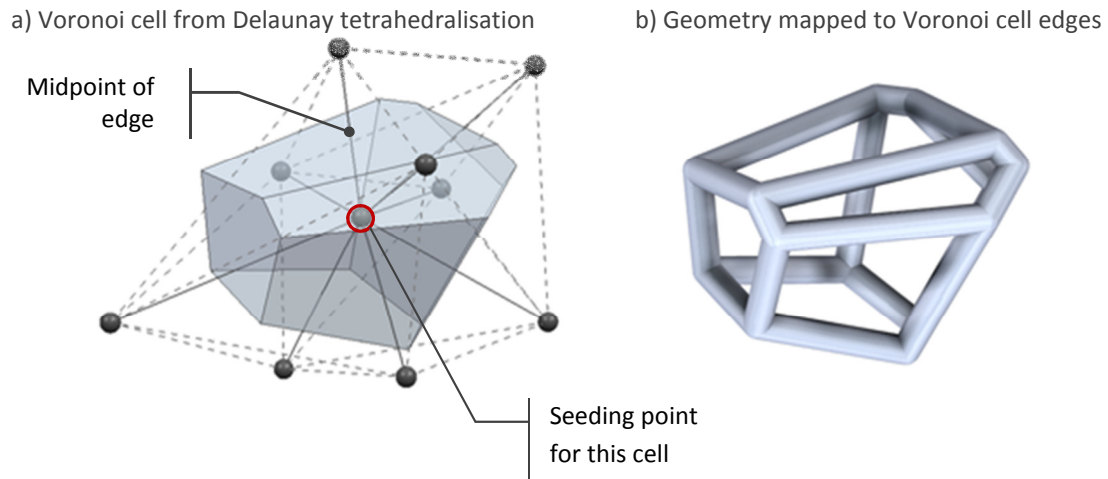


Figure 4-16: Constructing a random structure from a 3D Voronoi cell

By default, Voronoi tessellations are infinite in size. The cells that are not fully bound by neighbouring cells expand indefinitely. However a boundary can be included in Voronoi calculations to trim the tessellation. This is usually referred to as a clipped Voronoi tessellation [152]. This essentially means that to make a Voronoi tessellation conformal to a shape, it must be passed through the trimming method as discussed in Section 4.2.1.

Voronoi tessellations have been used to model natural cellular and crystalline materials [25,156,157] and molecular structures [158,159]. In these instances, a finite block of the tessellation is constructed, bound by a simple cuboid. Voronoi tessellations have also been implemented in the simulation of granular flow porosity [160] and as the basis of futuristic architectural concepts [161]. These Voronoi tessellations were clipped to slightly more complex shapes as a model of real-world objects. A particularly efficient algorithm for constructing Voronoi diagrams for use in refining tetrahedral meshes was developed by Yan *et al.* [152]. Implementations with Voronoi tessellations constrained by more geometrically complex boundaries are shown in Figure 4-17.



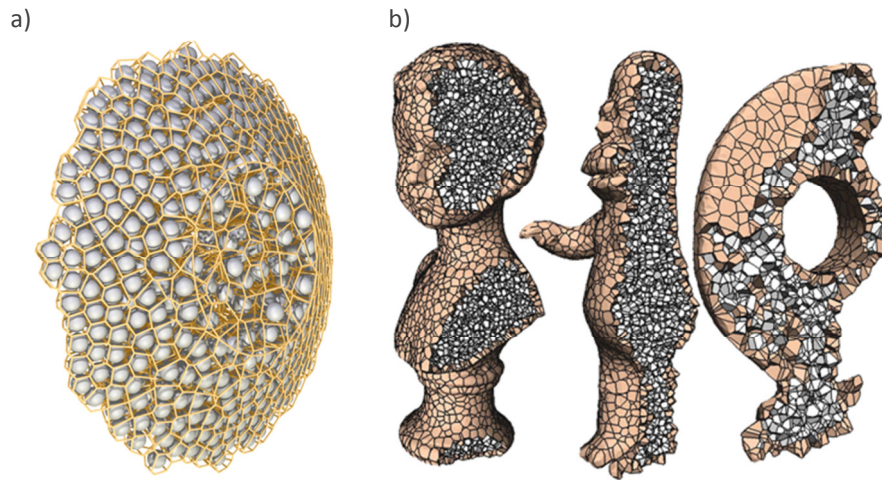


Figure 4-17: Applications of 3D Voronoi tessellations: a) in modelling of granular flow [160] and b) general modelling [152]

As with the general meshing techniques discussed in the previous section, a level of randomness is inherent in Delaunay and Voronoi tessellations. Random structure at the microscopic scale (as in many natural cellular materials) is considered as a bulk homogenous material at the macroscopic [25]. However, at the macroscopic scale that cellular structures can be replicated at by current AM technology, the ratio between cell size and part size is such that randomised structure would only serve to randomise its properties. With judicious placing of seeding points however, regular Delaunay and Voronoi tessellations can be constructed. By placing seeding points in a cubic array, a Voronoi tessellation will reveal an array of cubes. The more complicated regular array of seeding points as shown in Figure 4-18 will create a tessellation of truncated octahedra - a close approximation of the 'Kelvin cell' [25].

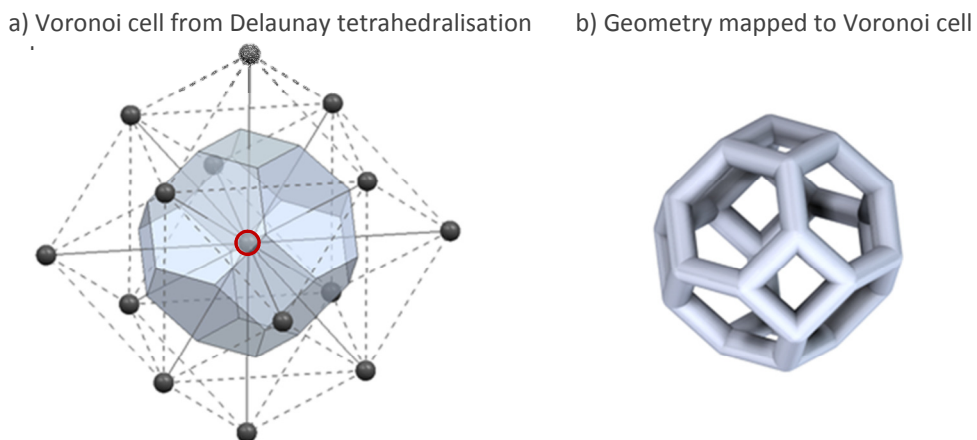
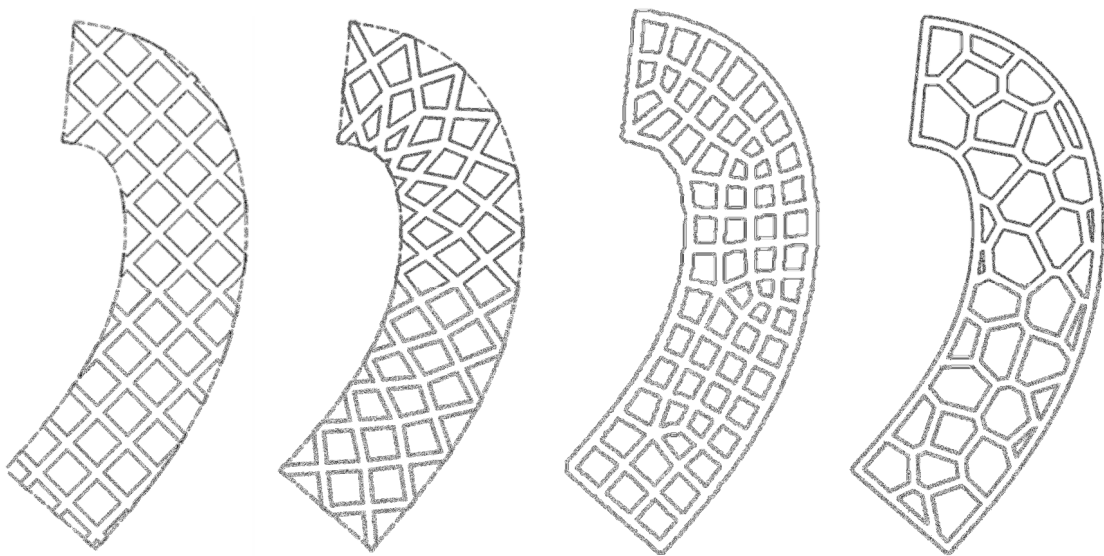


Figure 4-18: Constructing a regular Voronoi tessellation

### 4.3 Summary of Conformal Structure Methods

This chapter detailed several methods of conforming a structure to a shape: trimming, sweeping, meshing and Voronoi tessellations. A Voronoi tessellation is constructed from its dual, a Delaunay tessellation, which can actually be generated with FE meshing techniques. Because a Voronoi tessellation is infinite in size and must be trimmed to conform to a shape, it is categorised separately from standard meshing techniques. The trimming and sweeping methods generate structural geometry directly, while the mesh-based techniques generate what is essentially a polyhedral tessellation that must then have geometry mapped to it. Two dimensional examples of the conformal structure methods are shown in Figure 4-19 for comparison.

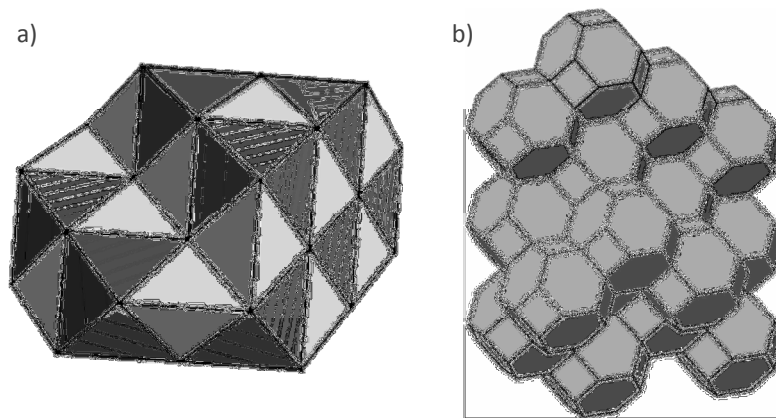


*Figure 4-19: Conformal structure methods: trimmed, swept, meshed and Voronoi*

The trimming method is the most robust in terms of the complexity of the shape it is possible to conform to. The mesh-based methods are also very robust, while the sweeping method is only suitable for certain types of conformal shape geometry. For this reason, sweeping is not considered further as a viable conformal method.

Each of these methods deform a structure in some way to fit a conformal shape. The sweeping and mesh-based methods consider the surface of the conformal shape as the basis of the structure generation, and 'fill in' the volume with a warped structure. As a result, these methods form 'closed' structures of intact (but varied) cells. In contrast, the trimming method populates a volume with structure and cuts it to fit the surface of the conformal shape. This results in a perfectly regular structure with 'open' cells at the boundary.

While meshing techniques allow the discretisation of an object into a variety of element types (such as tetrahedral or hex elements), the trimming method is compatible with any three dimensional tessellation. With this greater variety of tessellations brings a wider range of structure types, each with potentially different physical properties when manufactured. For example, the alternating cubic tessellation of tetrahedra and octahedra (as shown in Figure 4-20) will exhibit high rigidity, as the majority of the structure composed of triangular connections. In contrast, the bitruncated cubic tessellation of truncated octahedra (also shown in Figure 4-20) will produce a relatively compliant structure as it is very similar to the Kelvin cell which - as previously discussed - replicates the compressive profile of foams. While having completely different mechanical properties, both of these structures can be cropped to fit a repeatable unit cell that can be best implemented with the trimming method.



*Figure 4-20: a) alternating cubic and b) bitruncated cubic tessellations*

The trimming method is the most viable method of conforming a structure to a shape as it is robust and flexible. The trimming method can generate a conformal structure from any shape, can manipulate the widest range of structure types and minimises deformation of the structure. The main disadvantage is that regions of the structure are weakened from the boundary being cut. Ways around this are discussed in detail in Chapter 6.

## 5 | Research Methodology

### 5.1 Problem Identification

The literature has shown that there is a requirement for efficient modelling of conformal lattice structures for additive manufacture. The literature has shown that conventional CAD is unsuitable for the task as it is an inefficient method for representing models of high hierarchical complexity. As such conventional CAD cannot fully exploit the geometric freedom associated with AM when modelling lattice structures. Other methods of geometry representation were investigated and the voxel method in particular shows promise in the design of complex structures. Initial work on using the voxel method for structure generation was carried out by Hollister *et al.* [126], although due to the limited requirements the resulting structures were relatively simplistic.

Methods to conform a structure to a particular shape were also discussed and four were identified: trimming, sweeping, general meshing and Voronoi tessellations. The trimming method is the most widely adopted as it is the most robust, the most flexible in terms of structure types that can be processed and subjects the structure to the least deformation. The main issue with the trimming method, however, is the loss of connectivity between trimmed struts at the boundary of the structure. This region of trimmed struts are potentially weakened by the removal of supporting neighbours. No implementation of the trimmed method satisfactorily addresses this.

In short, a gap exists that could be filled by a voxel-based trimming method of conformal structure generation that addresses the issue of the weakened boundary. By ensuring compatibility with laser sintering, the method can be advanced to fully exploit the geometric freedom afforded by additive manufacturing.

## 5.2 Research Aims

The following work is split into two broad research aims:

1. Investigate a method to retain structural connectivity at the boundary of a trimmed structure.
2. Develop a conformal structure method that:
  - a) Utilises the trimming method of conformal structure generation
  - b) Implements research aim 1
  - c) Develops the voxel method to fully exploit the geometric freedom of AM
  - d) Efficiently integrates into the conventional route - the route of data flow from design to additive manufacture as described in Chapter 2

## 5.3 Research Approach

Research Aim 1 is developed as 'preliminary work' in Chapter 6. At this stage it is a concept that is investigated through an implementation of boundary representation. At the same time, a 'cut down' version of boundary representation is investigated to determine if this more conventional modelling method is a potential route when geometric accuracy is reduced. The chapter shows that reducing accuracy only works to a degree.

The conformal structure method required by Research Aim 2 is developed throughout Chapter 7. The method exploits voxel modelling to create an advanced structure trimming method that fully integrates this alternate modelling method with the existing infrastructure associated with the conventional route of data flow discussed in the literature. The concept developed as a response to Research Aim 1 is implemented, as are a number of extra features that capitalise on the geometric complexities that additive manufacturing is capable of.

Chapter 8 details testing of samples passed through the conformal structure method as, despite the advantages it presents, parts made by the method exhibit an extra artefact of surface roughness not present in conventionally designed AM parts. The primary goals of this testing are to quantify this artefact and suggest means to minimise it if necessary. Conclusions are discussed in Chapter 9, as are suggestions for future work to further develop the conformal structure method.

## 6 Preliminary Work: Adapting Boundary Representation

As the literature review demonstrated, the trimmed structure method of generating conformal structures is the most robust and flexible. However, out of the methods discussed it is unique in that elements of the structure are removed, i.e. the struts at the boundary. This implies a weakness at the boundary of the lattice structure where struts are no longer supported by neighbours. This chapter discusses existing skinning methods to address this and presents an alternative, novel method.

The literature also showed that the underlying advanced boundary representation (B-rep) method of geometric modelling used by CAD software was not suitable for generating complex conformal structures. The literature review showed that the simplified faceted B-rep modelling was more suitable. The reduction in accuracy of faceted models allows larger structures to be modelled before computer memory limits are reached. This chapter utilises what has been termed a 'sampled B-rep' method which takes this simplification of geometry further. The work discussed in this chapter was implemented in Matlab [162]. Although not the most promising modelling method identified by the literature, it is relatively straightforward to implement and thus useful for exploring the concept presented in the following section.

### 6.1 Skinning a Trimmed Structure

As discussed in Chapter 4, when trimming a structure to fit a conformal shape, the cells at the boundary are essentially cut open. This inevitably weakens the region of the structure at the boundary as the cut cells are no longer self-supporting. A strut, once it has been cut, immediately loses the support it gained from its now-removed neighbours. The entire boundary region of a trimmed structure will be weaker than the bulk structure which has serious consequences for the design of such conformal structures. One method to overcome this is to apply a skin to the trimmed structure boundary - a solid conformal shape that fully or partially encompasses the structure. An example of this is shown in Figure 6-1.

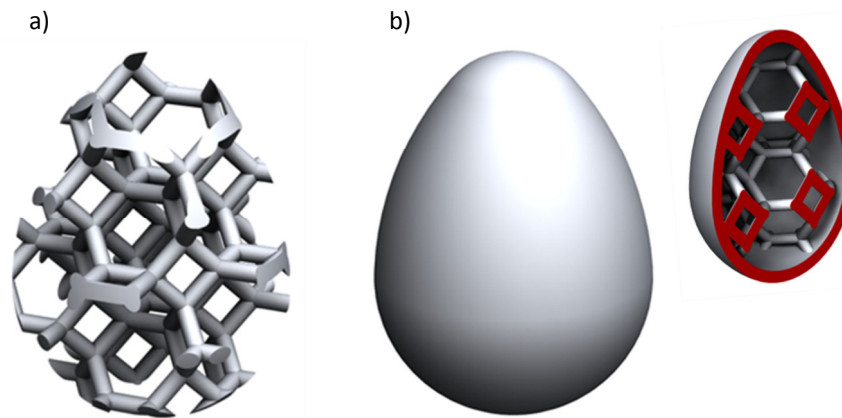


Figure 6-1: A trimmed structure with a) no skin and b) a solid skin

A solid skin increases the stiffness of a trimmed structure at the boundary. Sometimes potentially useful, this is nonetheless an undesirable modification for an application where flexibility is a key design requirement. A fully encompassing solid skin also makes manufacture a more difficult proposition. A completely solid skin makes it impossible to remove any supports as required by the additive manufacturing process, whether the additional support structure constructed by stereolithography or FDM, or the un-sintered, supporting powder that surrounds the part in powder-based processes such as laser sintering.

Drainage holes can be built into the solid skin at key locations to allow the removal of supporting material, however the effectiveness of this cannot be guaranteed. This may be completely ineffective for complex structure designs as well as particular conformal shape geometries that trap supporting material within them.

### 6.1.1 The Net Skin

This chapter presents a compromise between a solid skin and no skin. It has been named a 'net skin' by the author for its resemblance to a fishing net - an example is shown in Figure 6-2.

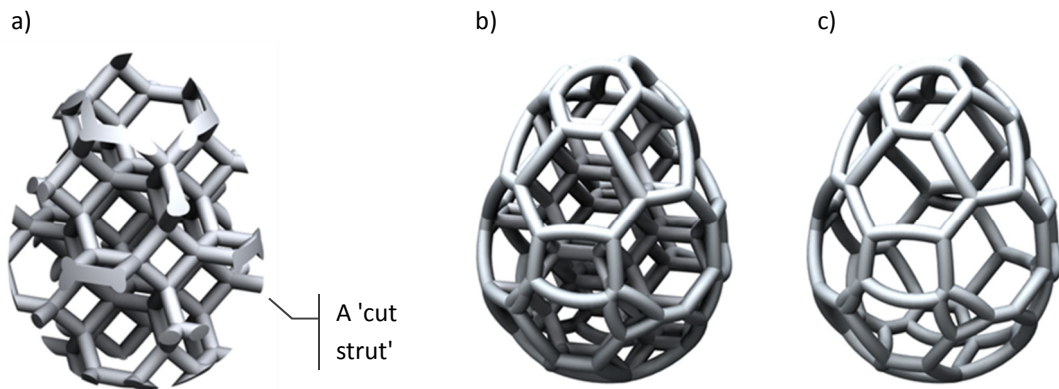


Figure 6-2: a) a trimmed structure, b) a trimmed structure with net skin and c) just the net skin

The net skin, defined by this work, is a method to re-connect the cut struts of a trimmed structure. If the boundary of a trimmed structure is considered as an array of individually trimmed cells, then cut struts can be grouped with others of the same cell. With this knowledge, these groups are bridged with new struts that form the net skin.

Any repeating structure can be defined with a unit cell of some sort. A unit cell is an arrangement of struts, and this arrangement usually takes the form of a polyhedron or group of polyhedra. Considering the polyhedron by itself, if it is trimmed to the same boundary as that of the structure, a new face with its own edges is formed. New strut geometry can be mapped to these edges to form connecting struts. This concept is illustrated in Figure 6-3. If this operation is performed with every intersecting cell of the structure these connecting struts combine to form a net skin. The polyhedral tessellation whose edges are used to map structure geometry is hence called a structure's 'base tessellation'.

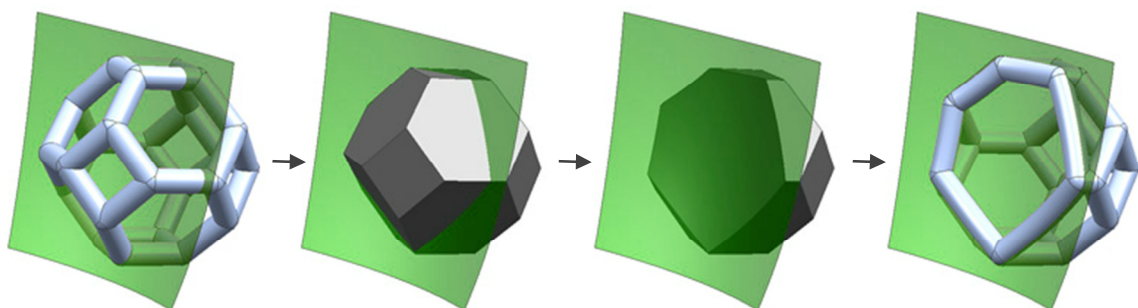


Figure 6-3: Constructing a net skin for a single cell by considering a structure as a based on a tessellation of polyhedra

A net skin can be considered as an element of a structure, rather than a specific type of geometry. Any type of geometry can be mapped to the newly formed edges that are generated when trimming



a structure's base tessellation. In terms of actually creating the net skin geometry shown in Figure 6-4, it is a relatively simple operation to sweep a 2D section across the length of a 3D guide: 'a' is generated by sweeping a circular profile across each edge, 'b' from a square profile and 'c' constructs a helical curve from the edge and subsequently sweeps a circular profile across that.

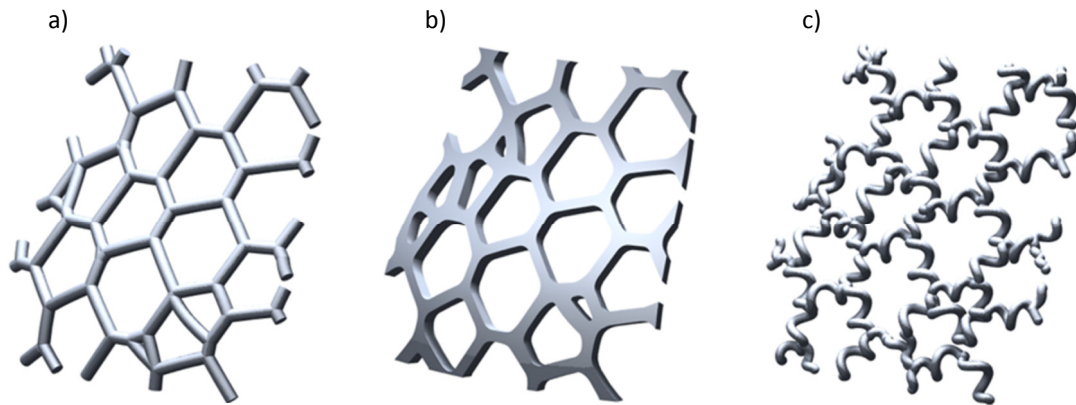


Figure 6-4: Types of net skin

Knowledge of the connectivity of the structure (i.e.: the a base tessellation) is important to ensure robust construction of net skin. A structure could potentially be trimmed without this knowledge, with nearest neighbours reconnected, but this will not always work. This is shown for a thin-walled conformal shape in Figure 6-5.

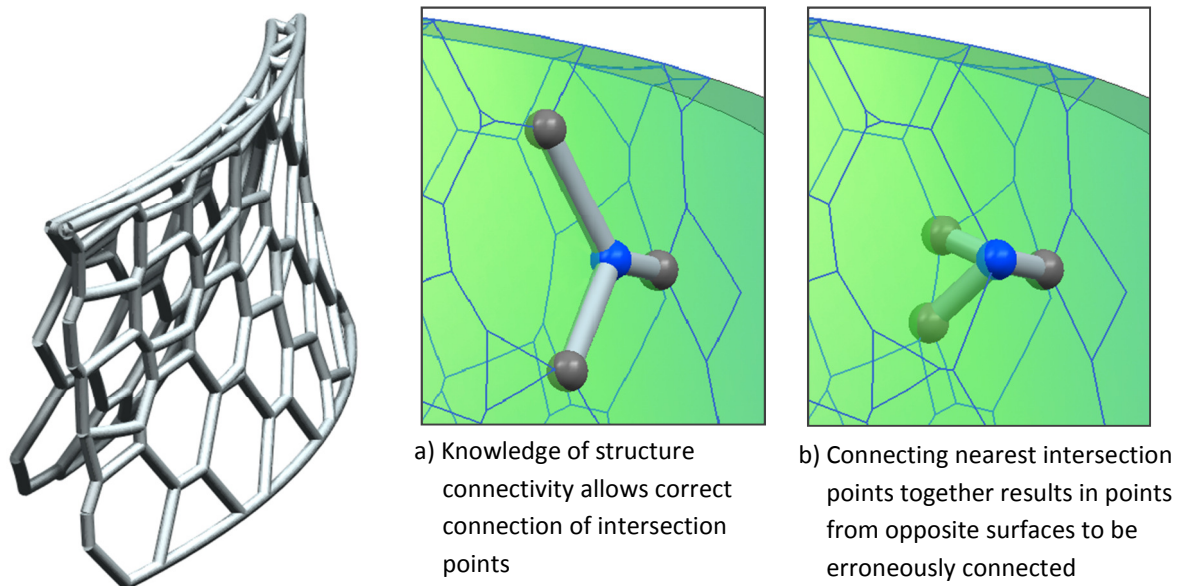


Figure 6-5: 'Nearest neighbour' failure to properly re-connect cut struts on a thin-walled part

## 6.2 B-rep Net Skin Construction Method

The method presented in this chapter is solely concerned with the generation of a net skin for a particular conformal shape; the generation of the trimmed structure would be a second step. Generating a net skin is essentially the Boolean intersection of a conformal shape with the base tessellation of a particular structure. In an attempt to speed up the process, the B-rep method of representing geometry is reduced to the minimum information required to fully describe geometry.

There are two phases of geometry manipulation to generate a net skin:

1. The intersection calculation between conformal shape and base tessellation. All geometry input into the method is sampled:
  - The conformal shape is reduced to a series of surface points. Rather than the standard set of vertex, edge and face tables used to represent B-rep geometry (see: Section 4.2.1), the conformal shape is defined by points only - essentially just a vertex table.
  - The base tessellation is also reduced to a series of points, only the points are still structured in a way that it is known what face and edge each point belongs to. This is achieved in a similar way to standard B-rep (tables that reference connectivity) but is distinct in that this description is not modified by the intersection calculation. The intersection calculation reads in the description of both conformal shape and base tessellation and generates new tables that represent intersecting geometry.
2. The generation of net skin geometry. The intersection information calculated by the first stage is used as template to build actual net skin struts. This uses faceted B-rep geometry to facilitate straightforward conversion to the STL format.

The following section details the basic method developed to generate net skin geometry, essentially the initial attempt. Section 6.4 details the individual changes made in the development of a more advanced method that still works on the basic principles of the initial method.

## 6.3 Basic Implementation of the Method

### 6.3.1 Defining the Conformal Shape

To demonstrate the principle that this method can be used on real world objects, a point cloud representing scan data is imported into Matlab. The point cloud is shown in Figure 6-6.

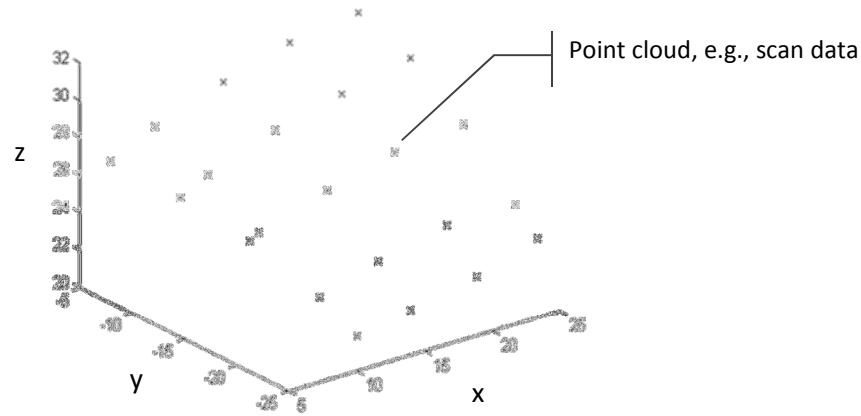


Figure 6-6: Representation of point cloud data

A surface is constructed based on this point cloud in the form  $z=f(x,y)$ , or where the height (in the z-axis) of each cloud point is a function of its x and y coordinates. From this the surface is created by generating a series of equally spaced points (in the x-y plane), the heights of which calculated from the interpolation function. In this case the interpolation method is linear - the surface generated between the original point cloud is linearly interpolated, as shown in Figure 6-7.

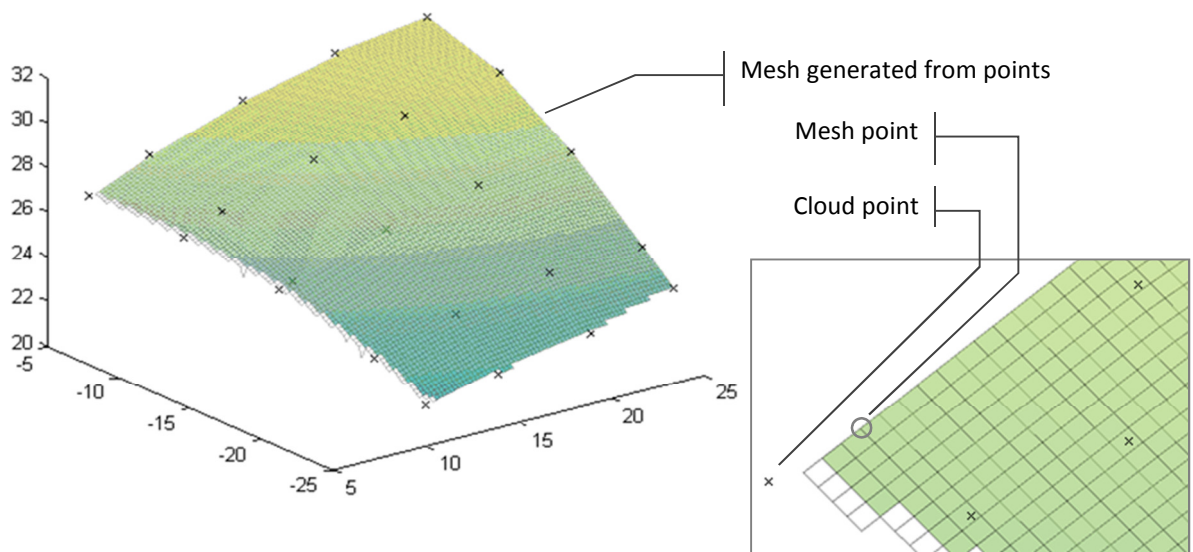


Figure 6-7: Constructing a surface from the point cloud

### 6.3.2 Defining the Base Tessellation

The next step is to populate space with a large enough base tessellation to fit the surface. An example truncated octahedron that intersects with the surface is shown in Figure 6-9. The base tessellation is created by calling a predefined matrix that represents a unit polyhedron and placing an instance of it at every point required to construct a tessellation that fills the surface. The unit

polyhedron matrix consists of vertex and face information, as depicted in Figure 6-8. Each face of the polyhedron is defined by the co-ordinates of the vertices that form it. Included for each vertex are variables that multiply and translate each point to allow control over polyhedron size (practically, the structure's cell diameter) and location in space (not shown in figure).

In this method, the base tessellation is evaluated as individual faces rather than a connected polyhedra. This is achieved through a FOR loop; a standard programming operator that allows code to be repeated a finite number of times. The FOR loop iterates through every face of the base tessellation. Any faces that intersect the surface will generate individual net skin struts that join up to form the complete net skin.

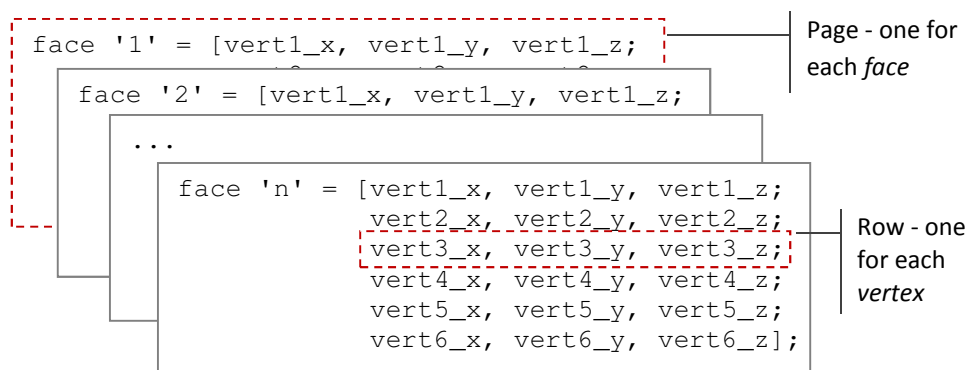


Figure 6-8: Structure of unit polyhedron matrix

Because a matrix must have the same number of rows and columns per page, the 'unit truncated octahedron matrix' is actually two matrices - one that defines the polyhedron's eight hexagonal faces (6 rows of coordinates in each of the eight pages for the hexagons' six vertices) and one for the six square faces (4 rows in 6 pages). Due to the layout of the unit polyhedron matrix, the edges of the polyhedron can be defined. The vertices that join to form a face are listed in a clockwise order, so the edges that define a face lie between sequential vertices. 'Edge 1' is defined by vertex 1 and vertex 2, edge 2 by vertex 2 and 3 and so on up to edge 6, defined by vertex 6 and vertex 1.

From this edge information a series of interpolated points are generated for each of the edges of the polyhedron. The interpolated points are stored within an interpolated points matrix; each set of edge interpolation points is grouped in together with the other edge sets that define a face of the polyhedron. This is what facilitates the correct reconnection of trimmed struts to form the net skin.

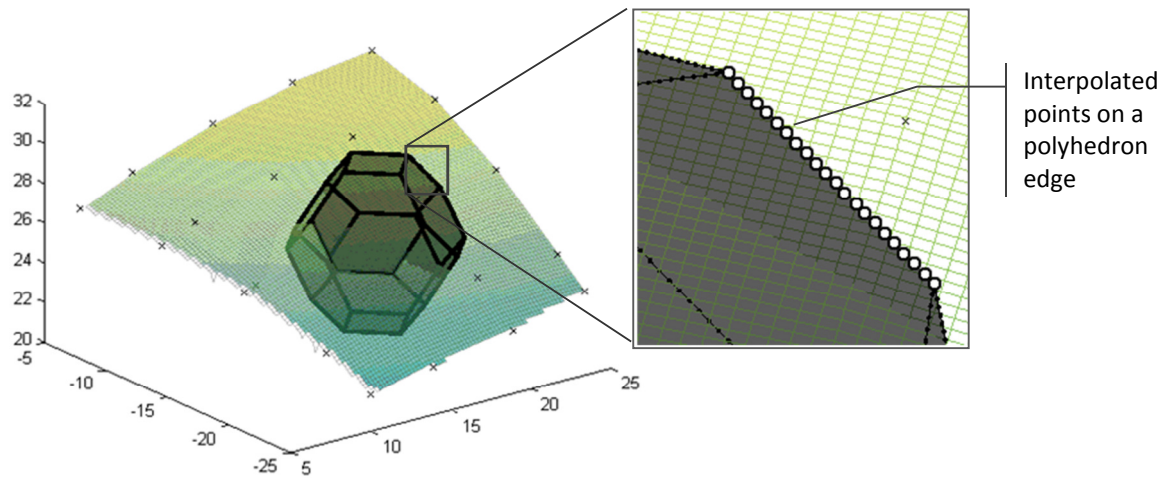


Figure 6-9: A polyhedron intersecting with a surface and interpolated points generated on its edges

Both the number of interpolation points and density of conformal shape points can be controlled with a resolution factor. The higher these two factors, the more accurate the intersection calculation. The intersection between each face and the surface is calculated by searching for matching pairs of co-ordinates between the two. This search is conducted within a tolerance as it is unlikely that between sampling both the surface and faces will yield identical co-ordinates. The base tessellation is checked for every edge of every face and in determining intersection points between faces and conformal shape, the method assumes the following (illustrated in Figure 6-10):

- That where an edge intersects the conformal shape, one and only one intersection point is required.
- That any face that intersects the conformal shape does so on two and only two of its edges.

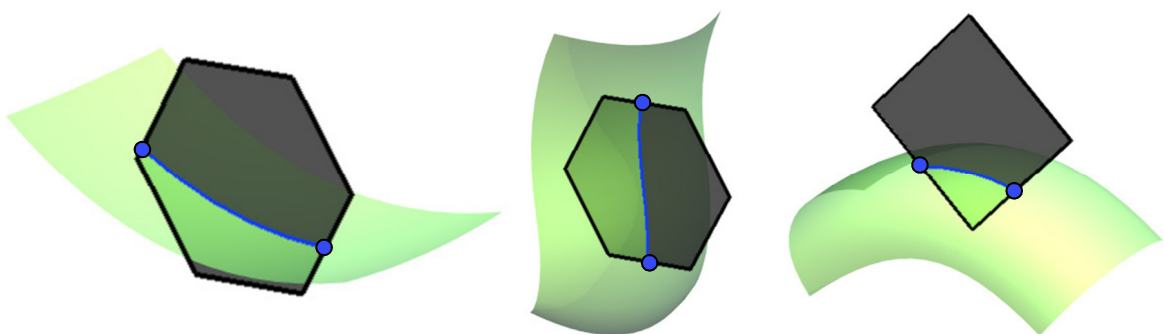


Figure 6-10: Instances of a face intersecting a surface

Where matching pairs of co-ordinates are identified, the intersecting edge point is written to an 'intersection points matrix'. These intersection points are shown in Figure 6-11. As each face is

checked, if an edge is found to intersect the surface the method expects to find a second. These are grouped as a pair in the generated intersection points matrix.

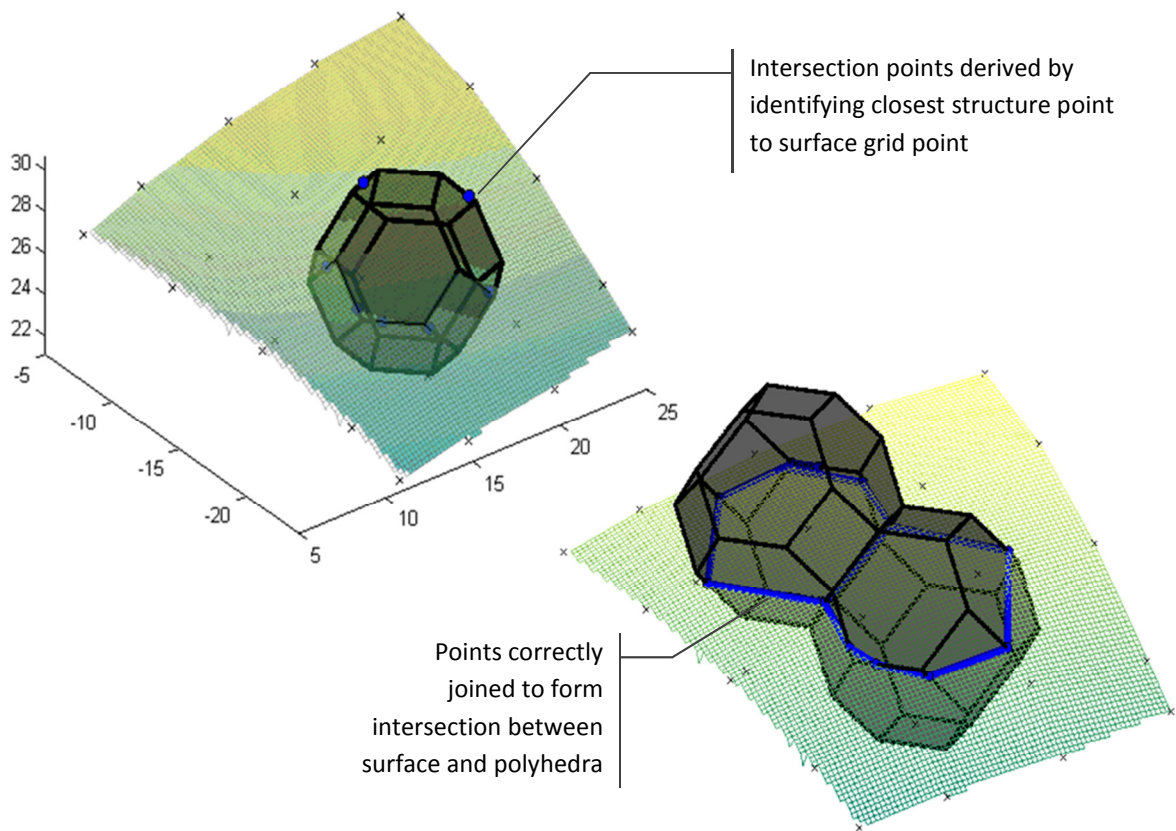


Figure 6-11: Finalised intersection points and correctly joining those points

### 6.3.3 Connecting Intersection Points

The next step is to correctly join the intersection points with lines to generate the net skin, also shown in Figure 6-11. Due to the way 'intersection points matrix' is structured, the connectivity of the original polyhedra is retained and the pairs of intersection points of each face form the start and end of each intersection line.

A straight line is constructed between points rather than a curved line that follows surface curvature, so only an approximation of the surface is constructed. However, because it is likely (in most cases) the surface curvature of the conformal shape is large enough to be considered almost straight between intersection points, this approximation is insignificant.

At this point, only the lines that form a wireframe representation of the net skin have been constructed. 3D solid geometry must be constructed that follow these lines to generate the structure. As a more straightforward example for this method, cylindrical struts are constructed along the intersection lines. Further improvements to the method could be developed that take this information to construct more complex net skin geometry, such as helical struts.

#### 6.3.4 Constructing Net Skin Geometry

Initial attempts to generate structure topology were focused on sweeping a cylinder along intersected lines, illustrated in Figure 6-12a. This replicates the method conventional CAD utilises to sweep geometry along curves. It is an appropriate method, although spheres must be generated between struts to close gaps on the net skin. As shown in Figure 6-12b, spheres must be positioned at the ends of each cylindrical strut to bridge the gap between struts at angles to each other.

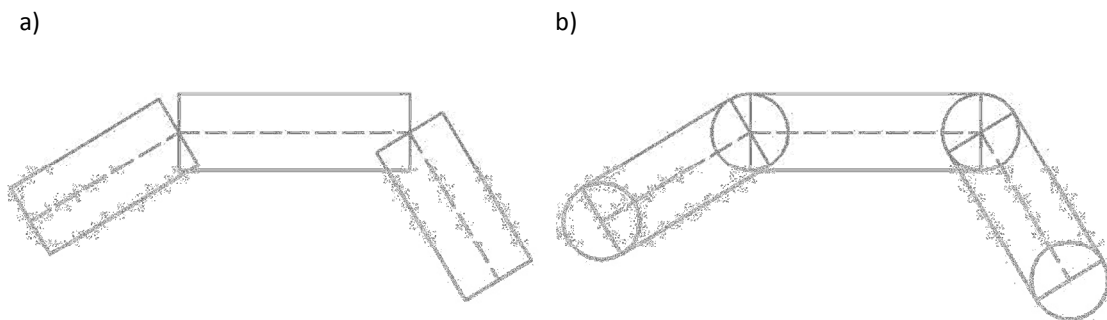


Figure 6-12: The necessity of spheres when mapping geometry to struts

Without a Boolean operation to unite them, this process also generates three separate bodies per strut. Although a perfectly feasible approach it is not the most elegant. For straight struts, a method was investigated to efficiently construct the sphere-capped geometry required as shown in Figure 6-12b.

The first step of the method is to generate points of a surface of a sphere at each intersection point on the net skin, demonstrated for a single line in Figure 6-13. The diameter of these spheres in practical terms is the 'strut diameter' of the structure. Sphere surface points are generated with a built-in Matlab function, derived from the equations shown in Figure 6-13 [163]. A specific point on the surface of a sphere can be measured with two angles: theta ( $\theta$ ) and phi ( $\phi$ ) analogous to longitude and latitude respectively. A series of sampled points are generated from these equations.

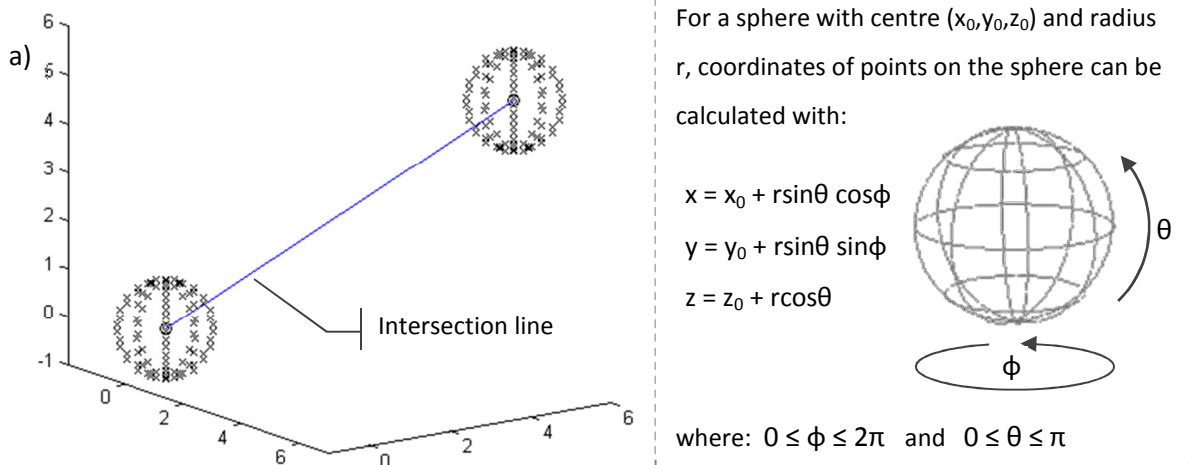


Figure 6-13: Generating the points to be enclosed by a convex hull

A convex hull is generated between the pair of spheres to create a watertight shell of polygons. This is also achieved with a built-in Matlab function. A simple 2D example is shown in Figure 6-14. For a set of points, the convex hull is the smallest convex polyhedron that contains all these points - analogous to a rubber band stretched around nails hammered into a board [164,165].

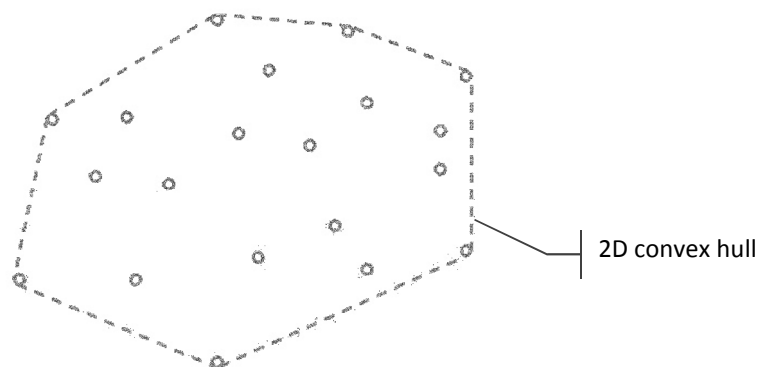


Figure 6-14: 2D convex hull example

Because the spheres used to generate convex hulls are translated between points irrespective of line orientation, where the convex hulls of intersecting struts meet they mesh into each other creating a smooth join. This is shown in the inset of Figure 6-15. A second advantage to this approach is that the convex hull is constructed with triangular polygons - facilitating straightforward conversion to the STL file format. This provides integration of the method with the conventional route from design to additive manufacture as defined in Chapter 2.



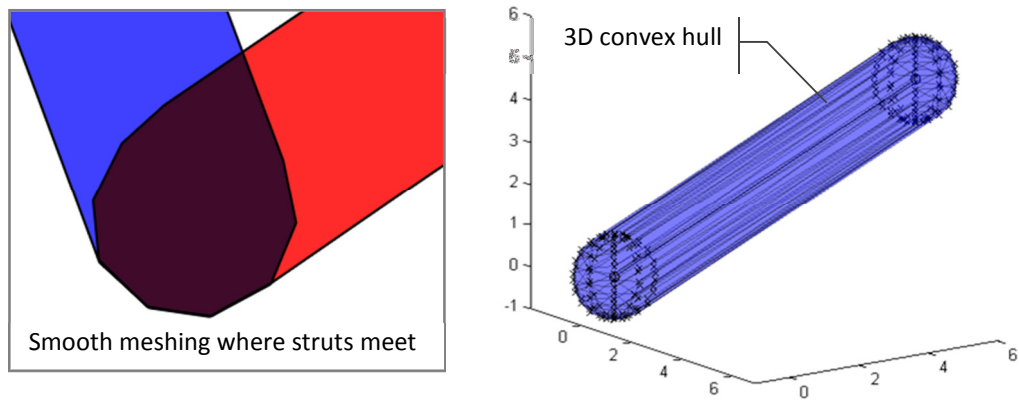


Figure 6-15: Net skin struts generated from convex hulls

By varying the sampling of the sphere surface points equation, control is gained over the level of faceting of the strut. Reducing the resolution will reduce the number of surface points generated, in turn reducing the number of polygons constructed and thus the file size of the written STL, as shown in Figure 6-16. There are more points generated by this method that are required for any particular strut (see that approximately half of the sphere points are within the convex hull in Figure 6-15), but there is no extra solid geometry constructed. Compared to the cylinder/sphere method, the polygon count for a similarly faceted convex hull is considerably lower.

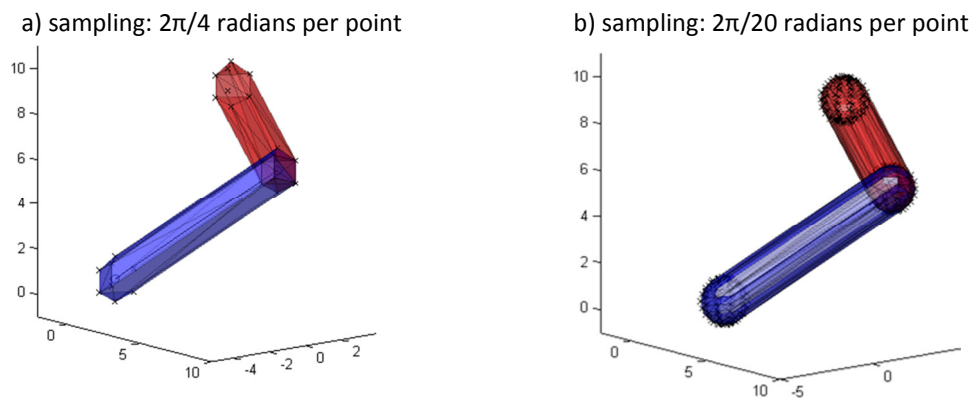


Figure 6-16: Varying level of faceting of convex hull struts

### 6.3.5 Writing Formats Suitable for Manufacture

There are two types of STL file format, both compatible with additive manufacture machines: ASCII and binary format. As discussed in Chapter 2, The ASCII format is structurally simpler to binary but less efficient with more complex geometry. Whereas the binary format is preferred for functional use, the more straightforward structure of the ASCII format is ideal for prototyping software. For this

reason, the geometry to STL conversion process writes ASCII format files. An example of the ASCII format is shown in Figure 6-17.

```

solid 001
  facet normal 0.000000e+000 -1.000000e+000 0.000000e+000
    outer loop
      vertex 0.000000e+000 0.000000e+000 0.000000e+000
      vertex -1.500000e+002 -1.000000e+002 0.000000e+000
      vertex -1.500000e+002 -1.000000e+002 -5.000000e+001
    endloop
  endfacet
  facet normal 0.000000e+000 -1.000000e+000 0.000000e+000
    outer loop
      vertex 0.000000e+000 0.000000e+000 0.000000e+000
      vertex -1.500000e+002 -1.000000e+002 0.000000e+000
      vertex -1.500000e+002 -1.000000e+002 5.000000e+001
    endloop
  endfacet
endsolid 001

solid 002
  facet normal ...
    outer loop ...
      endloop
  endfacet
endsolid 002
  
```

Figure 6-17: STL ASCII format [2]

Each strut generated in the process is considered as a separate body and written accordingly, with each polygon of the strut described by its vertices and normal data. The conversion process takes place directly after the strut geometry creation. The output of this process is an STL file of the entire net skin of the conformal part. The method is controlled with a GUI (graphical user interface) that prompts the user for structure parameters (i.e. cell and strut diameters) as well as the various resolutions and tolerances used to calculate intersection points, shown in Figure 6-18.

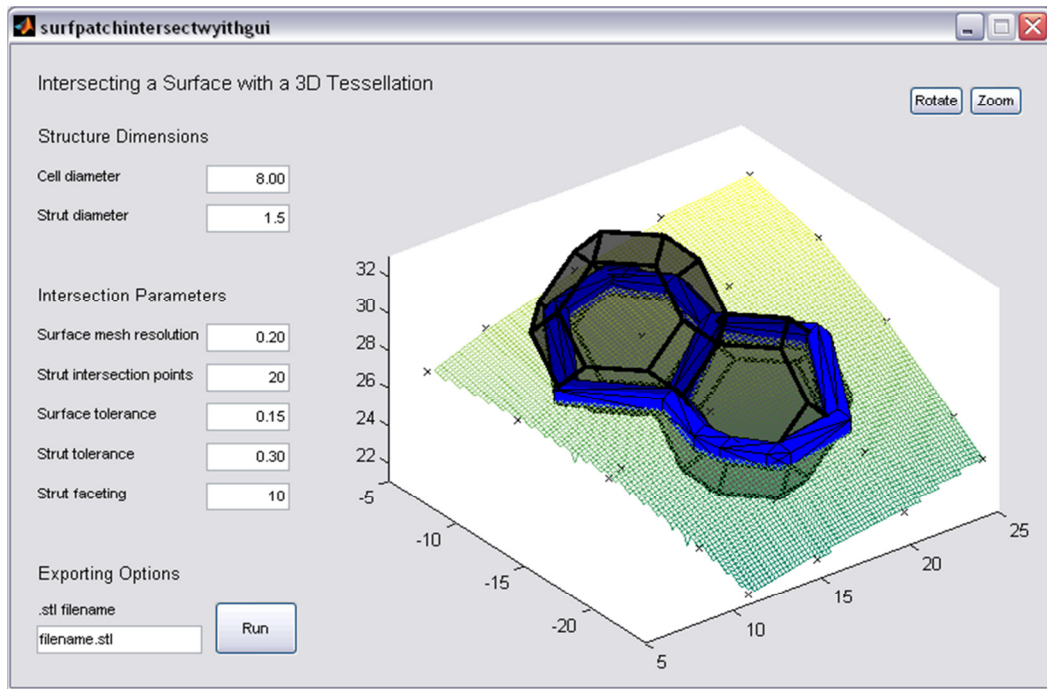


Figure 6-18: The completion the basic implementation

### 6.3.6 Summary of the Basic Implementation

This basic implementation of a method to construct a net skin uses a faceted form of boundary representation. The method simplifies the intersection between conformal surface and base tessellation by sampling both. Rather than calculating a perfect intersection between an edge and a mathematically complex surface, each are reduced to a series of points and similar pairs are identified through comparing their co-ordinates. However, the way this initial attempt was implemented is not suitable for scaling up. Currently, for every edge that is checked for intersections, every surface point is checked. The method slows dramatically for larger surfaces - if the number of points that describe the surface doubles, the time taken to check each edge for matches also doubles.

Additionally, the representation of the base tessellation is inefficient. The method checks the base tessellation for intersections one face at a time. For a single polyhedron, each face is checked once. For the two neighbouring polyhedra shown in Figure 6-18, the two faces that touch each other are both checked for intersections. In this example, two identical net skin struts are generated in the same location. As well as duplicating net skin struts, the actual storage of the base tessellation is also larger than necessary. For any edge of a polyhedron there are three copies of it stored.

Furthermore, the type of surface used to represent a section of a conformal shape is also limited in terms of complexity of shape achievable, as discussed in the following section.

## 6.4 Advanced Implementation of the Method

A more advanced version of the method was subsequently implemented to address the three limitations discussed in the previous section. The underlying method (i.e. the sampling of surface and base tessellation into a series of points to be compared for matches) is retained, so only the improvements are discussed. Additionally, the advanced implementation also presents a manner to automatically determine where the base tessellation is likely to intersect with the conformal shape and only constructs it in that region.

### 6.4.1 Representation of the Conformal Shape

In the basic implementation of the method, a surface in the form ' $z=f(x,y)$ ' was generated to define a section of a conformal shape. This surface is constructed from an irregular array of points, representing a point cloud from scan data. From this surface, a more regular array of points can be derived which is advantageous for intersection checks, as every region of the surface has a similar density of points. This means that intersection checks for one region of the surface will be as accurate as another and, thus, the same tolerance can be used for the whole intersection calculation.

There are two issues with this approach. Firstly, the resolution of the surface in the form ' $z=f(x,y)$ ' actually varies across it, depending on the surface curvature. The interpolation method spaces points equally along the x-y plane and projects them onto the surface, thus the resolution of the mesh is essentially lower on steep surfaces. This is shown in Figure 6-10 - viewed from the top, the surface appears to be an equally spaced grid, but on a steep section of the surface it is shown that the grid is stretched. The consequence of this is that the surface is less accurate on steeper sections. This can be overcome by setting the initial mesh resolution higher, but practically, a better way of constructing the surface mesh is required.

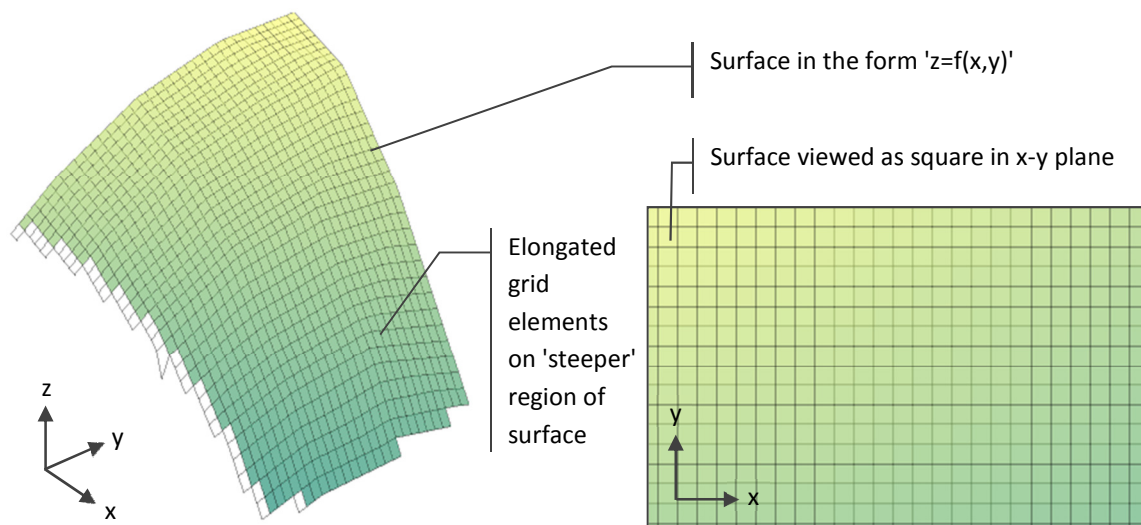


Figure 6-19: Mesh grid stretching on steeper curvature

A second issue is that it is not realistic to expect scan data as a common means to represent a conformal shape. As the literature review showed, conventional CAD is not suitable for representing large and complex lattice structures (high hierarchical complexity). CAD software is however suitable for models with shape complexity, i.e. technically able to represent any kind of shape. It would be considerably more useful for this method to accept CAD models as an input for the conformal shape, or indeed any kind of model discussed in the literature.

The advanced method has addressed both of these issues to some degree, by accepting an FE mesh as an input for the conformal shape. Rather than the volume meshes discussed in Chapter 4, this advanced method uses a surface mesh of the conformal shape, i.e. a mesh of polygons that solely represents its boundary, as shown for a region of a body armour concept as shown in Figure 6-20a. The mesh type used is comprised of triangular elements.

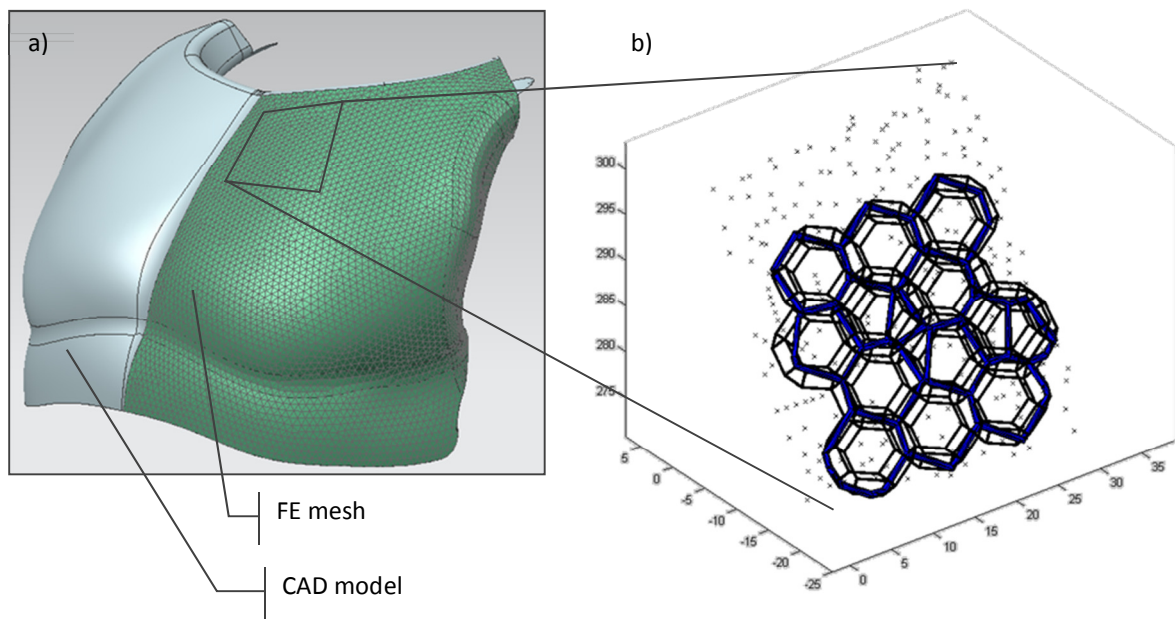


Figure 6-20: a) FE mesh generated from CAD model of a section of a body armour concept  
 b) Net skin struts constructed from intersection with FE mesh nodes (every 10th node shown)

FE meshing algorithms are packaged into some CAD programmes (such as Siemens NX), as well as in FEA software, and as such are fairly common. This is just as a means to sample the conformal shape and reduce it to a series of points for the intersection calculation. By meshing the conformal shape, elements and nodes are saved to a file - the nodes are extracted by the advanced method and used as surface points, as shown in Figure 6-20b. Although not a perfectly regular spacing (as this is not necessarily possible for complex shapes) the node spacing can be controlled by the meshing software, with distance between nodes set to within a tolerance.

The use of STL files was considered as another means of converting a CAD model to a triangular mesh, as CAD models are converted to STLs before fabrication already. However, unlike FE meshing, standard triangulation in STL conversion is controlled by surface curvature rather than by triangle size, as discussed in Chapter 2 and shown in Figure 6-21. This means that flat surfaces are described by a few large triangles, while highly curved surfaces are described by more smaller triangles. This is not conducive to a regular spacing of surface points and thus FE meshing was utilised.

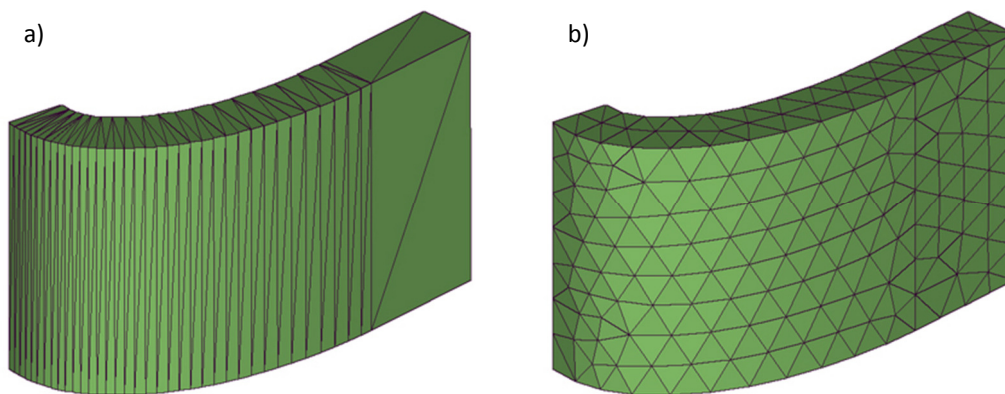
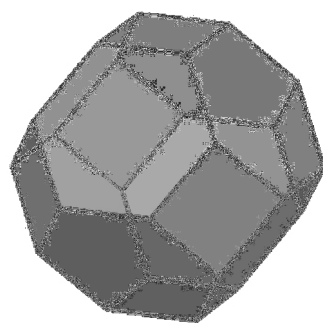


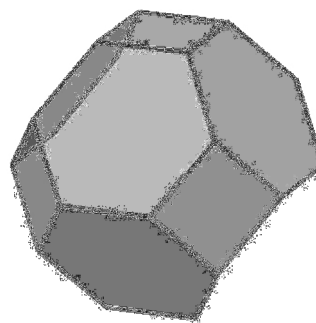
Figure 6-21: Triangulation of a shape: a) standard STL conversion and b) FE mesh

#### 6.4.2 New Unit Polyhedron

For the specific base tessellation used in the previous implementation (the Kelvin cell), it was shown that, as the basic method worked on a face by face basis, duplicate net skin struts were generated where neighbouring polyhedra touched. This would be the case for any tessellation used in the method. This is because a complete polyhedron was used as the repeating unit. By removing some faces from the unit polyhedron this duplication can be eliminated. For the Kelvin cell, this partial polyhedron is shown in Figure 6-22.



Complete polyhedron:  
 Vertices: 72 (48 duplicates)  
 Edges: 72 (36 duplicates)  
 Faces: 14



Partial polyhedron:  
 Vertices: 18  
 Edges: 24  
 Faces: 7

Figure 6-22: Complete and partial unit polyhedra - duplication apparent when tessellated

The manner in which this new unit polyhedron is checked for intersection with the conformal shape has also been changed. The basic implementation discussed in the previous section checked the base tessellation one face at a time. As such, each face was described by its own set of vertices and edges. As each vertex of the Kelvin cell is shared by three faces, each vertex was listed three times in the

unit polyhedron matrix. Similarly, each edge was listed twice as two faces share every edge. While this doesn't lead to any duplicated net skin geometry to be constructed (like duplicated faces do), this inflates the size of the matrices that represent the base tessellation unnecessarily, and hence the time to read them and the overall speed of the process increases.

The advanced implementation fully represents the new unit polyhedron and minimises duplication with a more formally structured series of matrices. Rather than a hierarchical structure where vertices are grouped according to which face they are in, the unit polyhedron is defined by a 'vertex matrix', an 'edge matrix' and a 'faces of edge matrix', as shown in Figure 6-23. This kind of geometry definition is similar to the structure of a standard boundary representation model as discussed in Chapter 2. The vertex matrix lists the co-ordinates of each of the polyhedron's vertices. The edge matrix lists the pairs of vertices that form each edge. The 'faces of edge' matrix list the two faces that share each edge. Some edges only exist on one face - those edges that border the polyhedron. While the exact structure of these matrices is specific to this particular tessellation, similar matrices could be defined for any tessellation.

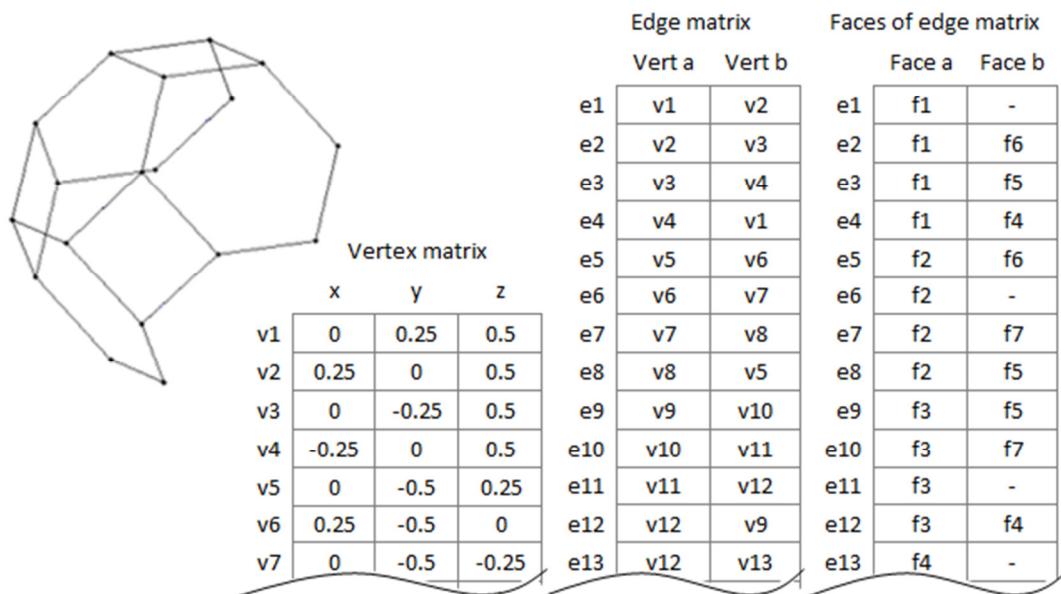


Figure 6-23: Partial polyhedron definition matrices

Checking for intersections between polyhedron and conformal shape works in the same way as described for the basic implementation of the method. The method generates interpolated points for each edge of the polyhedron as described in Section 6.3.2. For the intersection calculation, matches between these interpolated edge points and conformal shape points are identified. At this point, just intersection points are known. To generate the net skin, the correct connectivity between intersection points must be identified. While the vertex and edge matrices construct the polyhedron;



the 'faces of edge' matrix is used to connect the intersection points to form the net skin. For the particular polyhedron shown in Figure 6-24, intersection points a, b, c and d were found. 'a' is one of the interpolated points generated for edge 11 (e11). Similarly, 'b' is on edge 9 (e9), 'c' on edge 17 (e17) and 'd' on edge 19 (e19). This is already known to the method - the method checks for intersections between the polyhedron and conformal shape one edge at a time, so when an intersection point is found the edge that it is on can be identified. To correctly join the edges, the 'faces of edge matrix' is used to group the intersection points into pairs.

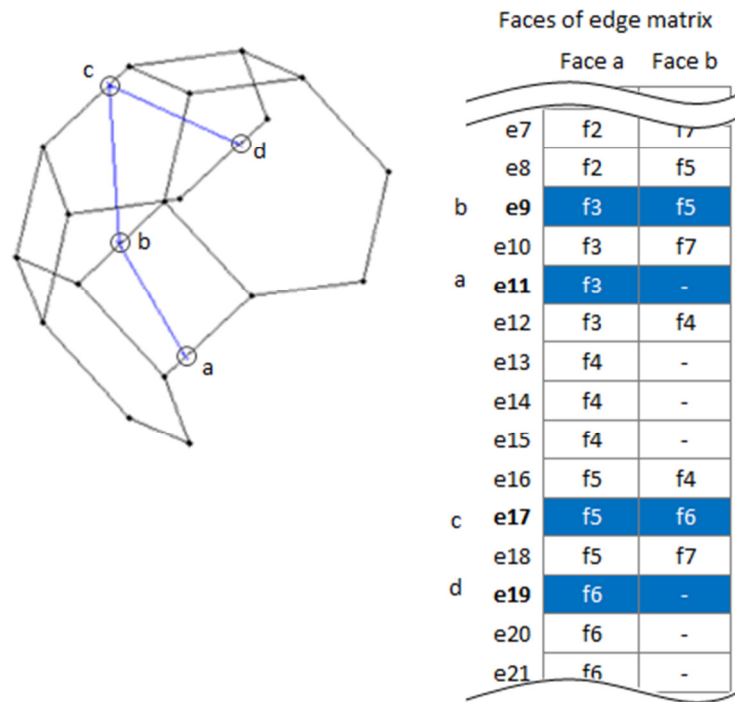


Figure 6-24: Using the partial polyhedron definition matrices to determine which intersection points connect

According to the 'faces of edge' matrix, e11 (the edge with intersection point 'a' on) is only a part of face 3 (f3). The edge with intersection point 'b' (e9) is shared by two faces - f3 and f5. Because f3 is identified to be shared by both e9 and e11, then the two intersection points associated with these two edges can be joined to form a net skin strut. This process is followed for each of the four edges:

- e11 and e9 are both on f3, therefore 'a' and 'b' connect
- e9 and e17 are both on f5, therefore 'b' and 'c' connect
- e17 and e19 are both on f6, therefore 'c' and 'd' connect

This method of representing the base tessellation stores geometric information much more efficiently than the implementation in the basic method. No net skin struts are duplicated across the tessellation and within each polyhedron, no vertices or edges are duplicated. There is still some

duplication of vertices and edges between neighbours within the tessellation but this is minimised. The reduction in duplication improves the overall speed of the process as less geometry must be checked for intersections and matrix size is reduced.

### 6.4.3 Reduction of Intersection Checking

For each point of the base tessellation that is checked by the basic method, every single point of the conformal shape was also checked. For a conformal shape comprised of 'n' points, and a structure comprised of 'm' points, the total number of intersection checks is 'm x n'. Additionally, the basic method presented no method to automatically populate the conformal shape with base tessellation. For any conformal shape imported (as an FE mesh), the first step of the advanced method is to populate a cuboid envelope around the shape with 'seeding points'. These seeding points are equally spaced and represent the centres of the polyhedra comprising the base tessellation, as shown in Figure 6-25a.

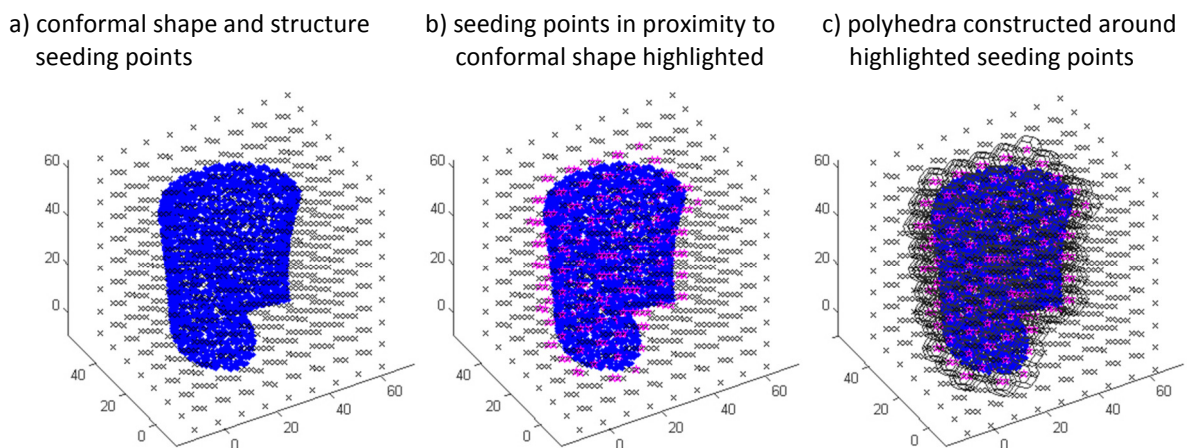
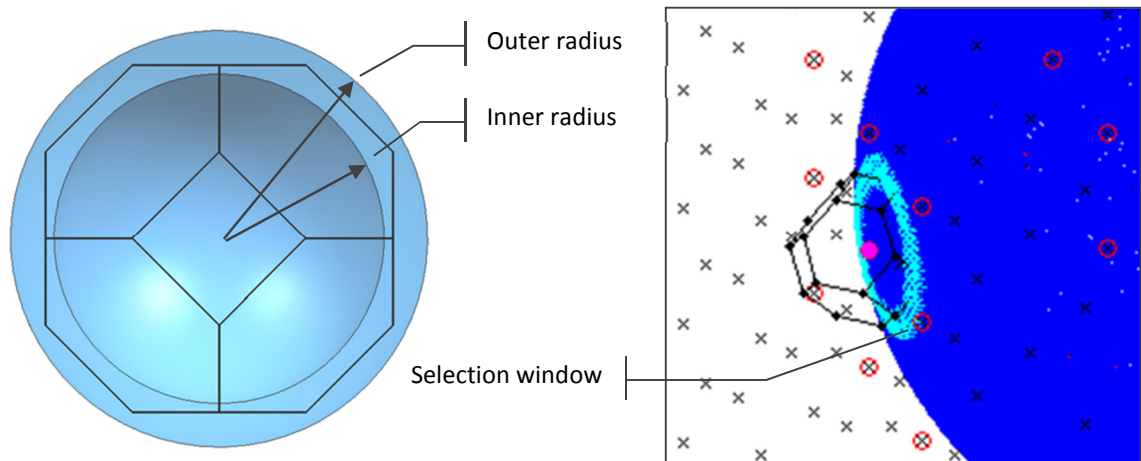


Figure 6-25: Constructing base tessellation to fit a conformal shape

The distance between each seeding point and the surface points of the conformal shape are calculated and those found to be within half the width of the polyhedron are identified, as shown in Figure 6-25b. Polyhedra are then constructed from these highlighted seeding points, as shown in (c). This generates a base tessellation solely of polyhedra that are likely to intersect with the conformal shape, drastically reducing the number of intersection calculations to be required. The partial polyhedron discussed in the previous section is implemented here.

Without further modification to the method, every structure point would still be checked for proximity to every conformal shape point. The advanced method greatly reduces the number of intersection checks required for each polyhedron by first identifying a small 'selection window' of

conformal shape points for each polyhedron. For each seeding point, conformal shape points within an 'outer radius' are selected. Points in that selection closer to the seeding point than an 'inner radius' are discarded from the selection. This constructs a window of conformal shape points that are likely to intersect with the points attributed to a particular polyhedron. This concept is illustrated in Figure 6-26.



Examples of selection windows for polyhedra in different locations:

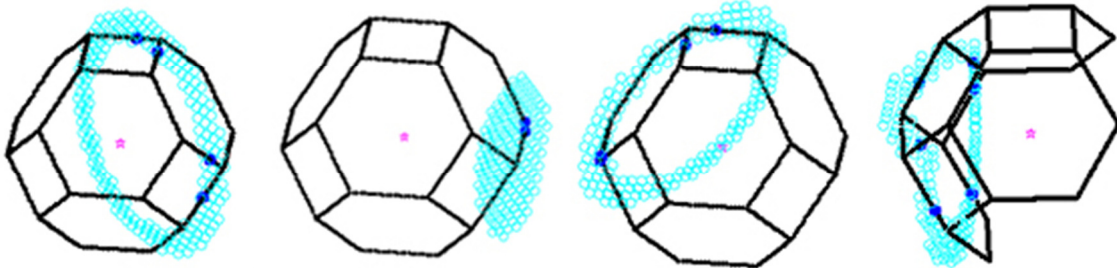


Figure 6-26: Identification of selection windows for individual polyhedra

Rather than checking every surface point against every set of edge points, this small 'selection window' of points is identified for use with each polyhedron. This means that the entire conformal shape is checked once per polyhedron, as opposed to for every point of every edge of the base tessellation. The same small window is used for every edge of a particular polyhedron and a different window is defined for every other polyhedron. This greatly reduces the number of intersection checks required between tessellation and conformal shape, as shown with the following example.

Take a theoretical conformal shape (of 100,000 surface points) and assume that there are 25 polyhedra intersecting it. Using the partial unit polyhedron discussed in the previous section:

Number of conformal shape points  $s_p = 100,000$

Base tessellation:

Number of polyhedra  $p = 25$

Edges per polyhedron  $p_e = 24$

Points per edge  $e_p = 20$

Basic method:

Number of points in base tessellation  $t_p = p \times p_e \times e_p = 25 \times 24 \times 20 = 12,000$

**Total number of checks required**  $c_b = s_p \times t_p = 100,000 \times 12,000 = 1.2 \times 10^9$

Advanced method:

Approx. no. points per window\*  $w_p = 200$

No. points per polyhedron  $p_p = p_e \times e_p = 24 \times 20 = 480$

No. checks required to calculate windows  $c_w = s_p \times p_p = 100,000 \times 25 = 2.5 \times 10^6$

**Total number of checks required**  $c_a = w_p \times p_p \times p + c_w = 200 \times 480 \times 25 + 2.5 \times 10^6$   
 $= 4.9 \times 10^6$

Efficiency:

**Total number of checks (basic / advanced) =  $1.2 \times 10^9 / 4.9 \times 10^6 = 245$  times fewer checks**

\*Number will vary depending on conformal shape and location of polyhedron relative to that shape

For a conformal surface comprised of 100,000 surface points, the advanced method requires 245 times fewer checks than the basic method. This relationship is shown in Figure 6-27, the ratio of 'fewer checks' calculated from varying the number of conformal shape points.

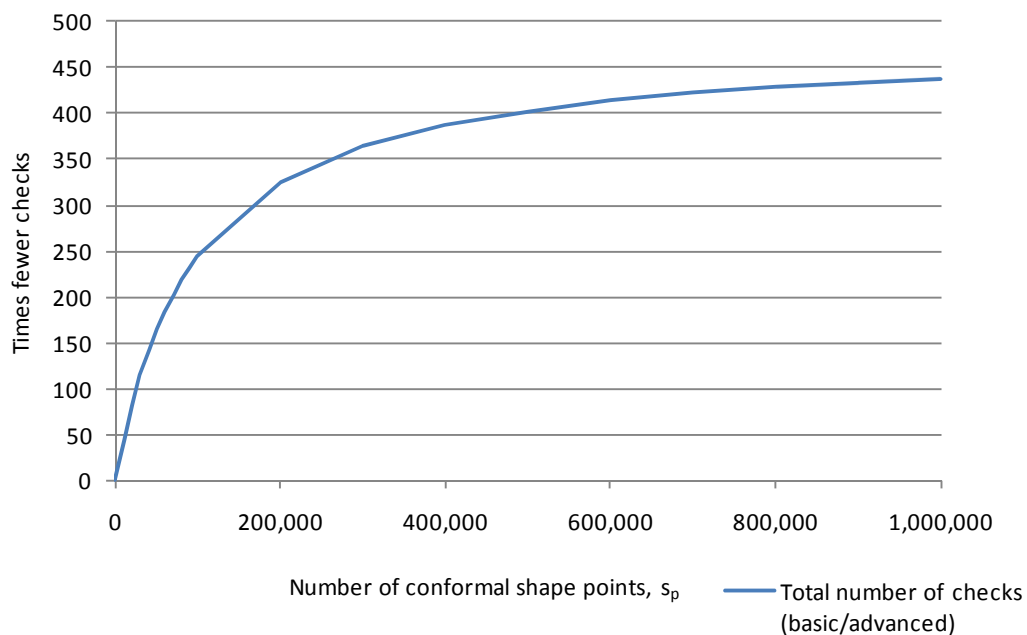


Figure 6-27: Number of times fewer checks required by advanced method compared to basic method

By the instance the conformal shape is represented by a million points, the advanced method performs almost 440 times fewer checks for intersections than the basic method. This is because the number of points generated by the advanced method is not as dependent on total number of conformal shape points as the basic method.

## 6.5 Discussion

This chapter has presented a way to skin a trimmed structure with a 'net skin' - an alternative method to re-connect the cut struts of a trimmed structure. The method utilised what has been termed a 'sampled' boundary representation technique in an attempt to speed up the process of generating geometry. In terms of surface complexity, boundary representation types as discussed in Chapter 2 can be classed as 'advanced' (as used in CAD software) or 'faceted' (e.g., the STL format). What complexity is lost in the faceted type is offset by reduced memory requirements and thus an increase in the size of lattice structures that can be represented. This sampled method takes boundary representation a step further, by reducing all input geometry to a series of points.

Before the method was fully investigated it became clear that it was still not a particularly suitable method for representing large structures. Despite the improvements made in the advanced implementation discussed in the previous section, the time taken to generate even the intersection points used as a basis for constructing net skin geometry became impractically long.

A more fundamental issue arose that further impeded progress with the method. The reduction of geometry to points requires that all intersections between conformal shape and tessellation points are checked to a tolerance, as discussed in Section 6.3.2. This is because it is very unlikely that conformal shape and tessellation points will match exactly. The conformal shape is converted to a series of evenly spaced points for use with the method - the advanced method uses an FE meshing algorithm to achieve this. However, for any given conformal shape there is no guarantee that an even mesh can be generated across it, an example shown in Figure 6-28.

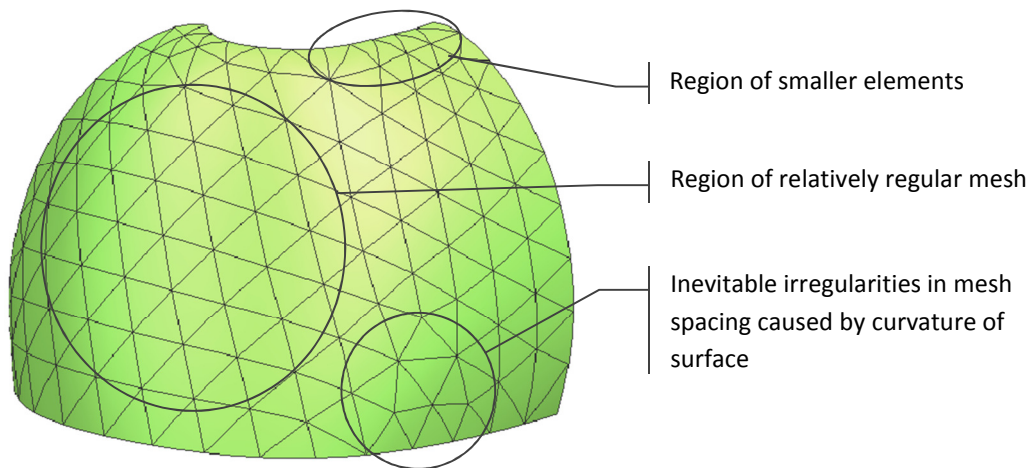


Figure 6-28: Irregularities in pattern of a surface mesh

As stated in Section 6.3.2, a single tolerance value is used to check for intersections between conformal shape and tessellation points. The tolerance is used to check for proximity between these points rather than exact matches. For conformal shape points generated from an uneven mesh, a proximity tolerance suitable for one region may not be for every other. For example, a tolerance that accurately picks the closest conformal shape point in the region of smaller elements (shown in Figure 6-28) may not yield any match in a region of larger elements. This is shown in Figure 6-29: the tolerance suitable for finding a match for 'edge b' does not yield a match for 'edge a', where the conformal shape points are more spread out.

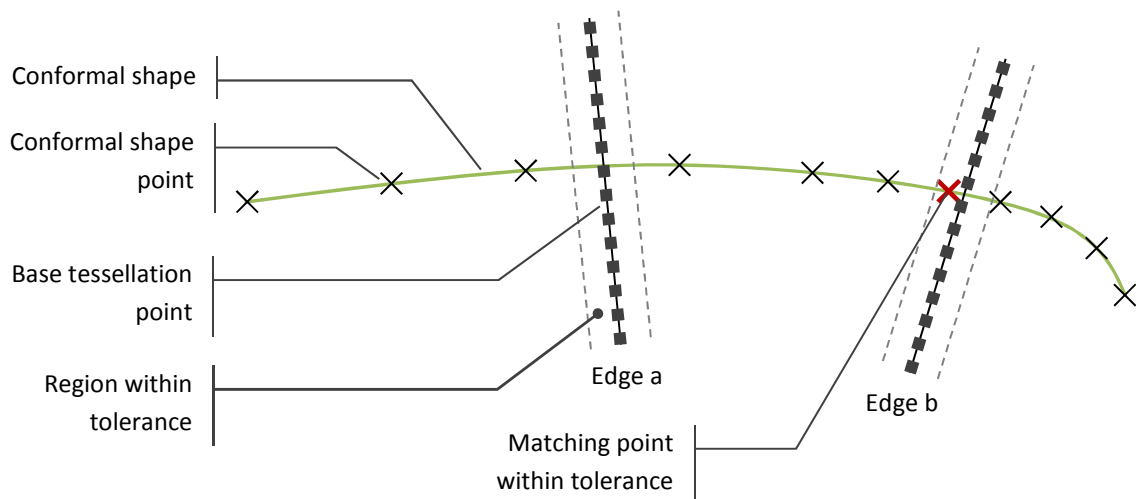


Figure 6-29: An intersection tolerance that is suitable for a region of fine mesh may not be for a coarser region

For this reason, it is not possible to check that a complete and correct net skin has been generated. The method cannot guarantee that any combination of FE mesh resolution and intersection tolerance will find all intersections between conformal shape and base tessellation.

## 6.6 Summary

This chapter has defined an alternative method to skin a trimmed structure to either a solid skin or no skin, termed a 'net skin'. The work in this chapter implemented a Matlab-based method to realise it. The method presented throughout this chapter was defined as a sampled boundary representation method. This 'sampling' was achieved by decomposing continuous B-rep surfaces into two sets of points and checking for pairs of points in close proximity. This was with the aim of developing a less accurate but faster alternative to standard B-rep modelling. Despite the changes, this method is ultimately not a robust way to generate a net skin, nor particularly fast. The preliminary work discussed in this chapter was dropped in favour of another approach to geometric modelling that was being investigated in parallel. This approach is voxel-based which, as discussed in the literature, shows greater promise in the construction of hierarchically complex lattice structures. This work is presented in the following chapter.

## 7 | The Development of a Conformal Structure Method

The findings of the literature review have shown that conventional CAD is not optimised for the generation and manipulation of large arrays of structures, to the extent that alternate methods of structure generation have been developed specifically for the purpose of bypassing conventional CAD entirely. A 'conventional route' of data flow from design to additive manufacture exists: a CAD model is converted to an STL file, which is then converted to a slice file that inputs into the AM process. Structure generation methods have been developed that construct geometry at each of these steps, as well as some methods that avoid this route altogether. Such alternate methods of geometry creation include voxel modelling and function representation.

The aim of this chapter is to present a novel method that has been developed to generate trimmed structures quickly and in a robust manner. The method is based on voxel modelling and as such, has the ability to generate structures with geometrically complex strut types. Its speed is largely independent of structure complexity. The method is capable of processing any repeating structure and a method to functionally grade a conformal structure is also presented. On top of this, the process also generates a net skin. The process has been written in Matlab and the code can be found in Appendix I.

### 7.1 The Conformal Structure Method

This section details the step-by-step process the conformal structure method takes to generate a trimmed, skinned structure from an input shape, termed the 'conformal shape'. The conformal structure method detailed in this chapter can be considered as a post-processing step of a CAD model. The conformal shape is modelled in a conventional CAD package and input into the structure trimming process which populates the conformal shape with structure. A solid skin can also be supplied by the user as a secondary input if either a solid or net skin is required. The solid skin is generated by hollowing the conformal shape input, a relatively straightforward task for conventional CAD or STL manipulation software. This allows the user to utilise the strengths of the CAD package (such as freeform, parametric geometry creation) whilst removing the dependence on it for areas



where it is not optimised (generating large arrays), as discussed in Chapter 2. The net skin is generated separately, but in parallel to the trimmed structure and combined in a subsequent step by this method. A general overview of this method is shown in Figure 7-2.

The conformal structure method presents several significant advantages over conventional CAD modelling. When compared to conventional CAD, complex structure creation is faster with this method. It is also largely independent of structure complexity, which is presented in this chapter with the trimming of two structure types. These two structure types are shown in Figure 7-1. A straight strut structure and a more geometrically complex helical strut structure, comprised of spring-like struts.

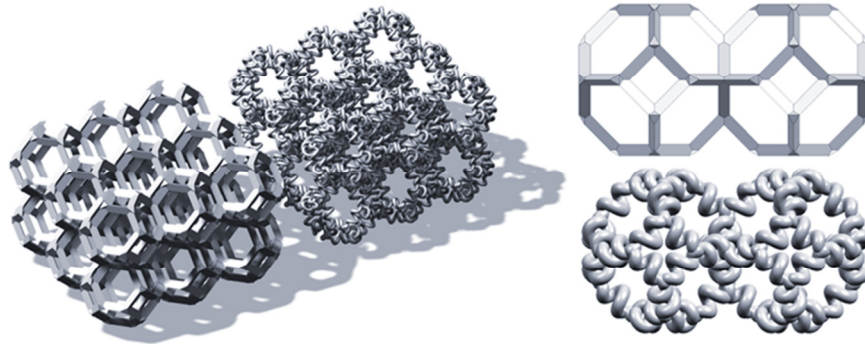


Figure 7-1: Straight strut (with triangular cross section) and helical strut structures

Both of these structure types are based on the Kelvin cell detailed in the previous chapter, but it should be noted that this process will function with any 3D tessellation supplied to it.

### 7.1.1 Importing a CAD Model for Processing

To generate a trimmed structure, the conformal shape must first be modelled in CAD. The model must be solid: a 'watertight' boundary with no gaps between faces. As discussed in Chapter 2, the *de facto* standard file format used in the translation between CAD file formats and additive manufacturing machines is the STL file - a boundary representation of the model composed of triangular polygons.

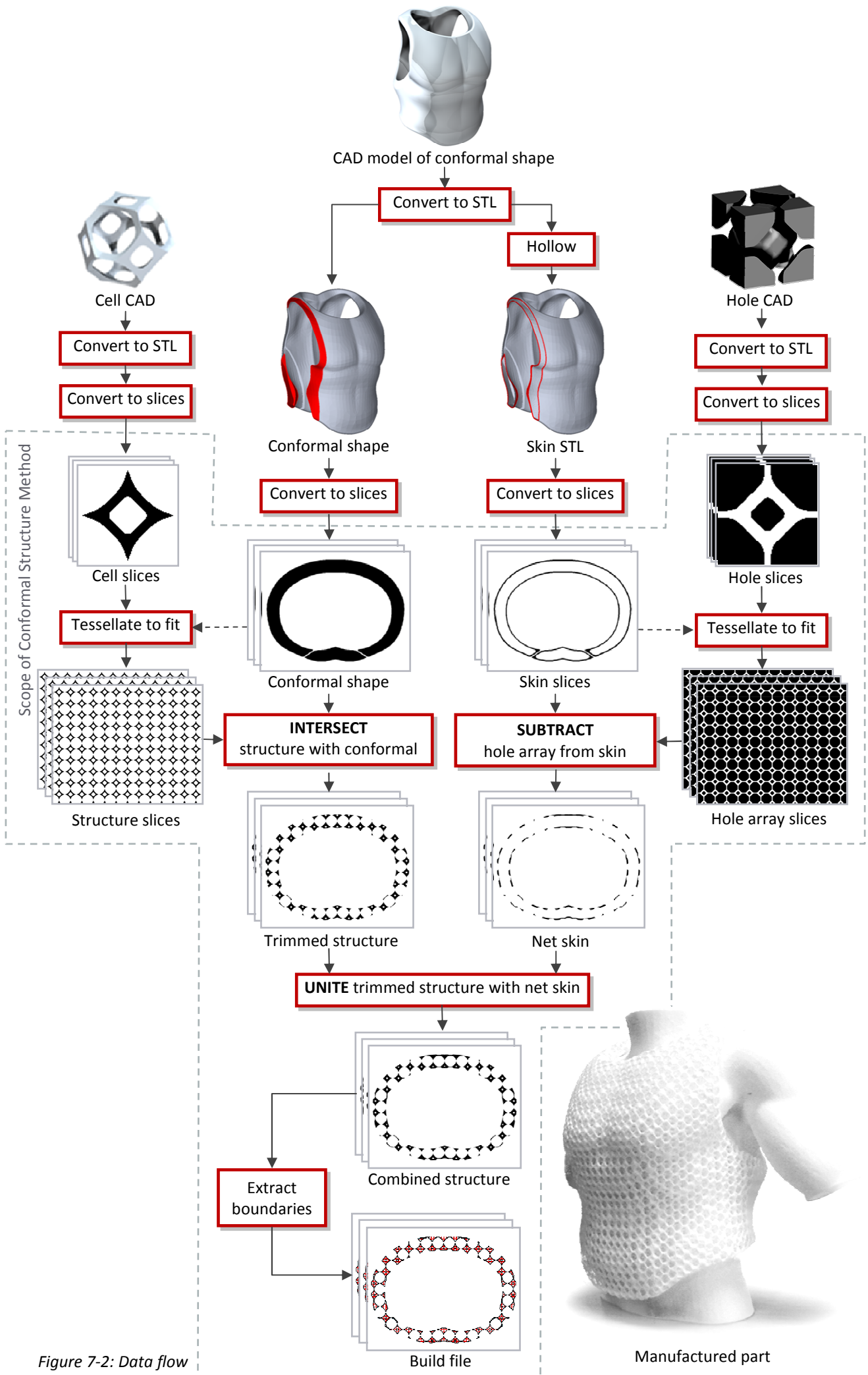


Figure 7-2: Data flow

Each polygon is described with a normal vector - by assessing the normals of each of the polygons that comprise an STL body, interior and exteriors can be determined. Examples of a CAD model and an STL representation are shown in Figure 7-3.

As additive manufacturing machines build parts layer by layer, the STL file is converted to a 'slice file'. A slice file is essentially a layer-by-layer representation of the part; cross-sections of the part taken at increments equal to the layer thickness of the machine, as shown in Figure 7-3. Depending on the AM process, a slice file can take one of two broad formats: a raster-based slice file or vector-based slice file. The structure trimming process deviates from convention here by requiring raster-based slice files regardless of the AM process used to fabricate the parts.

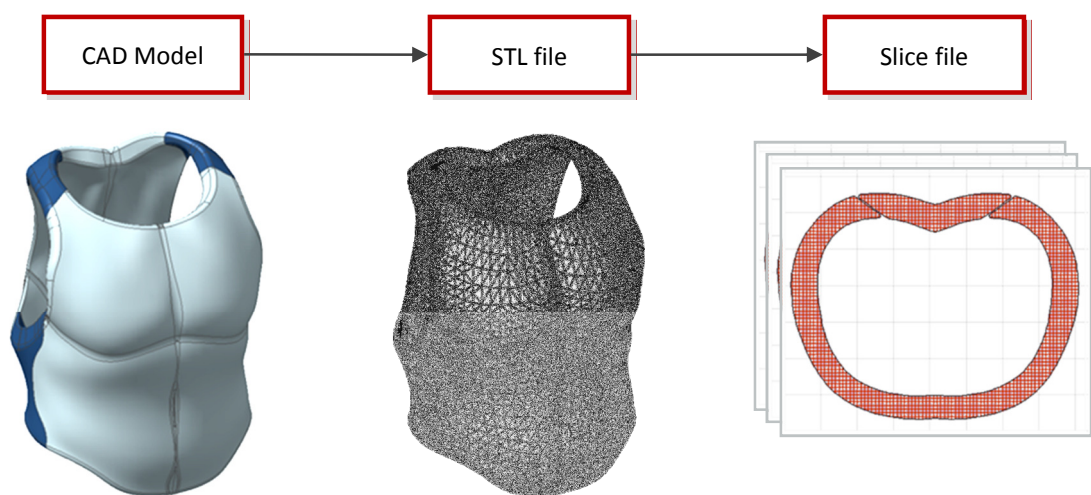


Figure 7-3: The conventional flow of data from modelling to manufacture

The actual slicing process is performed by a freeware command line utility called 'Slice' [166], although the process is not dependent on any particular slicer. The slicer outputs a series of black and white bitmaps - by convention black signifies solid model while white signifies empty void. As the bitmaps only consist of black or white pixels, the boundary of the model is represented by a stepped profile - the closest approximation that a binary bitmap can make of a smooth profile. This 'pixel-stepping' is shown in Figure 7-4. When trimming a structure for manufacture by laser sintering, one raster file represents a 0.1mm build layer. With the conformal shape converted to a suitable format, the structure trimming process can begin.

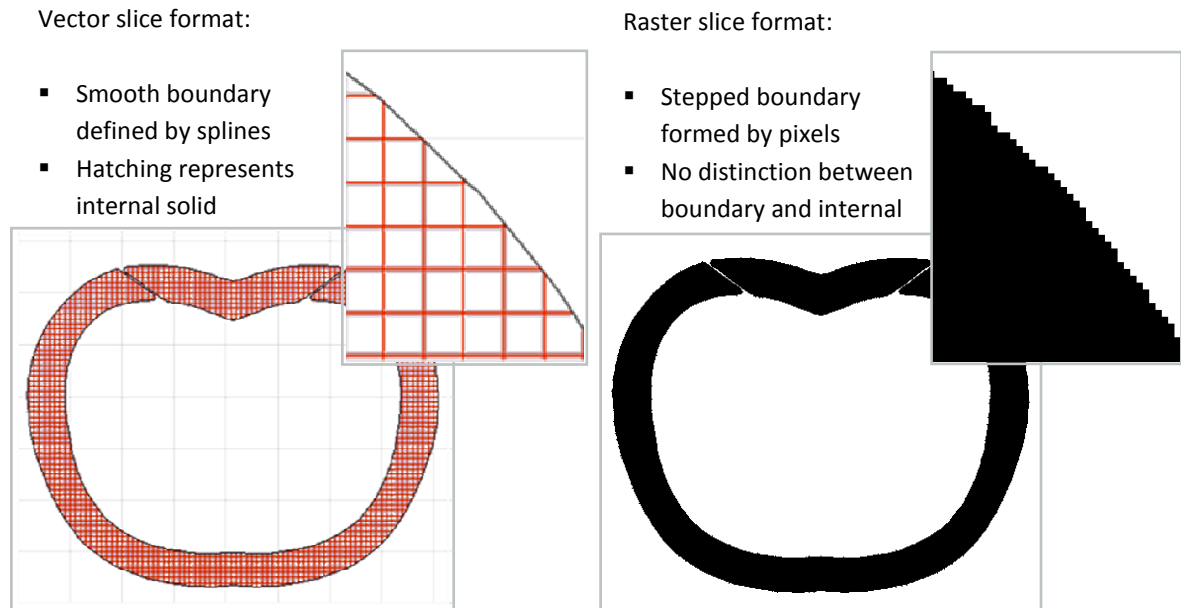


Figure 7-4: A single layer of a vector and raster slice file

### 7.1.2 Trimming Structure to a Conformal Shape

The entire method is structured in a way that a single layer of structure is generated and written to an output file at a time. This applies to both elements of the structure - the internal structure and net skin are generated for the first slice, combined and written before the second slice is processed. This is achieved through a FOR loop: for this application, the number of times the process must be completed is equal to the total number of layers that comprise the conformal shape. The process starts with the bottom layer of the part (layer 1) and once all structure generation code is completed, the FOR loop executes the code again for layer 2. A summary of this process loop is shown in Figure 7-5.

```

Select conformal shape and cell type
for current layer (increment from first layer to last layer)
    read conformal shape and cell slice
    generate internal structure slice
    generate net skin slice
    combine internal structure and net skin slices
    write combined structure to output slice file
end

```

Figure 7-5: Basic code structure of the conformal structure method

During the investigation of this process, a library of cell types were designed. A selection of these are shown in Table 7-1. The cell types have been modelled in CAD and were sliced into a sequence of raster slices in the same manner as the conformal shape. The trimming process is not limited to the specific types shown in Table 7-1; any 3D structure can be processed. Once a new structure type has been sliced it can be added to the structure library. The structure is reduced to a single cell: a repeatable element of the structure that when arrayed orthographically will reconstruct it.




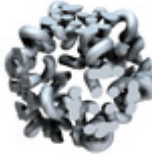










Cell	Kelvin / truncated octahedron				Weaire-Phelan	Cubic	Alternated cubic
Strut type	Triangular cross-section	Cylindrical	'Wave'	Helical	Cylindrical	Helical	Cylindrical
Model view							
Example slice							

Table 7-1: A selection of structure types integrated into the structure trimming process

At this stage, a series of cell and conformal shape raster slices have been generated and numbered to retain their correct order. To trim the structure to the conformal shape, an operation analogous to a Boolean intersection (logical operator:  $\cap$ ) is applied between corresponding conformal shape and cell slices. The bitmap slices are first converted to a binary colour depth (black pixels = 0, white pixels = 1) and read in to Matlab as a matrix, where each element of the matrix represents a pixel of the bitmap. A simplified example is shown in Figure 7-6.

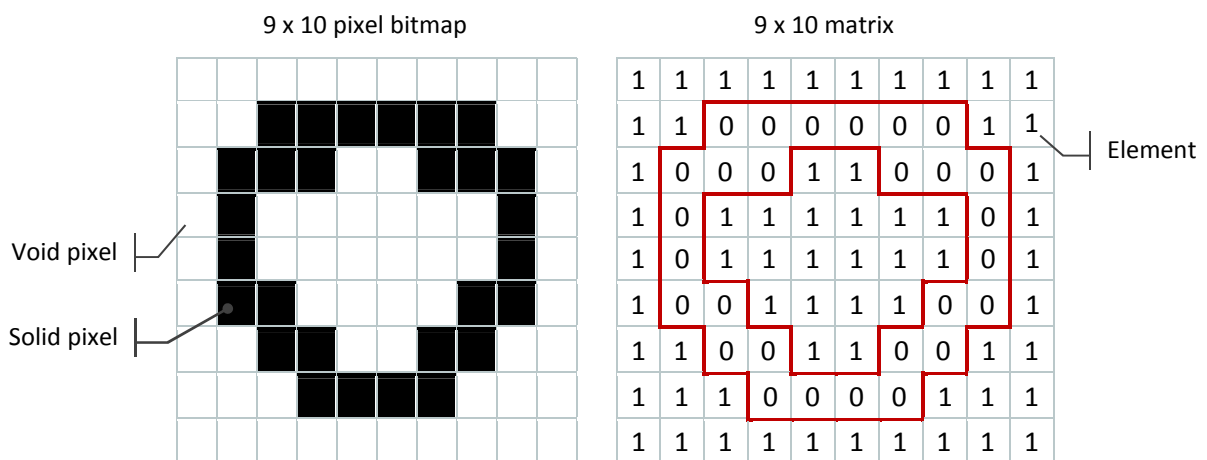


Figure 7-6: Bitmap to matrix conversion of a single slice

The structure trimming process first registers the size of the conformal shape matrix (i.e.: row and column dimensions). An empty matrix is created that is the same size as the conformal shape matrix, termed the 'trimmed structure matrix'. A cell type is selected and the size of the cell matrix is also registered. Starting from the top-left corner, the conformal shape matrix is checked element by element. If the element is a one (i.e. a white pixel/empty space), a value of one is written to the trimmed structure matrix in the corresponding location, as demonstrated with 'check a' in Figure 7-7.

If the element of the conformal shape matrix is a zero (i.e. a black pixel/geometry) then a corresponding element from the cell is copied to the trimmed structure matrix. Two examples of this are shown by 'check b' and 'check c' in Figure 7-7.

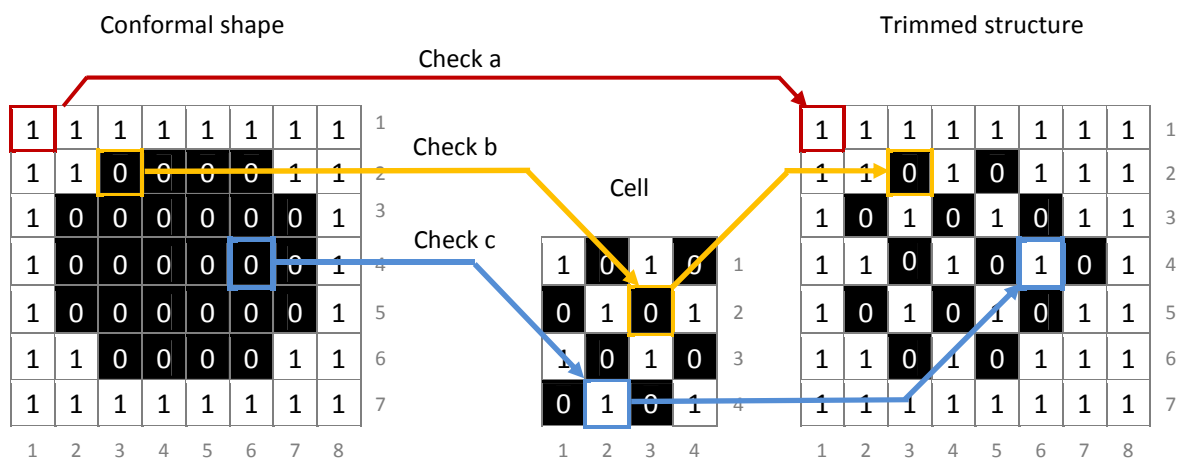


Figure 7-7: Trimming structure to a conformal shape - analogous to a Boolean intersection

The cell matrix is smaller than the conformal shape matrix even in this simple example - in practice the conformal shape can be many times larger than the cell matrix. In order to fit the cell matrix to the conformal shape, it is tessellated to fit it. In this simple example, the top left corners of both the conformal shape (conformal shape(row 1,column 1)) and the cell (cell(1,1)) line up. So when the process identifies a zero value element at conformal shape(2,3) (check b), it copies the value of cell(2,3) to the corresponding location on the trimmed structure matrix.

A counter is employed to repeat the cell matrix across the length (x direction) of the conformal shape matrix. A counter is a function that is set to a particular value that resets when that value is reached. In this case, the counter is set to the width (x direction) of the cell matrix. In this example, the cell matrix is four elements long, so when the process reaches the fifth column of the conformal shape matrix, the counter resets to one and the first element of the cell is copied over. 'Check c' in Figure 7-7 demonstrates this. The location of the element to be checked is conformal shape(4,6). This is beyond the initial position of the 4x4 cell matrix, so the matrix is translated 4 columns to the

right to be in a position where it overlays with the element being checked. This results in the value from cell(4,2) being copied to the corresponding location on the trimmed structure matrix. A second counter is used for the width (y direction) of the cell in a similar manner.

Additionally, the cell needs to be tessellated against the height of the conformal shape. For example, if a cell is 15mm high and the layer thickness is 0.1mm, the cell will be described by 150 matrices. Therefore the first and 151st layers will use the first cell matrix; a third counter increments as each new layer is processed. A more complex example of a layer of trimmed structure generation is shown in Figure 7-8. The process is analogous to a Boolean intersection between the conformal shape and a structure of repeated cells; the result is any solid that occurs on both the conformal shape and the structure.

Boolean intersection:

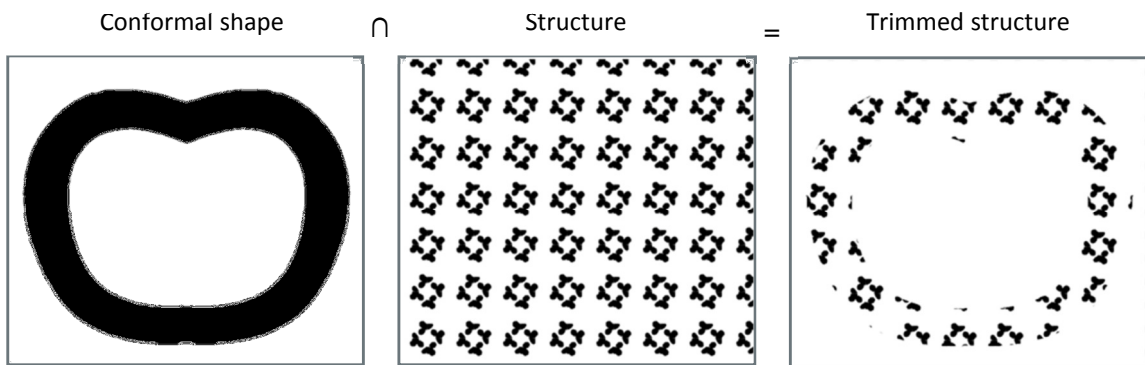


Figure 7-8: Generating trimmed structure slice - analogous to a Boolean intersection

### 7.1.3 Constructing the Net Skin

The net skin is constructed in a parallel process to the trimmed structure and as such requires the combining of the two in a subsequent step. In a similar manner that the trimmed structure step utilises a library of cells, the net skin step of this method also requires a library of repeatable elements. Whereas the trimmed structure step intersects structure with conformal shape, the net skin step subtracts 'holes' from a solid skin. The 'hole cell' is similar to an inverted version of a cell, where the internal cavity of a cell is now a solid volume as shown in Figure 7-9.

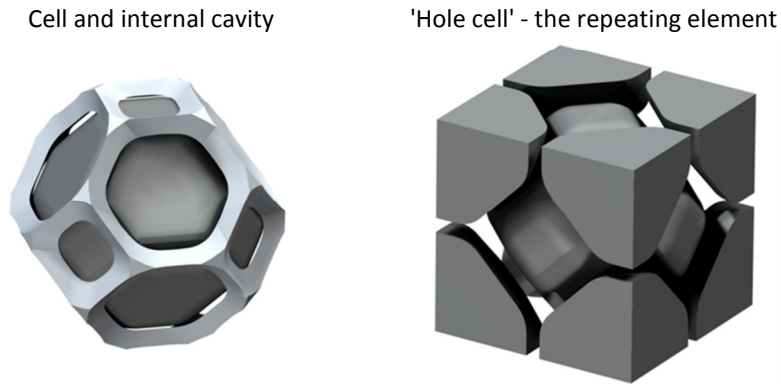


Figure 7-9: Comparison of cell and 'hole'

The analogy to the conformal shape in this step of the conformal structure method is the skin. Figure 7-10 shows a section view of a conformal shape and a solid skin. The solid skin is a hollowed version of the conformal shape, with a wall thickness that is by default set as the strut diameter of the structure.

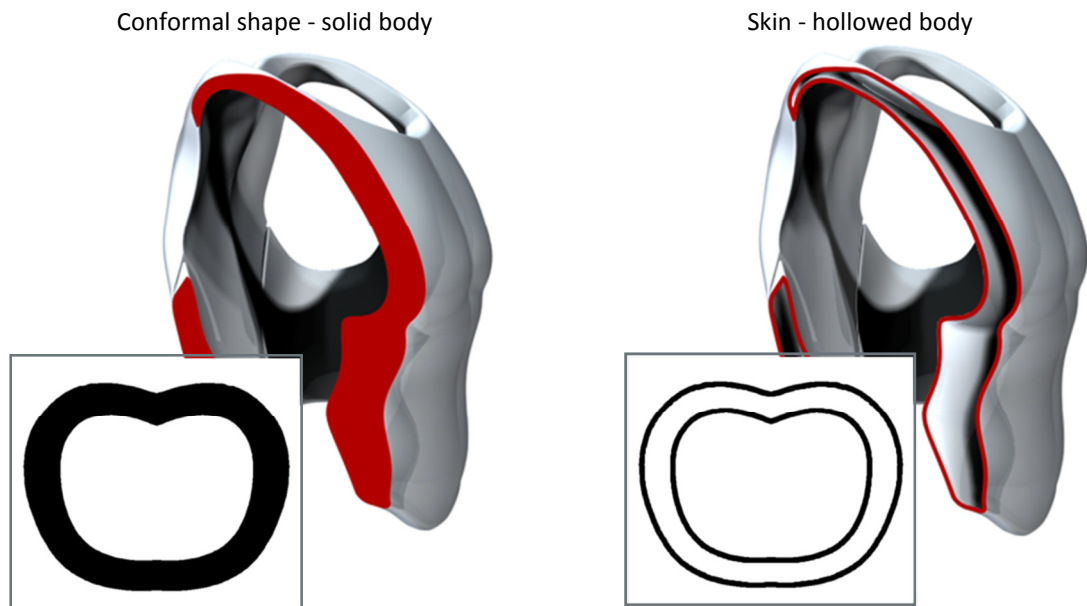


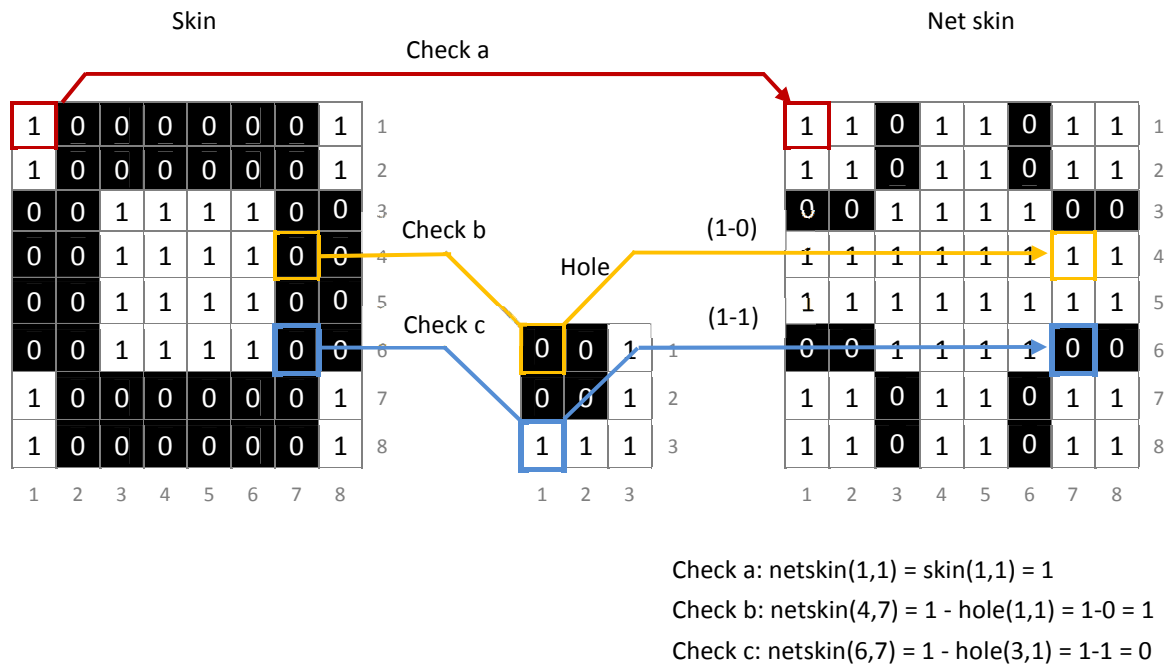
Figure 7-10: Difference between conformal shape and skin

The skin is generated from the conformal shape before slicing - either during the CAD model or STL conversion stages. The hole cell is stored in a library as bitmap slices. The skin is sliced at the same time as the conformal shape and the hole slices are tessellated to fit the skin slices. Where the net skin process differs is how the two sets of slices (matrices) are combined. To generate the net skin, the hole matrix is subtracted from the skin matrix.

An empty matrix is created the same size as the skin matrix which becomes the net skin matrix. The process checks the skin matrix element by element - if the element has a value of one, a one is



written in the corresponding location on the net skin matrix. If the element has a value of zero, the value of one minus the corresponding element from the hole matrix is copied to the net skin matrix, as shown by 'check b' and 'check c' in Figure 7-11. The hole matrix is repositioned as required in the same manner that the internal structure process tessellates the cell matrix.



Boolean subtract:

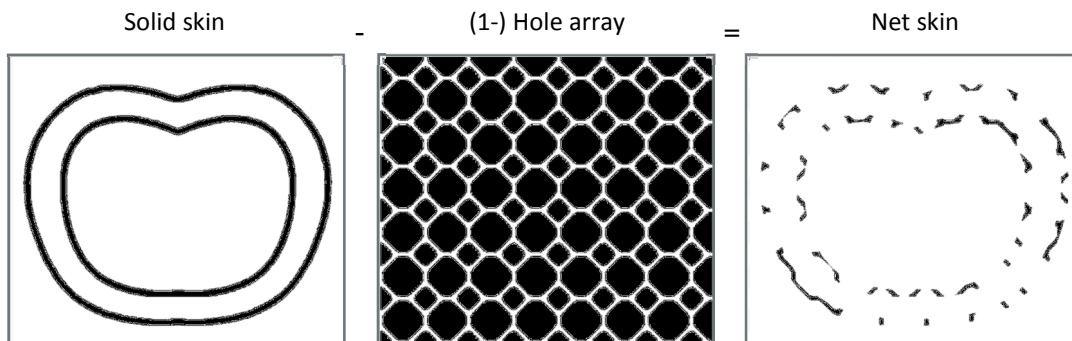


Figure 7-11: Net skin matrix calculation

This particular method of generating a net skin is limited in the type of geometry that can be constructed. For example, this process could not construct a net skin of cylindrical struts. This is because the method does not create individual struts (as the method described in the preliminary work chapter does), rather the struts are what is left from the subtraction operation. The geometry of the net skin that this process constructs is shown in Figure 7-12.

The strut geometry can be controlled to some extent by varying the geometry of the hole cell. The width of the struts is controlled by the diameter of the hole cells and the height of the struts is equal to the wall thickness of the skin.

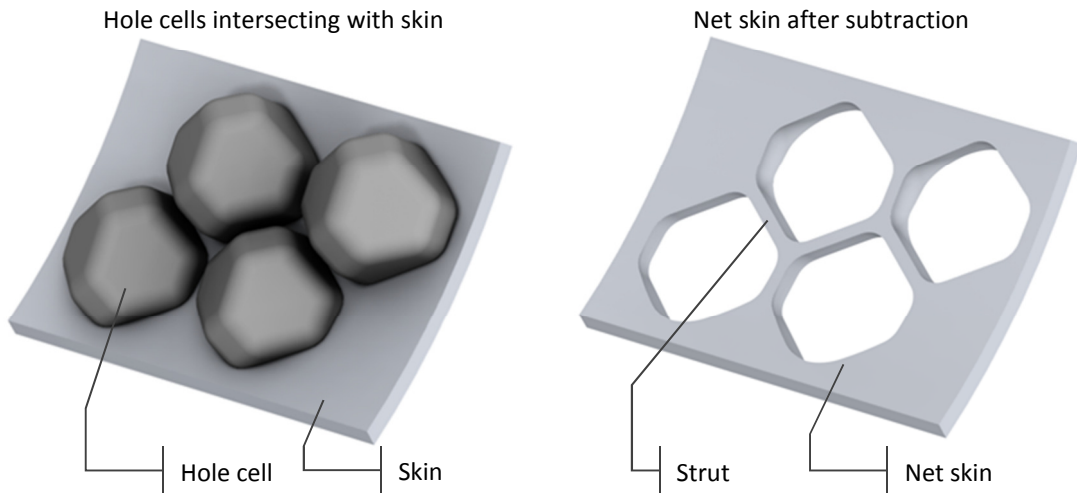


Figure 7-12: Net skin generation by Boolean subtraction

Although the hole cell shown in Figure 7-9 is shown with a particular strut type, the design of it is solely dependent on the cell type selected for trimming to a structure. Regardless of the type of strut geometry trimmed to generate the internal structure (see Table 7-1 for a selection), the hole cell will remain the same for a particular cell type. This is shown in Table 7-2.

Cell type	Kelvin / truncated octahedron			
Strut type	Triangular cross-section	Cylindrical	'Wave'	Helical
Cell				
Hole cell				

Table 7-2: Hole cell selection

A procedure has been further investigated to design the hole cell that corresponds to a particular structure type, illustrated in Figure 7-13. The structure in question (Figure 7-13a) is selected and its base tessellation determined (b). The faces of the polyhedra are offset inwards by half of the required strut diameter (c). An optional step is to apply a small fillet to the edges of the polyhedra (d) - this will have the effect of filleting the holes of the net skin (e) in an effort to prevent cracks from propagating on otherwise sharp corners.

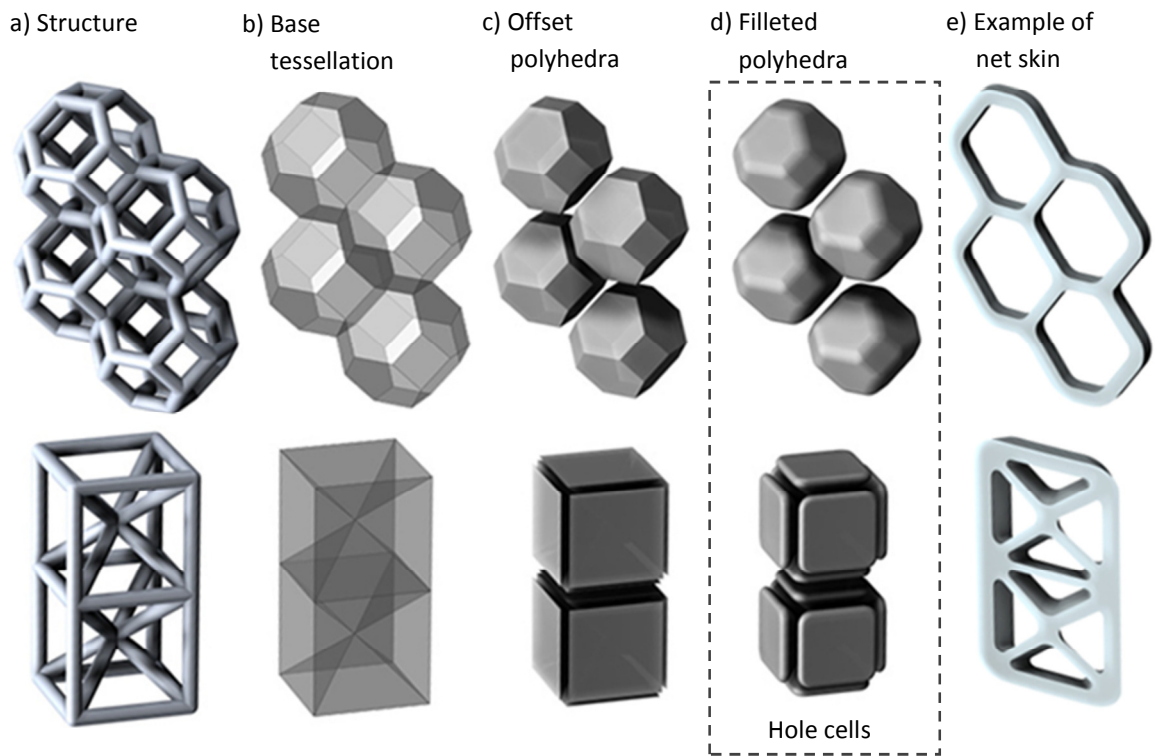


Figure 7-13: Method of generating a 'hole cell' from a given structure

This method of constructing a net skin, while exploiting all the advantages of the conformal structure method, is limited in terms of net skin design. A second disadvantage is that the net skin is not always completely formed. In most cases the hole cell completely intersects the solid skin; when the hole cell is subtracted the hole passes completely through the solid skin. Depending on the position and shape of the solid skin, there may be instances where the subtraction of a hole cell does not pass all the way through, an example shown in Figure 7-14.

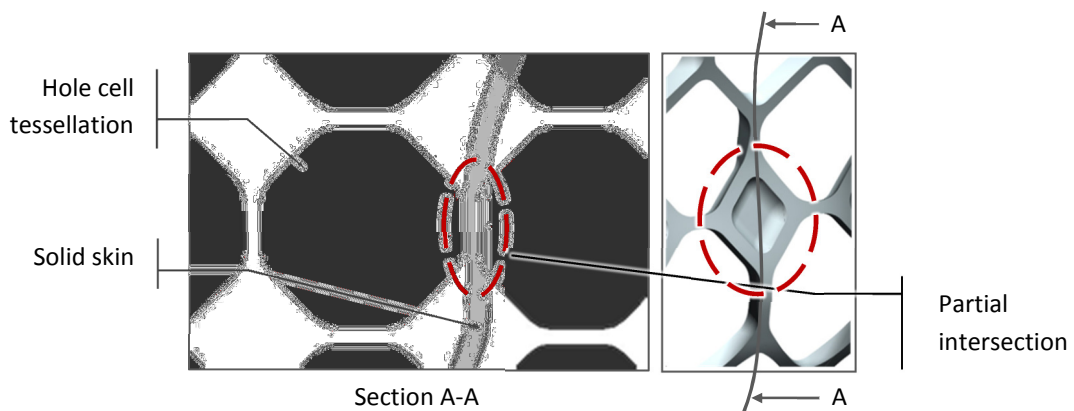


Figure 7-14: Partially formed net skin where hole cell tessellation does not completely intersect solid skin

By reducing the offset value applied to the base tessellation (Figure 7-13b) the likelihood of these partial hole formations can be minimised. However this also has the effect of reducing the overall strut thickness of the net skin. Modifying the design of the hole cell was also investigated with the

addition of extrusions on each of the faces of the hole cell. An example is shown in Figure 7-15. The problem with this approach however is that the same extrusions used to cut through partially formed holes also cut through other correctly generated struts of the net skin.

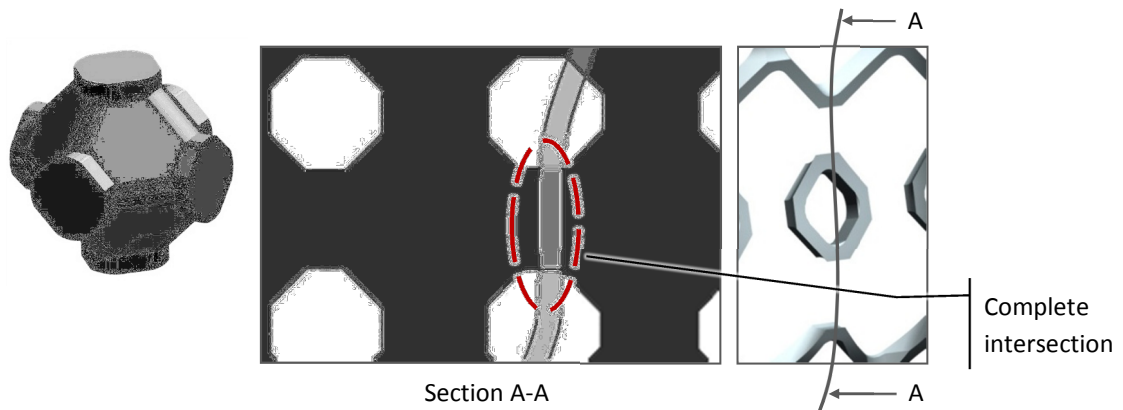


Figure 7-15: Modified hole cell that completely intersects skin in dashed region, although the same modifications on other instances of cell destroys neighbouring struts (compare with Figure 7-14)

As stated at the beginning of this section, the hole cell is similar to an inverted version of the structural cell, although not identical. If an inverted cell is used as a hole cell, a similar result as Figure 7-15 is observed.

#### 7.1.4 Combining Structure Elements

By this point, the trimmed structure and net skin have been separately generated. They exist as a pair of equal size matrices. To combine the two, an IF statement is used as follows: for every element of the net skin matrix, if the value is zero (i.e. black pixel, or solid geometry), convert the corresponding element of the trimmed structure matrix to zero. This overlays the net skin matrix on top of the trimmed structure, essentially uniting the two to form a combined structure matrix. This is shown in Figure 7-16.

Boolean unite:

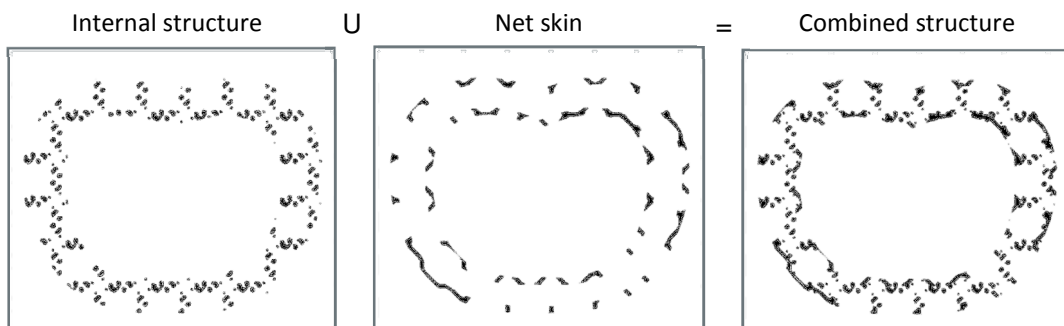


Figure 7-16: Combining structural elements

### 7.1.5 Conversion to Slice File

The combined structure matrix represents the complete trimmed structure, which is compatible with AM processes that utilise raster slices, such as 3D printing. However to be suitable for AM processes that require vector-based slice files (such as laser sintering) the matrices of ones and zeroes must be converted to a more meaningful format. Essentially, to convert each matrix to a vector slice file, the outlines of the shapes represented by the matrix are traced. This is depicted on a small section of an individual strut in Figure 7-17.

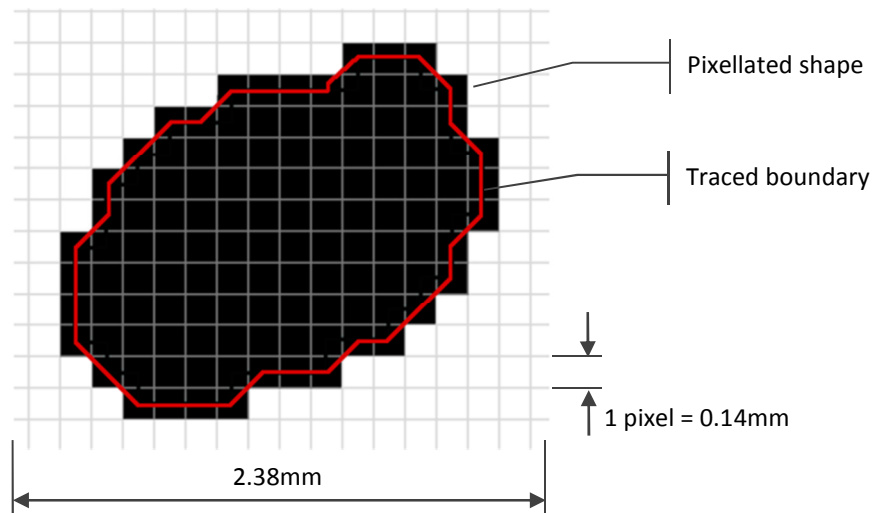


Figure 7-17: Tracing boundary of pixellated shape

The method to trace boundaries already exists in Matlab - a built-in function called 'bwboundaries'. The method is an image processing tool and works by scanning a bitmap matrix for a boundary pixel, i.e. a black pixel with white neighbours. When a boundary pixel has been located, the eight neighbouring pixels are checked under the same criteria and from this a string of ordered boundary pixels are generated. Due to this eight-neighbour search, diagonal lines are constructed at pixel corners, as shown in Figure 7-17. This means that a 45° line of pixels will generate as smooth a boundary as a 0° line. Separate strings are generated for each individual shape on the bitmap. Each boundary pixel is defined by its row and column co-ordinates; these co-ordinates are multiplied by a scaling factor to convert to millimetre measurements. This scaling factor is dependent on the resolution of the bitmap slices input into the methodology: by default, one pixel equals 0.14mm.

Each boundary string is written as a 'polyline' to a Common Layer Interface (CLI) file - a type of slice file compatible with laser sintering machines [103]. There are two types of CLI file - ASCII and binary. ASCII CLI files are structured with a header section at the start of the file, followed by a geometry section that is split into layers, as depicted in Figure 7-18. This structure makes the integration of writing a CLI file into the process relatively straightforward because - as discussed previously - the conformal structure method works layer by layer. As a layer of trimmed structure is

generated, the boundaries for that layer are written to the CLI file and it is appended with new data as each layer is generated.

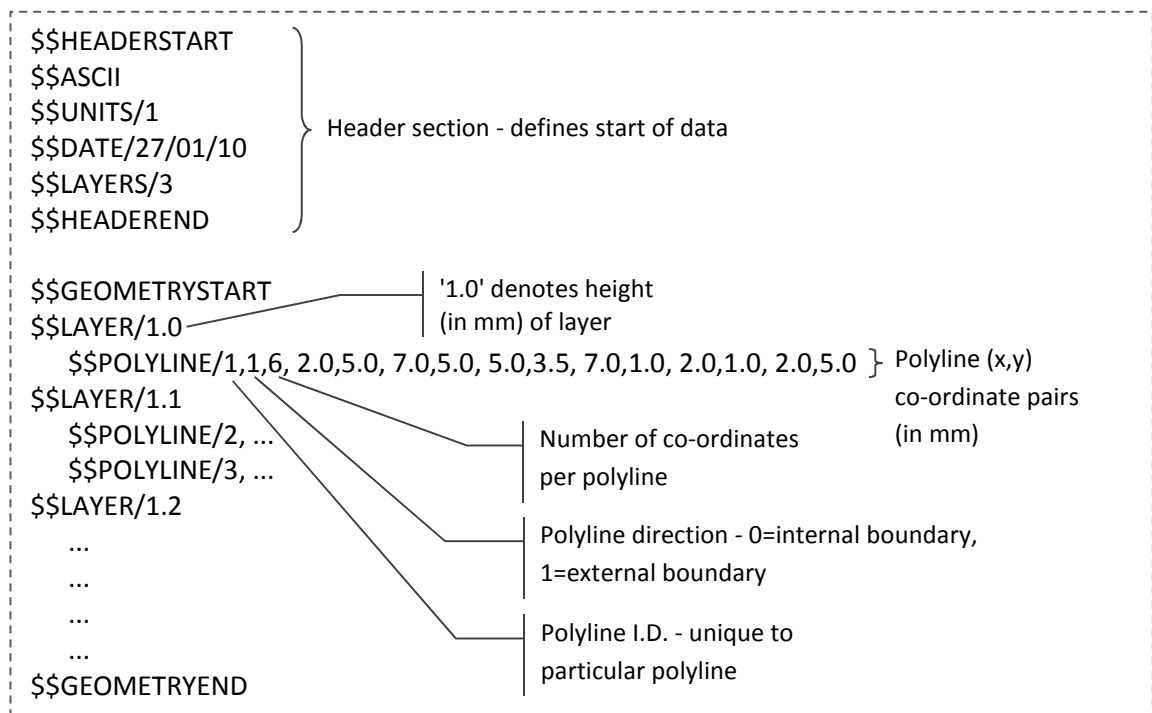


Figure 7-18: Structure of a Common Layer Interface (CLI) file

This section has detailed how by taking a part modelled in CAD and converting it to a bitmap slice format, a conformal structure can be generated. The next section develops this concept further by detailing a method to create functionally graded structures - structures where the strut geometry can be varied across them. Subsequent sections detail further advances of the conformal structure method.

## 7.2 Functional Grading

The potential of functionally graded structures is to be able to design optimised structures with different properties in particular regions, determined by a set of loading and boundary conditions. This can be achieved by varying strut geometry across a part. While the conformal structure method is not at the stage where optimised structures are possible, this section details the groundwork that has been completed to make it a possibility.

The conformal structure method described in the previous section considers matrices of zeroes and ones to represent solid and void respectively. This is analogous to the black and white bitmaps used to describe the process: where a conformal shape is black, a particular structure is constructed. The conformal structure method has the capability to generate functionally graded structures by

overlaying a greyscale image on to the conformal shape. A simple example would be to overlay a gradient, as demonstrated in Figure 7-19.

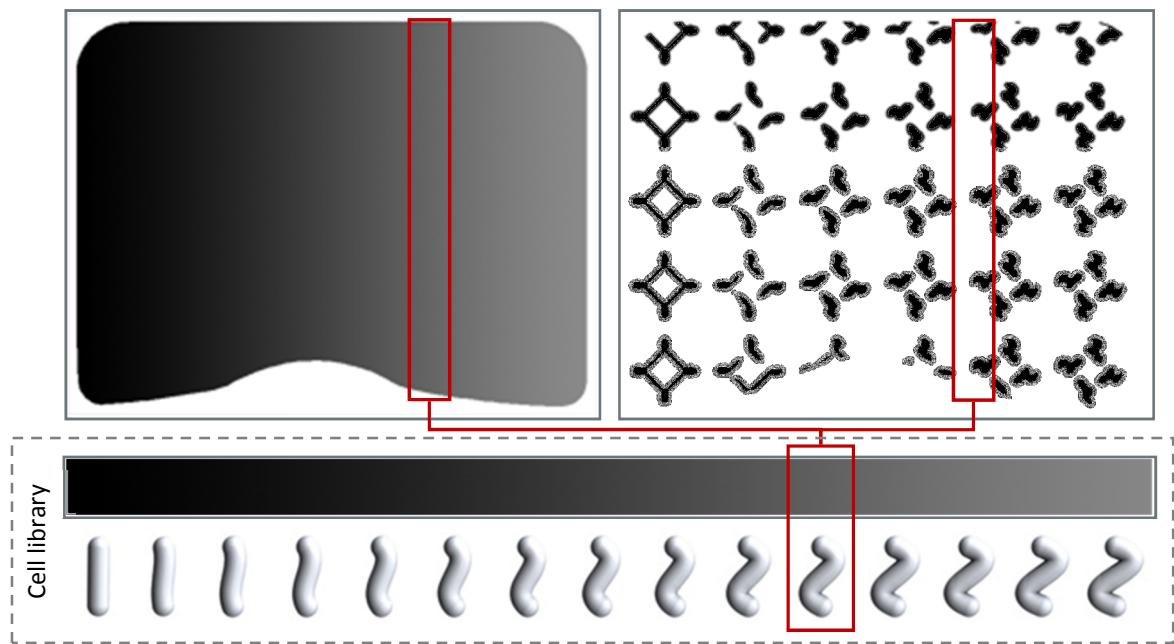


Figure 7-19: Functionally grading a conformal shape

Rather than the binary reasoning of 'if black, construct structure', the process becomes 'if a pixel is a particular shade of grey, construct a particular increment of structure'. A gradient is depicted with an 8-bit greyscale image that is overlaid over each conformal shape slice with a Boolean intersection (to allow the assignment of different structure types to a range of values), the resultant conformal shape slice is also a greyscale, 8-bit bitmap.

To be able to generate a functionally graded structure, a library of cells is required that increment from one end of the graded range to another. The example in Figure 7-19 grades from a straight strut to a wave-like strut design. Between the two extremes are wave-like struts that gradually reduce in amplitude to zero. Another simple example of a suitable range would be to grade from a thin strut of a particular design to a thicker variant, as shown in Figure 7-20. As before, the cells have been modelled in conventional CAD software and converted to a series of bitmap slices.

For the particular example shown in Figure 7-19, there are sixteen increments of the strut design; the difference in amplitude between increments is less than can be resolved by the bitmap resolution (i.e. less than the pixel width: 0.14mm). In practice, any number of design increments could be implemented for functional grading. The overlaid image used to determine the placing of strut increments is an 8-bit bitmap. This is a 256 state greyscale range (values of 0-255), so when divided up into the sixteen increments shown in Figure 7-19, sixteen shades of grey correspond to each strut increment. 8-bit greyscale bitmaps are a standard bitmap format and thus many programmes exist that can generate them.

Many image-manipulation programmes can generate gradients that can be written as 8-bit bitmaps, making the generation of gradients a straightforward task. Any 8-bit bitmap can be input to generate functionally graded structures, a few examples (composed in Adobe Photoshop) are shown in Figure 7-20. The eventual aim is to accept density maps from optimisation software to construct varying structures optimised to a specific purpose, as detailed in Section 9.3 as recommendations for further work.

The functionally graded structures generated by this methodology are varied very smoothly - blending between strut design increments is not constrained to a cell-by-cell or strut-by-strut limit. If the gradient requires that a change in strut increment should occur over the length of a strut, this is what is constructed. This same method is used as the basis for the work completed in the next section.

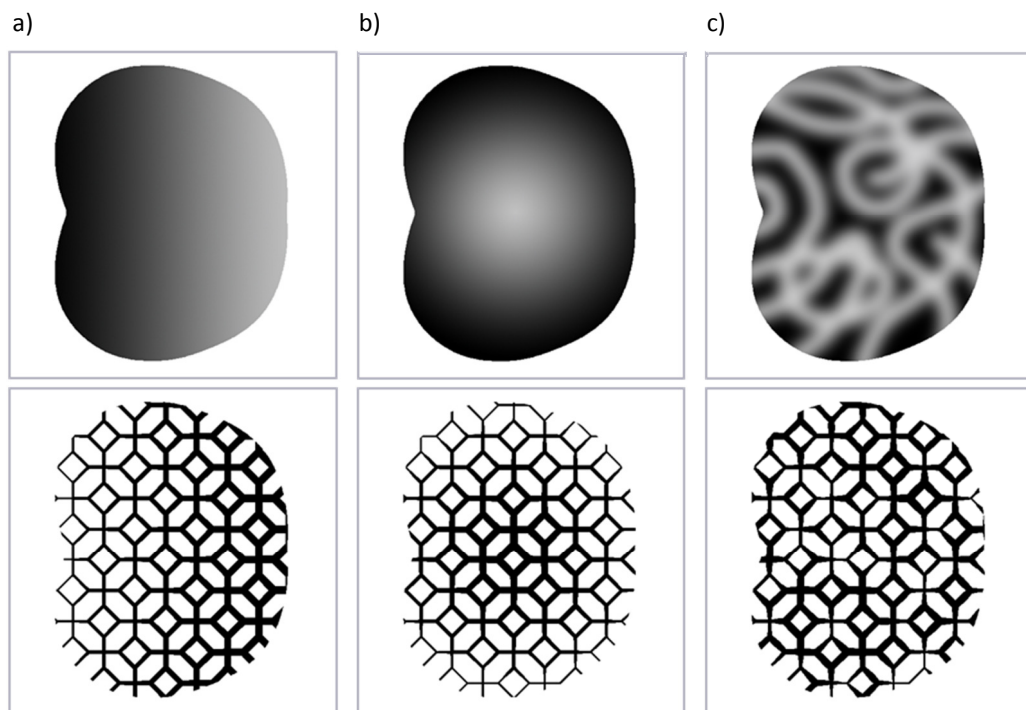


Figure 7-20: Functionally graded structures generated from input gradients

### 7.3 Ensuring Net Skin Connectivity

The conformal structure method generates internal structure and net skin in two separate steps. For the two structural elements to line up, the process ensures that the cell and 'hole cell' are initially positioned in the same place when tessellated over the conformal shape and skin slices. This ensures that the trimmed struts of the internal structure line up with the nodes of the net skin.



Depending on the strut type used on the internal structure, this may not be sufficient to ensure correct connection with the net skin. The geometry of the net skin is determined by the base tessellation of structure, as detailed in Section 7.1.3. So if the structure has been augmented with a helical strut for example, a strut when trimmed will not necessarily line up with the constructed net skin. The helical strut shown in Figure 7-21 illustrates this.

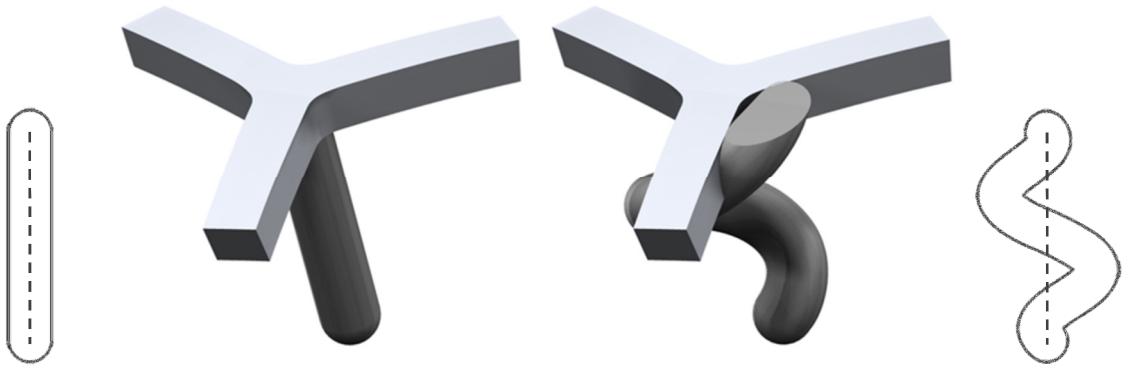


Figure 7-21: Misalignment between a complex strut and the net skin

Utilising the functional grading method developed for the generation of structures with variable geometry, this problem can be overcome. The neatest and most robust way to ensure net skin connectivity is to reduce the strut's helical diameter to zero before the point the strut intersects with the conformal shape boundary. The functional grading approach detailed in the previous section can blend between strut types given a greyscale image. Given the correct greyscale image, this approach can be used to ensure net skin connectivity. A graded outline that follows the boundary of the conformal shape will achieve this, as shown in Figure 7-22.

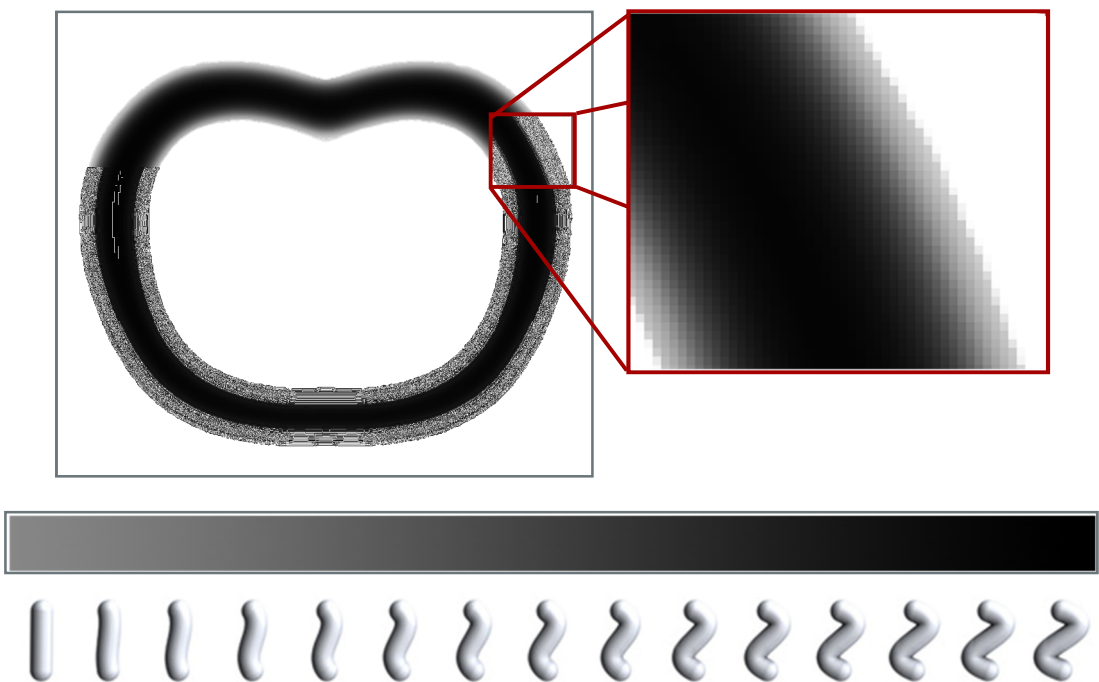
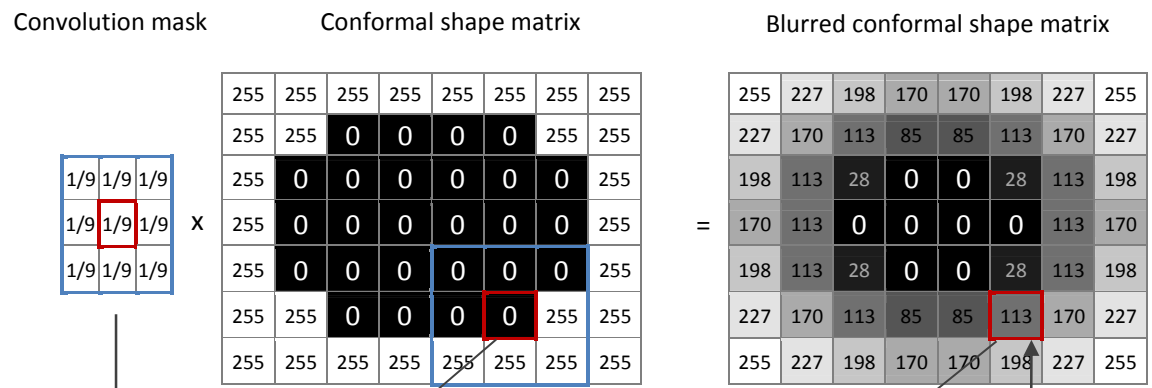


Figure 7-22: A graded boundary

For any given conformal shape, this graded outline can be generated automatically, which is achieved by blurring the image. The entire image is blurred, but because there is only contrast between black and white at the conformal shape boundary, the result is that only the boundary appears blurred.

Each conformal shape is blurred through 2D convolution - the conformal shape slice is again considered as a matrix rather than an image for this process. The conformal shape matrix is convolved with a mask matrix, simple examples of which are shown in Figure 7-23a. The mask matrix is a square matrix that contains a set of multipliers. The example in Figure 7-23a shows a 3x3 mask matrix - the sum of the multipliers must equal one, so each element equals 1/9. The conformal shape matrix is blurred element by element - the calculated values of each blurred element are influenced by its neighbours.

a) Matrix convolution:



b) Convolution calculation:

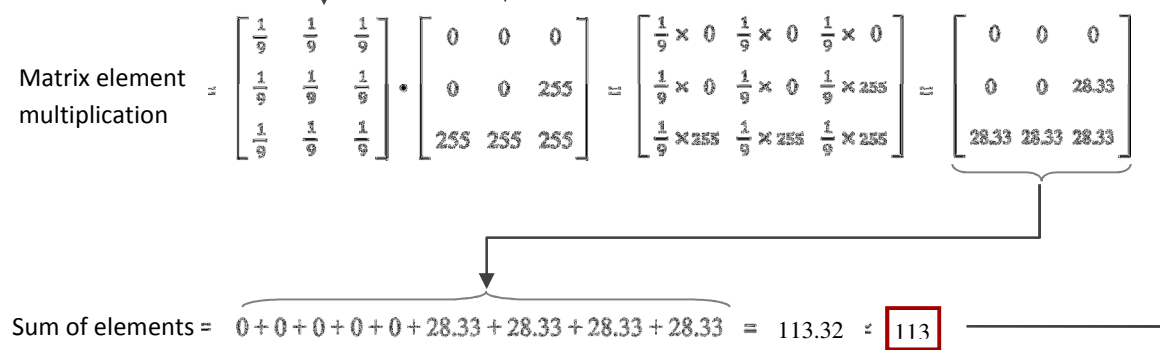


Figure 7-23: Convolution calculation for a single element of a conformal shape matrix

For any initial element (an example highlighted in red in Figure 7-23a), the mask matrix is multiplied with it and its neighbours (highlighted in blue). The matrix multiplication is a dot product where

each element of the mask matrix is multiplied by the corresponding conformal shape element (Figure 7-23b). The multiplied elements are then summed and the nearest integer of this value becomes the value of that initial element in the blurred conformal shape matrix. This is because the blurred conformal shape matrix is reconverted back into an 8-bit bitmap, that only accepts integers between 0 and 255.

The blurred conformal shape matrix possesses the required graded boundary, however the boundary of the conformal shape has expanded in the convolution process. This is shown in Figure 7-24. This would result in a trimmed structure that was slightly larger than the input geometry. To rectify this, the blurred conformal shape matrix is trimmed to the original conformal shape matrix (through the same method that trims structure to the conformal shape as described in Section 7.1.2).

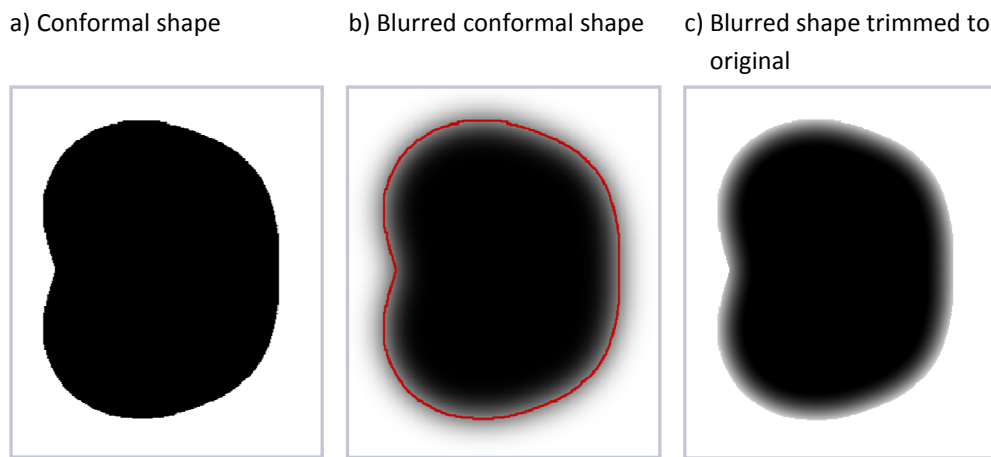


Figure 7-24: Trimming the blurred conformal shape matrix to the original conformal shape

In practice, the mask matrix is larger than the 3x3 matrix used in the example in Figure 7-23. increasing the size of the mask matrix increases the level of blurring. By default a 30x30 mask matrix is used to generate a wide enough greyscale band around the conformal shape, although this can be varied as required.

The functional grading process detailed in Section 7.2 uses this blurred conformal shape as an input to generate a structure composed of helical struts that transform into straight struts at the conformal shape boundary. This ensures that each boundary strut connects to the net skin. Figure 7-25 demonstrates this geometric grading. The blurring process has generated a graded boundary around the conformal shape - for the lightest value of grey, a straight strut is mapped to the internal structure slice; as the greys get darker, helical struts of increasing helical diameter are mapped to the internal structure slice. The majority of the conformal shape slice remains black and the actual strut type chosen for the structure is mapped to it.

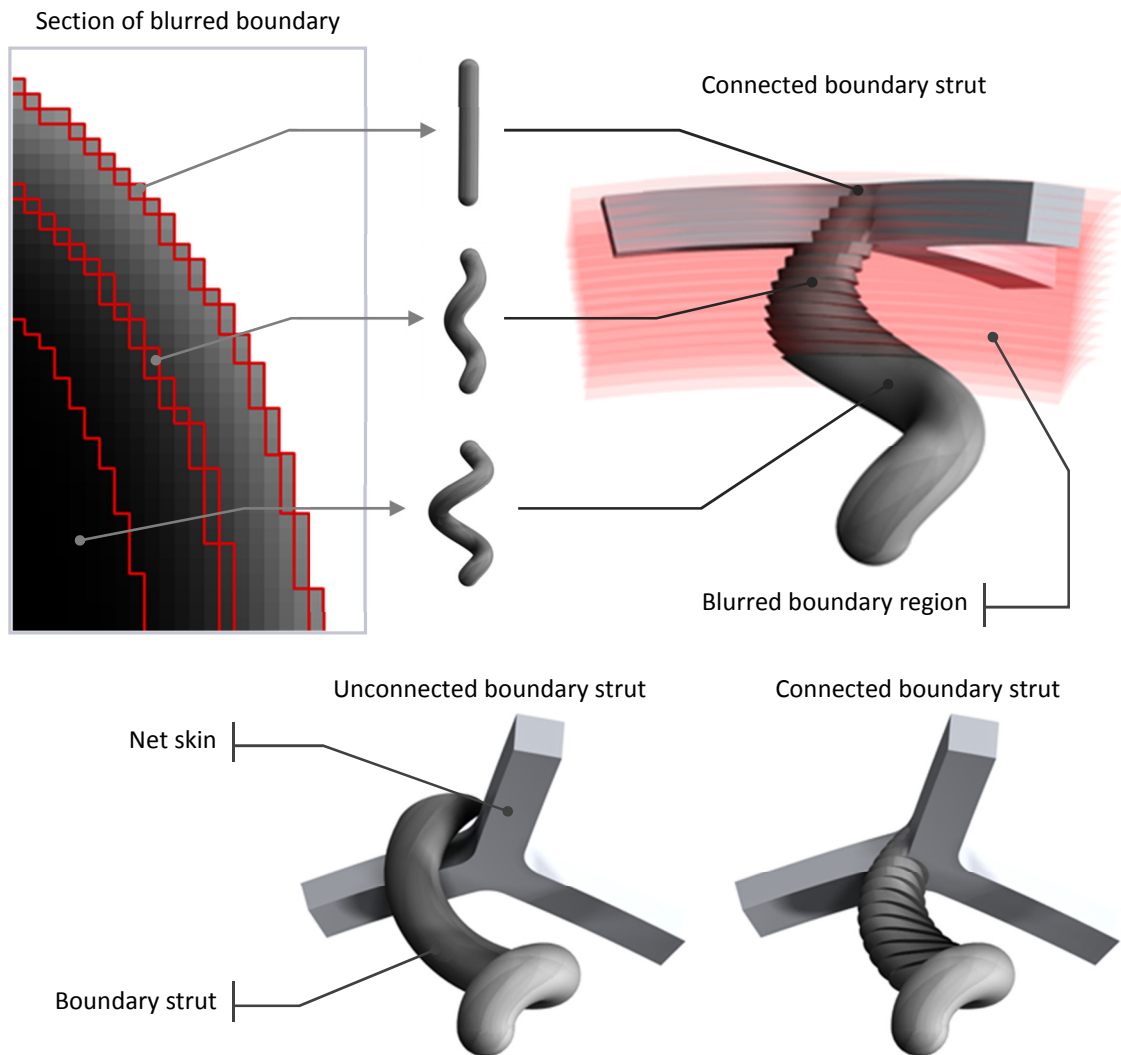


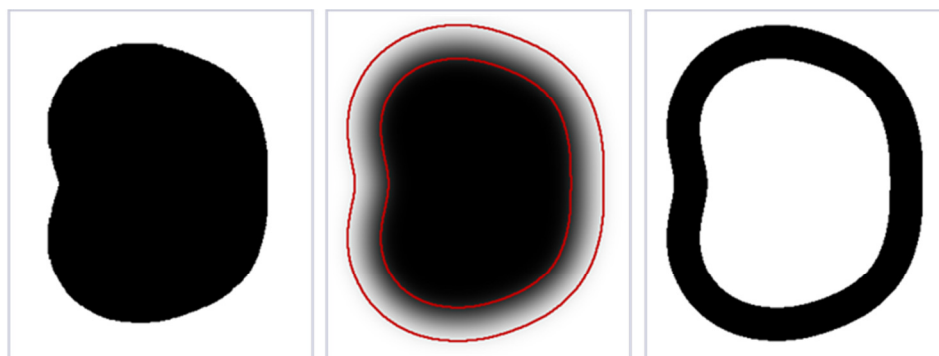
Figure 7-25: Re-aligning a complex strut with the net skin

#### 7.4 Constructing a Skin from a Conformal Shape

The conformal structure method requires two inputs to generate a skinned, trimmed structure, although some work has been done to reduce the inputs required to one. A method has been devised to automatically construct the skin slices from the conformal shape input. The advantage of this is that it reduces the work load on the user. The method uses 3D convolution (rather than the 2D convolution implemented in the previous section), applied to the entire conformal shape, rather than individual slices.

Each conformal shape slice is read and written as a page of a three dimensional matrix. In much the same manner in which a square mask is used to blur an image in 2D convolution, a cube mask blurs the conformal shape volume in 3D convolution. An individual slice depicts this blurring in Figure 7-26. Any element of the matrix that retains its original value or zero or one (i.e.: any element that

has not been blurred) is converted to one. Any element that has been blurred (i.e.: any element that has any value between zero and one) is converted to a zero, as also shown in Figure 7-26. This has the effect of generating a solid skin, with a thickness that is controlled by the strength of the convolution. The stronger the convolution (i.e. the larger the convolution mask), the thicker the generated skin. This can then be used as the input for the construction of the net skin.



*Figure 7-26: Generating a hollow skin from a conformal shape*

The advantage of this convolution method is that a hollowed conformal shape can be automatically generated as the input skin required by the conformal structure method. The user need only supply the one conformal shape for trimming a structure to and any type of skin required can be automatically produced within the method. The disadvantage with 3D convolution is that it is computationally expensive. The time taken to convolute a voxel model is generally higher than that of performing a 'hollow' operation in conventional CAD software. As conformal shape size increases, the time taken to generate a skin through convolution quickly becomes long enough to be considered unfeasible, although with further work a similar or more efficient means may be developed.

## 7.5 Structure Visualisation

When generating structure at the slice level it becomes difficult to visualise a design. This can be an important issue for a user that is unable to have complete confidence in a design that cannot be visually inspected. It also makes communication of a design to other stakeholders difficult. The structure can be observed layer by layer, but this is not a particularly clear visualisation technique as the design cannot be viewed as a whole.

To rectify this, a method has been developed to generate a 3D surface model from the raster slice data using isosurfaces. As discussed in Chapter 3, isosurfaces are a common means to convert a voxel model to a surface model. By generating a triangular polygon from groups of neighbouring voxels that comprise the boundary of the trimmed structure, a faceted boundary representation is

generated. An example of the method is shown in Figure 7-27, with a GUI that has been developed to allow the user to import saved data from a particular structure design that is automatically output from the conformal structure method.

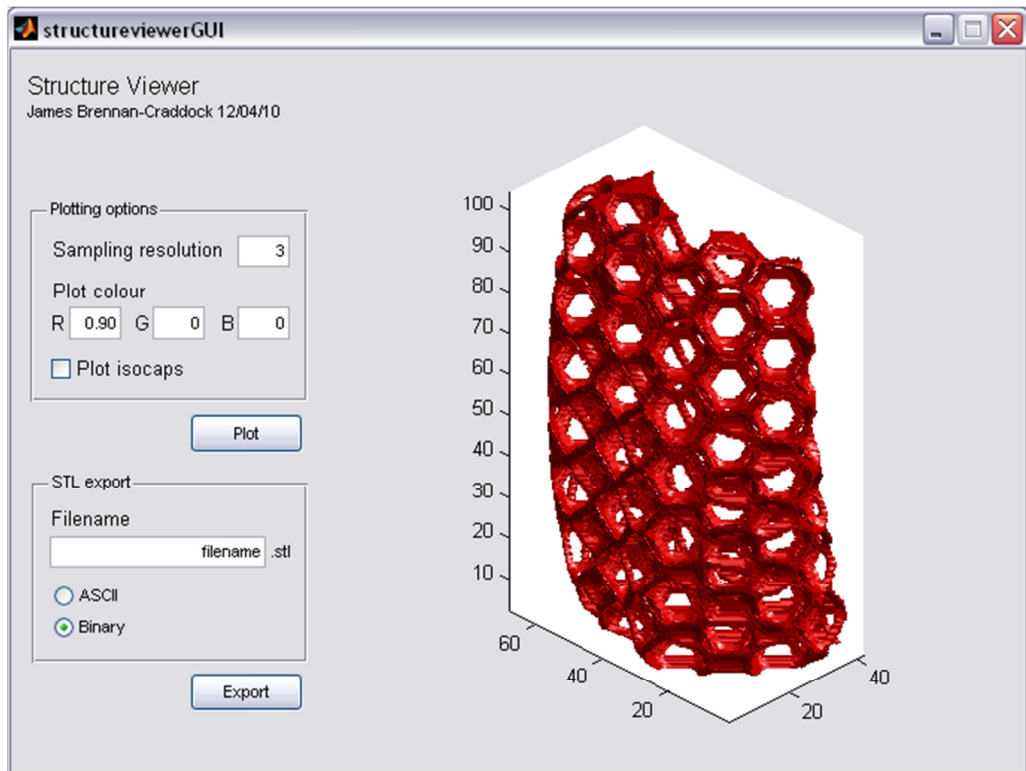


Figure 7-27: Structure visualisation with isosurfaces

Stepping between layers is observed with the isosurface, which, while aesthetically displeasing is a more accurate representation of an additively manufactured part. In that sense, this structure visualisation method can be considered more of a WYSIWYG ('what you see is what you get') display than the mathematically perfect curves and surfaces represented in a conventional CAD model.

As also discussed in Chapter 3, the issue with an isosurface generated from a voxel model is that each isosurface polygon is on the same scale as the voxels it is constructed from. This generates a surface that is composed of many small polygons and quickly becomes computationally expensive to display. The structure visualisation function utilises a simple method to alleviate this problem. When loading the trimmed structure data, the user also selects a sampling resolution value to construct the isosurface at. At a sampling resolution of 1, the exact isosurface is constructed. At a sampling resolution of 2, the method skips every other voxel in the model to construct a coarser isosurface. A sampling resolution of 3 utilises every third voxel and so on. Examples are shown in Figure 7-28.

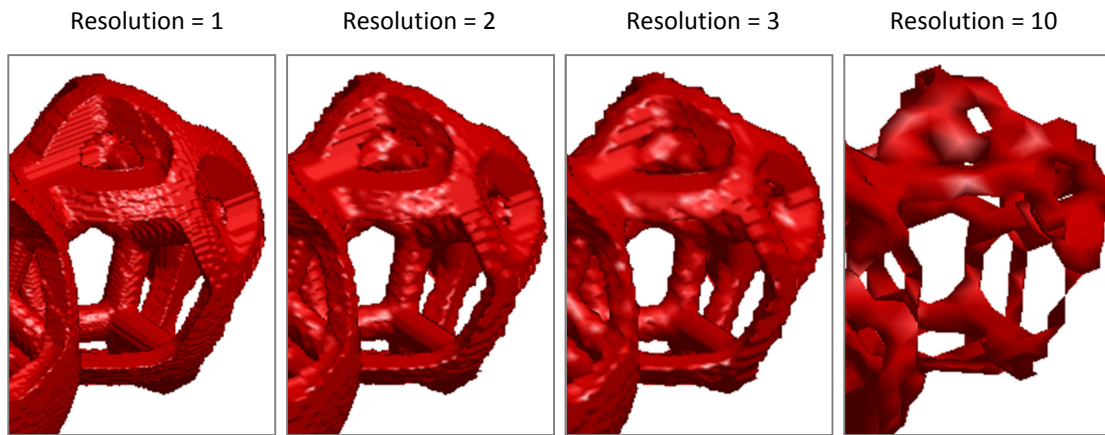


Figure 7-28: Effect of sampling resolution on isosurface quality

The voxel skipping occurs in all three axes and considerably increases isosurface construction speed and interaction rates with the model (smoothness of zoom and rotation). The obvious trade-off is that the displayed model is less accurate with increased sampling resolution. For pure visual communication of a design, empirical observations suggest that a sampling resolution of 3 allows a reasonable compromise between accuracy and rate of interactivity.

Also included in the visualisation function is a script to write the isosurface in an ASCII STL format if required. The standard process of writing a CLI slice file is still recommended (especially for larger trimmed structures), but the ability to write STL files gives further freedom to the user.

## 7.6 Strengths and Limitations of the Conformal Structure Method

### 7.6.1 Speed and Robustness

The speed of the conformal structure method is dependent on the size of the input slices. A large slice with no conformal shape geometry on it will take approximately the same time to process (for the majority of the steps in the method) as the same size slice full of geometry. The process that actually trims geometry considers one pixel at a time. As stated in Section 7.1.2, if a conformal shape pixel is black (solid geometry) then a corresponding structure pixel is copied over. If the conformal shape pixel is white (empty void) then the original void pixel is copied over. Either way, the same operation is carried out, just referencing pixels from different origins. The same is true when considering structure complexity.

Table 7-3 compares processing time for simple and complex structures trimmed to a particular conformal shape. The conformal shape is comprised of 250 x 460 pixel slices and totals 350 pixels (otherwise considered as 250 x 460 x 350 voxels). A simple, straight strut structure and a helical

strut structure were each trimmed to fit the conformal shape 25 times. The mean completion time and standard deviation are presented.



	 Straight strut		 Helical strut	
<b>Breakdown of method steps</b>	<b>Mean time (s)</b>	<b>SD</b>	<b>Mean time (s)</b>	<b>SD</b>
Initialising	0.17	0.03	0.18	0.01
Reading input slices	48.48	0.59	46.11	0.72
<b>Generating trimmed structure</b>	<b>1.77</b>	<b>0.06</b>	<b>1.77</b>	<b>0.06</b>
Generating net skin	1.63	0.19	1.60	0.22
Combing internal and net skin	0.32	0.01	0.32	0.01
Tracing boundaries and writing CLI file	0.04	0.00	0.04	0.00
<b>Total time</b>	<b>52.42</b>	<b>0.82</b>	<b>50.02</b>	<b>1.00</b>

Table 7-3: Breakdown of time spent by each stage of the conformal structure method for two strut types

As highlighted, the actual time spent trimming is the same for both structure types. The majority of the time is spent reading the input slices of the conformal shape and skin. The method has high repeatability, as shown by the low standard deviations across 25 iterations.

The method has no concept of how complex a structure is, because regardless of its geometric complexity, it is still represented by the same number of pixels as any other structure. Conversely, the method is less efficient at trimming structures to a conformal shape that do not fit efficiently in a cuboid envelope, as shown in Figure 7-29.

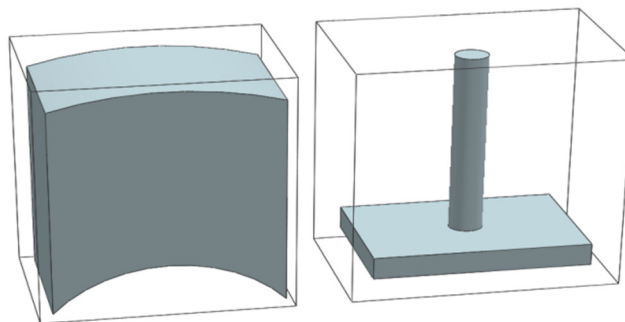


Figure 7-29: Shapes with different volumes in the same 3D envelope

The two shapes shown in Figure 7-29 will take approximately the same time to process through the method, despite having very different volumes, and thus different volume of structure required. This new method is highly robust in that regardless of input, a trimmed structure will be generated as long as the matrix is of the right format. There are no issues with surface validity that are readily



apparent in conventional B-rep CAD models. Additionally, the process is scaleable; resolution can be easily changed to generate large parts at a low resolution (for 'draft quality' structures or for coarser AM processes) or small parts at a high resolution.

### 7.6.2 Initial Overhead

An important issue of the method is the high level of initial overhead. Currently, every cell type that the method uses has been modelled in CAD and then converted to raster slices. The geometry in this form is not parametric - there is no elegant way to vary the dimensions of these voxelised cells. Cell diameter can be varied by resizing the slices, but this will also scale any other parameters, such as strut diameter accordingly. Currently, the cell library includes cells with 2mm and 2.5mm diameter struts. If a 2.2mm strut diameter cell is required, a new cell must be modelled and imported into the library. For the infinite combination of cell types and cell parameters available, adding new, specific cell designs to the library quickly becomes unfeasible. A potential fix for this is to implement structure construction within the method. The user selects a cell type, which is then automatically constructed and sliced. As they are required, cell types can be constructed, sliced and stored for future use.

### 7.6.3 X-Y Plane

A constraint of the process is that structure orientation must be set at the beginning of process. Unlike the conventional route of CAD-STL-slice file, where orientation and positioning of a model in a build can be determined at the STL manipulation stage, generating a trimmed structure with this methodology requires correct orientation at the beginning. Once the part has been sliced and cross-sections determined, the part is frozen at its current angle. The part may still be rotated around the z-axis (as shown in Figure 7-30) but the part cannot be easily rotated in other axes without significant deformation.

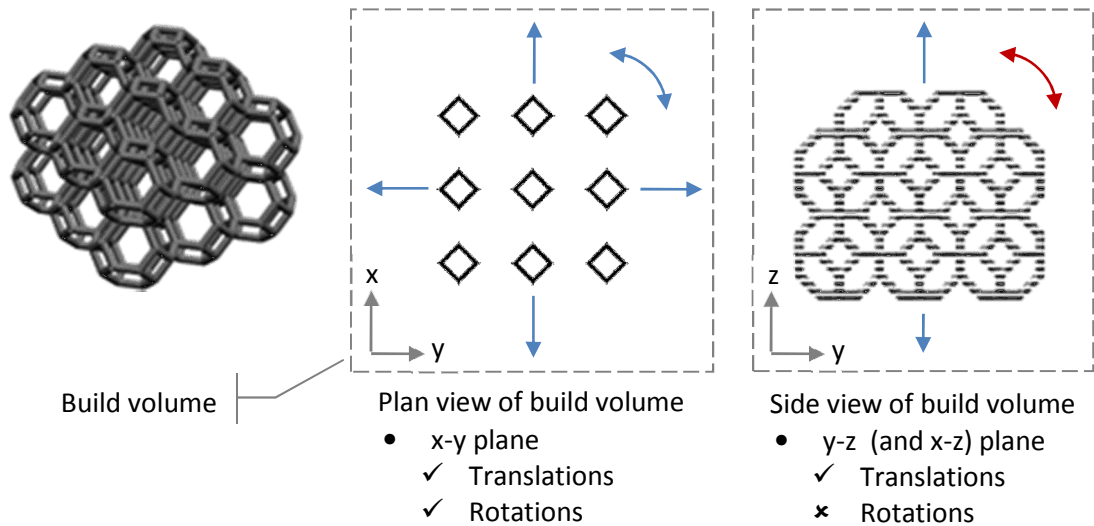


Figure 7-30: Orientation lock of sliced parts

The repercussion of this is that some planning is required at the early stages of trimmed structure generation to ensure that the output can fit into the laser sintering build volume. This becomes particularly important if the part is to share the build volume with other models.

#### 7.6.4 Pixel Stepping

Stepping in the z-axis (between layers) is a given for an AM process, but this method also generates stepping in the x and y-axes due to the pixel-based geometry manipulation that occurs.

The nature of bitmaps means a part's boundary is ultimately described as a stepped profile. Parts made by this process will potentially have this surface artefact that parts from other processes won't. The method of tracing the bitmap when converting to a vector-based slice file reduces the jaggedness of a part boundary as discussed in Section 7.1.5. Nevertheless, the pixel stepping will be a component of the surface of any part produced. Bitmap resolution can be increased so that this component will be insignificant compared to other roughness factors attributed to AM processes, however there is a trade-off with the speed at which the conformal structure method works.

## 7.7 Summary

This chapter has detailed the investigation of a novel method for fast and robust generation of conformal structures. The method is also capable of constructing a 'net skin', a new skin type that allows easy post-processing of parts while retaining the connectivity of the structure at the

trimmed boundary. An example part that has been generated with an internal structure of helical struts in excess of 100,000 in number is shown in Figure 7-31.



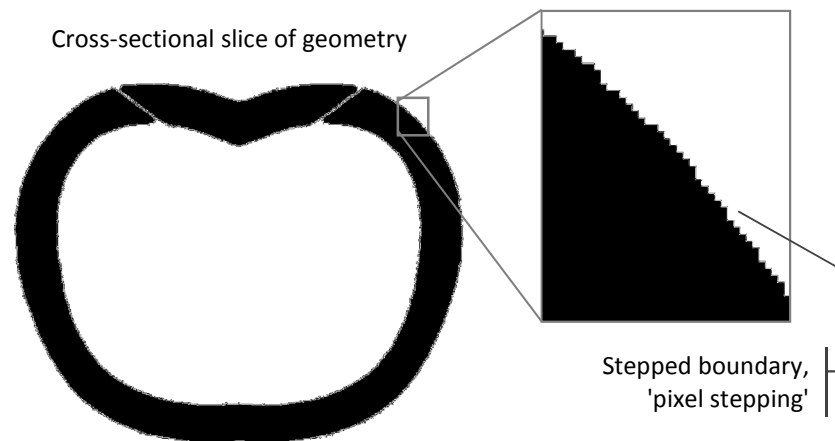
*Figure 7-31: A concept chest protector for taekwondo designed by the author on the Scuta project*

The conformal structure method also implements several complimentary features, such as the ability to visualise the trimmed structure as a 3D surface, the ability to export STL files as well as the default CLI slice files, and the means to automatically generate the input required for skin generation. This chapter has also demonstrated the method's ability to generate functionally graded structures, laying the groundwork for integration with optimisation techniques.

While the pixel-based process affords many advantages, the primary being that the method's speed is largely independent of structure complexity, it is the basis of the method's weaknesses. The most significant weakness is that any part constructed has a stepped surface, due to the discrete nature of the pixels that describe it. This 'pixel stepping' is an extra geometric artefact on parts produced by this method that other processes do not. For this reason, the effect of pixel stepping must be quantified, which is the aim of the next chapter.

## 8 | The Effect of Pixel Stepping on the Surface Roughness of Conformal Structure Method Produced Parts

The conformal structure method described in the previous chapter generates trimmed structures for additive manufacture in a rapid and robust manner. This is achieved through converting 3D geometry into a series of rasterised slices. Any geometry created through this methodology is ultimately described with a jagged boundary as a direct result of the discrete pixels that form it. Parts made by this methodology will have this 'pixel stepping' that parts generated conventionally would not, as shown in Figure 8-1.



*Figure 8-1: Pixel stepping at a boundary*

There are two primary disadvantages associated with surface roughness. Aesthetically, an extra source of roughness can detract from the look of the part. More importantly, the stepping attributed to the conformal structure method could potentially form the starting point for cracks to propagate from, ultimately weakening the structure.

Although the conformal structure method could generate parts for any of the additive manufacturing processes, the significance of any pixel stepping will be determined on polymer parts produced by laser sintering. As discussed in Chapter 1, laser sintering is the only AM process that can fully realise the geometrically complex structures that this method is capable of generating. It is intended that the resolution of the raster slices used in the method is high enough that this pixel stepping will be insignificant compared to the roughness attributed to the powder-based process. However, since the method produces parts with an extra source of roughness, the extent of this must be identified.

## 8.1 Experiment Aims

The primary aim of this experiment was to determine if the pixel-stepped profile of parts produced by the conformal structure method was manifest on manufactured samples. Because pixel stepping varies depending on the angle of the boundary, the experiment aimed to determine at what angles this is most significant.

Because laser sintering is a powder-based process, the surface roughness of parts produced is relatively high [167]. This experiment aimed to ascertain if and to what degree this existing source of surface roughness masked a pixel-stepped boundary. If pixel-stepping was obviously apparent in manufactured parts, the need to improve the output of the conformal structure method would be evident.

## 8.2 Experiment Method

### 8.2.1 Sample Design and Manufacture

Rather than attempting to measure pixel-stepping on a complex structure, a series of simple test samples were designed, as shown in Figure 8-2. The sample was a 25x25mm flat square of 4mm thickness; a set of which were oriented through a range of angles in the slice plane. The samples were manufactured on an EOS Formiga P100 polymer laser sintering machine.

Two sets of samples were generated - one set generated in CAD, one set generated through the conformal structure method. The CAD set were generated conventionally; modelled in CAD and converted to STL before conversion to a standard vector slice format for the P100 (the open source CLI format). The second set were also modelled in CAD, but then converted to a raster slice format. The raster slices were input into the conformal structure method (omitting the structure generation step) and the pixel-stepped boundaries of the rasterised shapes were traced and written to a vector slice file (CLI). This resulted in a set of samples with pixel-stepped boundaries through a range of angles. This ensured that the experiment objectives were met - comparing the conformal structure method samples with the conventional samples would determine if there was an effect on surface roughness, while comparing conformal structure method samples with each other would determine where any stepping artefacts were most significant.

As discussed in the previous chapter (in Section 7.3.6) the method of tracing employed in the raster to vector conversion slightly modifies the profile of a part. Rather than a part exhibiting the sharp stepping of a pixelated border, the tracing method essentially chamfers this 90° stepping to 45°

steps, as shown in the inset of Figure 8-2. Although this reduces the jagged effect of pixel-stepping somewhat, the stepping artefact is still present. For this reason, the stepping is still referred to as 'pixel stepping' as pixellation is the cause.

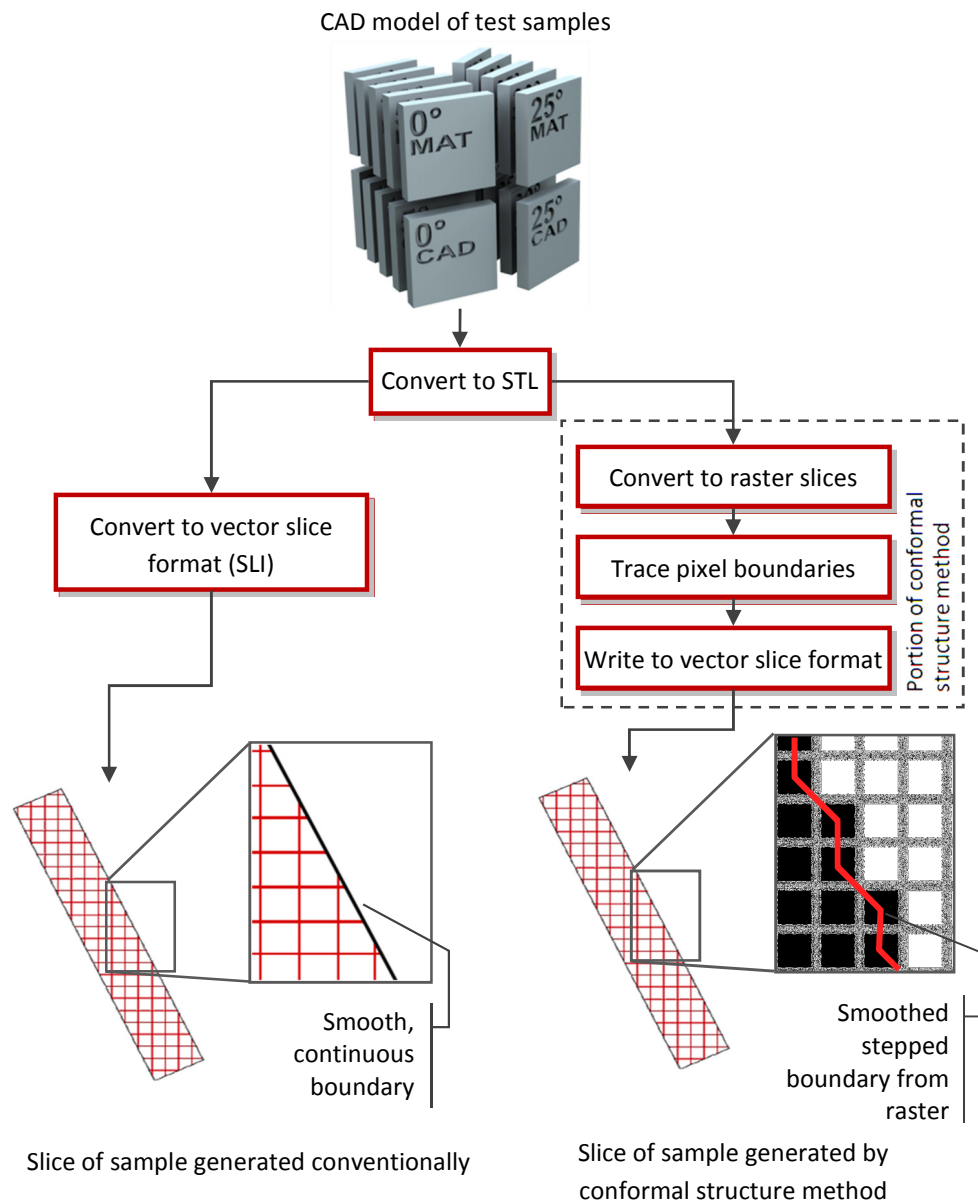
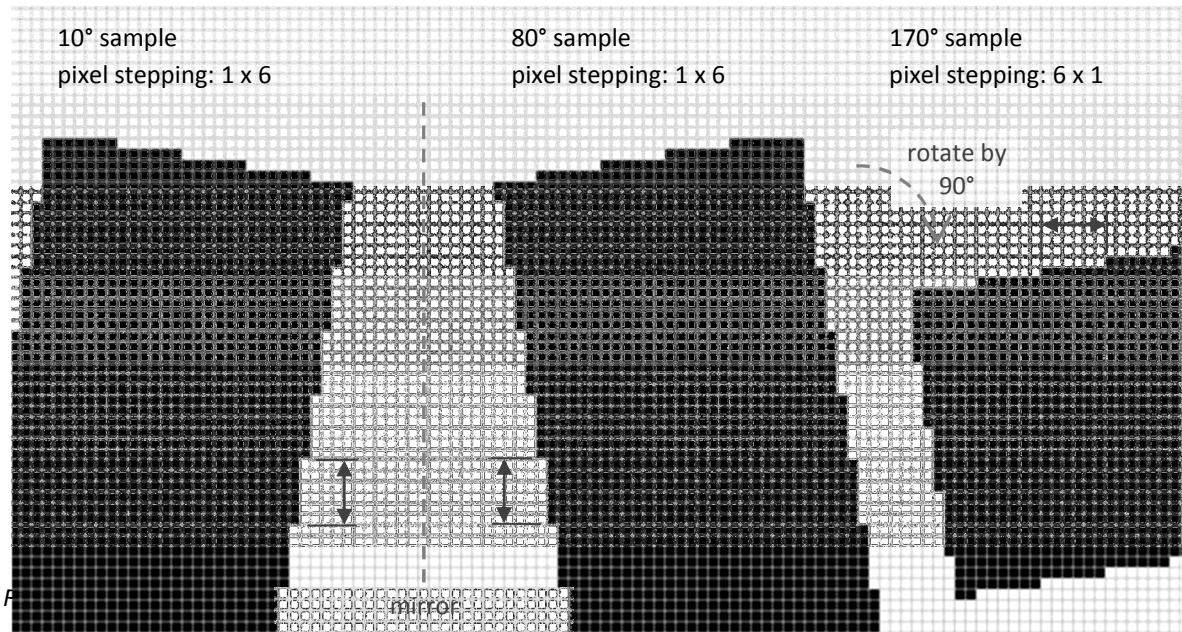


Figure 8-2: Data flow for conventionally generated and method generated samples

The range of rotation through the sample set was 0° to 45°, at 5° increments - this covers the full 360° range of pixel stepping 'patterns' that could be generated while maintaining a reasonable number of samples to analyse. For example: the pixel stepping patterns that form from 0° to 45° are reflected at 45° to 90°, meaning that the 45° to 90° do not need to be tested. Similarly, 90° to 180° is a reflection of 0° to 90°, and 180° to 360° is a reflection of 0° to 180°. This is shown by transforming a sample in Figure 8-3; by reflecting and rotating the 10° sample, an 80° and 170° sample can be generated that have the same stepping pattern.



The conformal structure method can generate parts at any resolution, which affects the scale of pixel stepping. For the purpose of this experiment, the resolution of parts produced by the conformal structure method was set at 0.14mm/pixel - the default resolution of the slicer used to generate the input bitmaps.

On the underside of each sample the name, build angle and whether it was generated in 'CAD' or by the conformal structure method ('MAT' - short for Matlab) were written to allow for easy identification. The sample set was duplicated to allow comparison between examples of the same angle; for each angle, two samples were manufactured. This would also go some way to mitigate the potential of external factors affecting a particular sample and skewing results.

The samples were built in the centre of the build volume of the P100, where the process scans at its most accurate [168]. The samples were built on the machine's 'speed' setting - a default scan setting available in the control software that optimises build parameters for build time. Virgin PA2200 - EOS' standard nylon material - was used to manufacture the samples.

## 8.2.2 Surface Roughness Measurement

A Taylor-Hobson Talysurf CLI 2000 system was used to measure the surface profile of each sample. A 2D profile was measured across the width of each sample in the direction that stepping could occur on the conformal structure method-generated samples.

The Talysurf system has the capability to measure both with contact and non-contact methods. The contact method drags a diamond-tipped stylus across a surface to accurately measure a profile. The

non-contact method uses optical means to measure surface roughness and has the advantage that the risk of scratching samples is eliminated. The method is called scanning white light interferometry (SWLI) [169]: A light source is reflected off the measured surface and compared with a reference. From this the position of the surface can be determined [169]. On this apparatus the contact method has a higher resolution; the stylus has a finer point than the laser spot ( $0.2\mu\text{m}$  versus  $0.8\mu\text{m}$ ) and the spacing between measurements is twice that of the optical alternative ( $0.5\mu\text{m}$  versus  $1.0\mu\text{m}$ ).

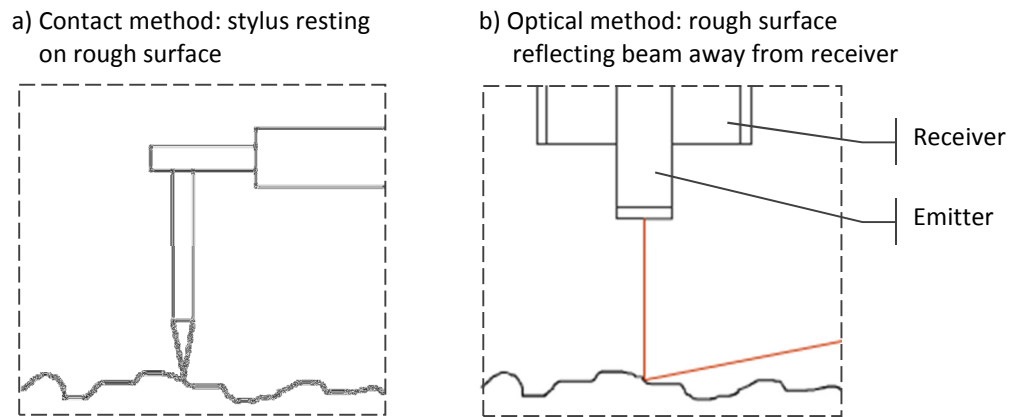


Figure 8-4: Excessive reflection off a rough surface with optical measurement

The test setup is shown in Figure 8-5. The sample was placed onto the platform with the potential pixel-stepping direction parallel to the direction of platform movement. The sample was lined up with an arrangement of locator pins and bar and fixed into place with a small amount of putty. The identifying 'CAD' or 'MAT' markings on each sample were used to ensure a standard orientation between them.

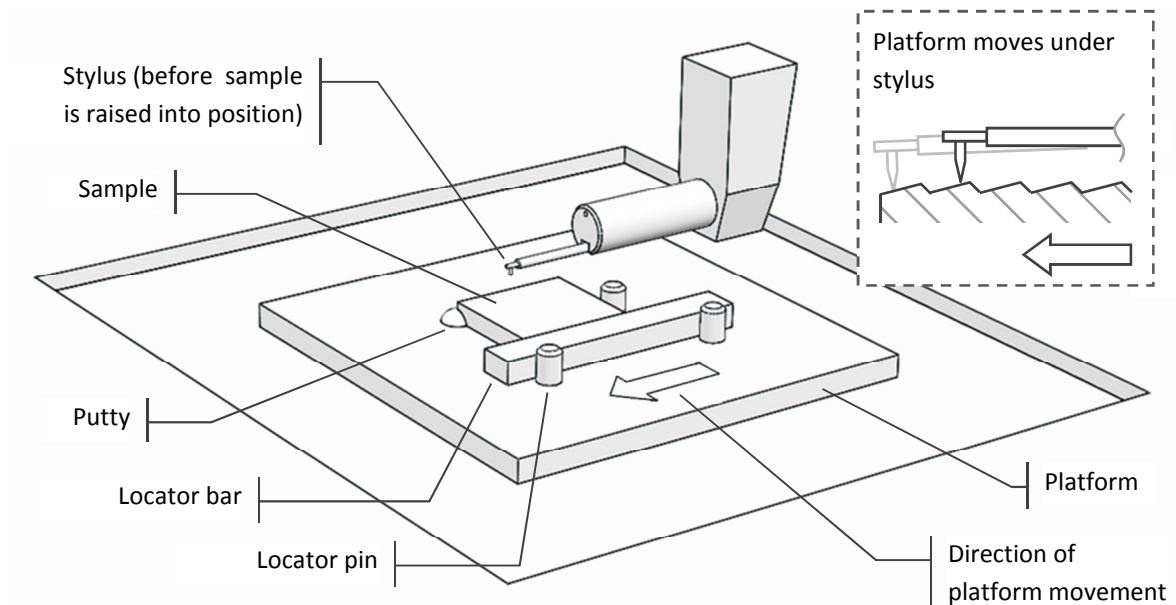


Figure 8-5: Talysurf CLI 2000 test bed



To perform the profile measurement, the platform first raised the sample into position. The platform itself moves while the stylus mounting remains stationary, although the stylus is free to move vertically to trace over the surface features of the sample. Each sample was 25mm in length, but the scan length for each measurement was set to 23mm. This was to prevent the stylus from 'falling off' the edge of the sample which could potentially damage it. At a 500 $\mu\text{m/s}$  scanning speed, there was a 0.5 $\mu\text{m}$  spacing between measurements. This measurement resolution exceeds both the minimum feature size the laser sintering process is capable of (0.1mm [170]) and that that the conformal structure method was set to (1 pixel = 0.14mm). This ensured adequate sampling for the analysis discussed in the next section.

### 8.3 Fourier Analysis

For any physical, fluctuating wave - such as a sound wave or a radio transmission - Fourier theory can be employed to decompose it into the range of sine waves that compose it [171,172]. Fourier transforms convert the wave into the 'frequency domain' by plotting it as its constituent sinusoids of different frequencies and amplitudes [173]. The magnitude of a sinusoid's amplitude is a measure of how significant a component it is of the wave. Fourier analysis has a broad range of uses, and was utilised in this testing to analyse surface roughness.

By converting a surface profile to the frequency domain it was possible to view the constituent spatial frequencies (or 'wavenumbers') of the surface roughness. Figure 8-6 illustrates this for one of the samples generated through the methodology (a sample rotated by 10°).

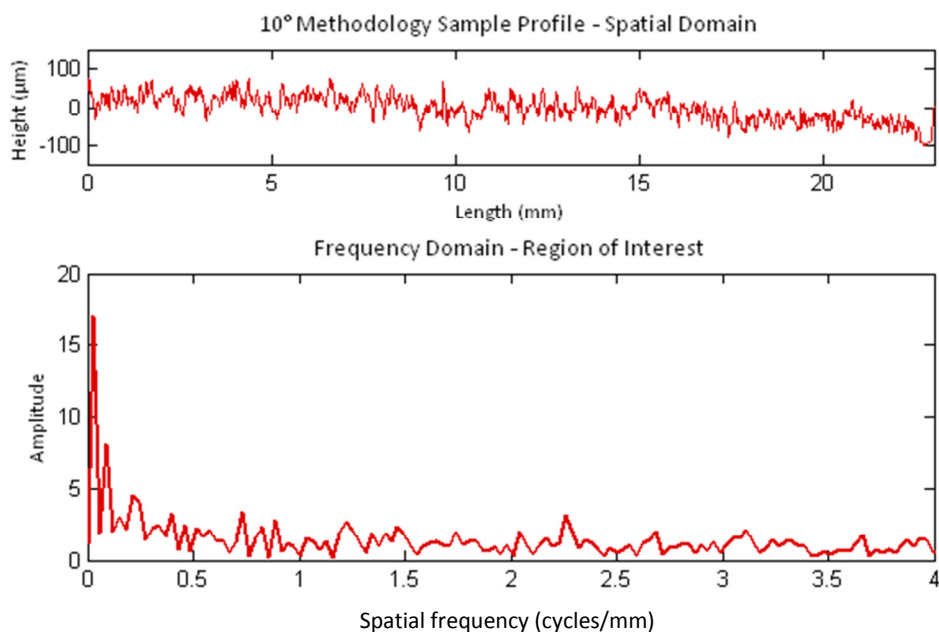


Figure 8-6: Profile and Fourier transform of 10° sample

The form of the profile in the frequency domain is termed the 'Fourier spectrum'. Figure 8-6 shows the original profile in its entirety, and a cropped view of the spectrum which highlights the region of interest. This region is of interest because it is the area of the Fourier spectrum with highest amplitudes. Thus in this region are the spatial frequency components that are of greatest significance to the surface profile.

For computational analysis, any real-world profile must be sampled; a continuous analogue wave must be converted into a digital sequence of values. The Nyquist sampling theorem governs proper sampling in Fourier analysis, which states that sampling rate must be at least twice that of the highest frequency component in the waveform being sampled [174]. The average grain size of PA2200 powder diameter ( $56\mu\text{m}$  [175]) was considered the highest possible frequency component of surface roughness for this testing. The  $0.5\mu\text{m}$  measurement spacing detailed in the previous section is used as the sampling rate, which at ten times smaller than average grain size satisfies the Nyquist sampling theorem requirement.

The purpose of the Nyquist sampling theorem is to ensure that the phenomenon of aliasing does not occur during analysis [174]. An insufficient sampling rate can obscure the true nature of a waveform, as shown in Figure 8-7. In this example, the true, high frequency signal has been insufficiently sampled, so when reconstructed digitally prior to analysis, aliasing makes the signal incorrectly appear as a lower frequency.

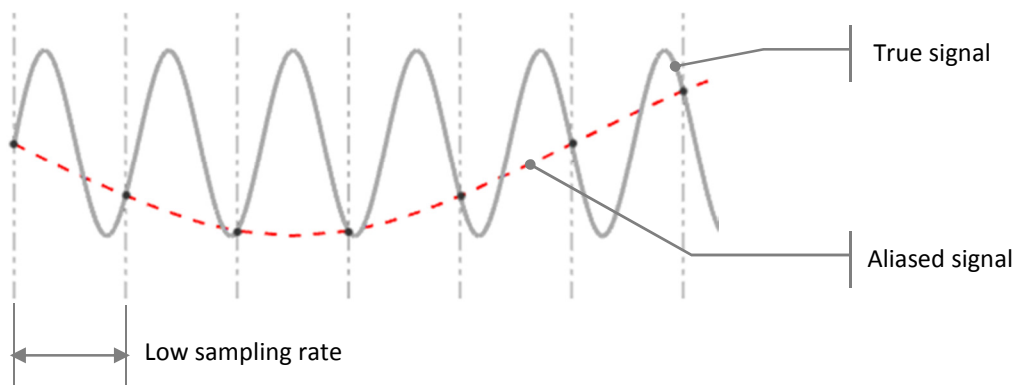


Figure 8-7: Low frequency aliasing

For any sample generated by the conformal structure method, the expected periodicity of the pixel stepping can be calculated. The pixel-stepped profiles resemble what in signal processing is called a sawtooth wave, a standard waveform that periodically ramps upwards then sharply drops. However, there are some notable differences between a sawtooth wave and the stepped profiles, illustrated in Figure 8-8. The first profile shows the sawtooth wave used for comparison with one of the pixel-stepped profiles, and has been scaled appropriately. The first major difference is that the

stepped profiles do not necessarily possess the perfect periodicity that a sawtooth waveform has. This is shown in the second profile in Figure 8-8.

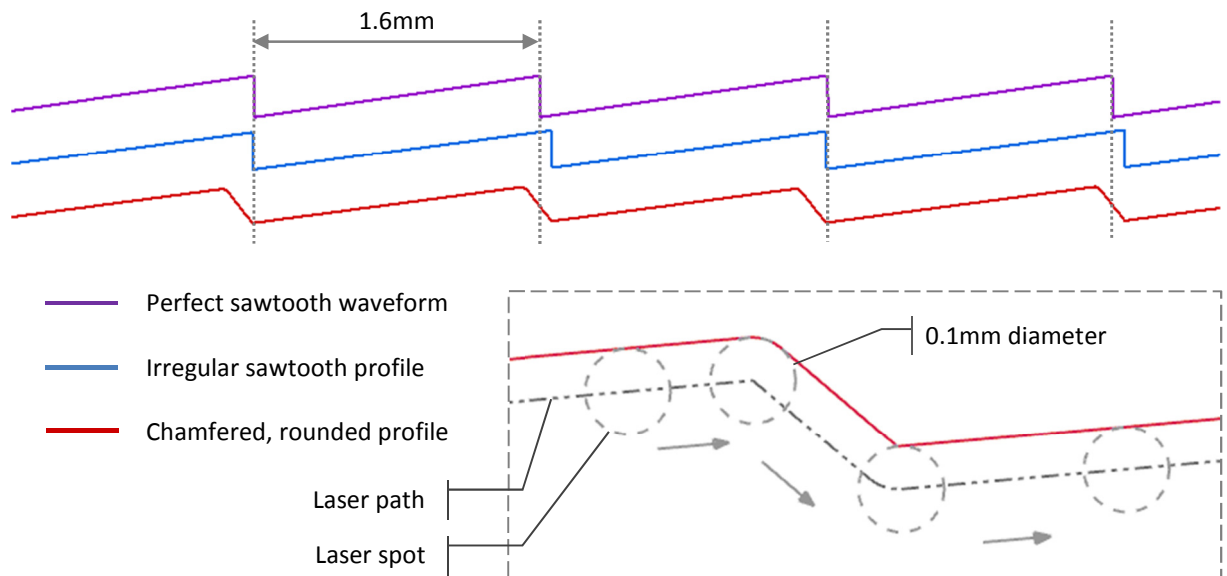


Figure 8-8: Transformation from a sawtooth profile to closer representation of actual pixel-stepped profile

The third profile possesses the aforementioned chamfering of the stepped profiles that occurs because of the tracing method employed in the generation of a slice file. With this modification, the profile resembles the actual slice file used in manufacture. During manufacture, the laser used to sinter the part has a diameter of 0.1mm. Taking this into account, the corners of any sintered profile will have rounded corners at the diameter of the laser. These rounded corners have also been applied to the third profile.

Figure 8-9 shows the sawtooth waveform in the frequency domain, a series of discrete peaks reducing in amplitude as frequency increases. The other two profiles are also shown. Just by adding some irregularity, the distinctiveness of the sawtooth profile is reduced. Most of the peaks are reduced in amplitude, with the exception of the lowest frequency, highest amplitude peak at 0.6 cycles/mm. This corresponds to the 'wavelength' of 1.6mm shown in Figure 8-8.

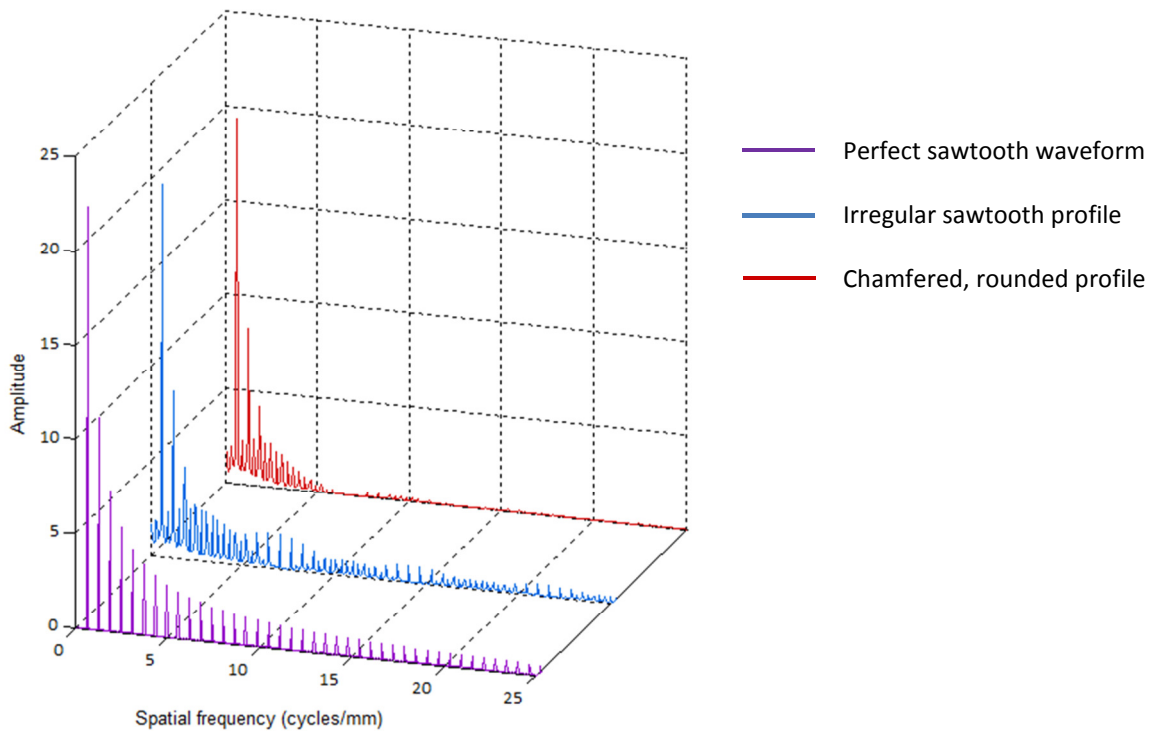
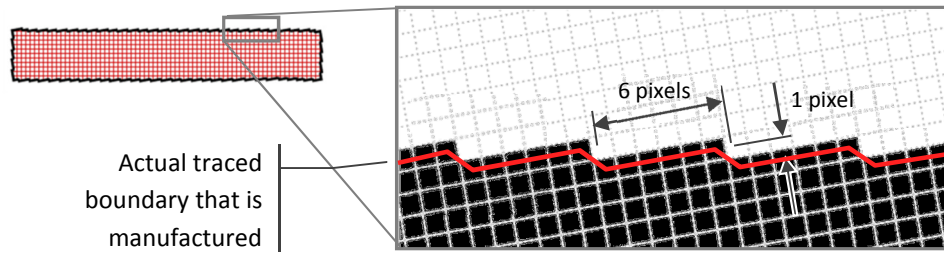


Figure 8-9: Transformation from a sawtooth profile to a closer representation of an actual pixel-stepped profile

The effects of chamfering the waveform add to the decimation of the distinctive sawtooth frequency domain shape. Again, with the exception of the lowest frequency peaks, each peak has dramatically reduced in amplitude to the point where all frequencies above 6 cycles/mm are reduced to almost nothing. This is to be expected as it is the high frequency components that give a sawtooth profile its sharp corners. What this essentially shows is that although a sawtooth waveform has apparent visual similarities to the pixel-stepped profiles in the spatial domain, in the frequency domain there is not much to be gained in comparing them. However, the lowest frequency component remains relatively unchanged between the profiles, so becomes a reliable signal in and of itself to check whether pixel stepping is a significant component of a particular profile. This lowest frequency corresponds to the wavenumber of the overall stepping; the fundamental frequency of the profile's repetition.

For each of the samples tested, the expected fundamental wavenumber of pixel stepping can be calculated. The empirically derived 'scaled wavenumber',  $k^*$  is used throughout this chapter as it matches calculated wavenumbers with actual results.

Pixel stepping of a 10° rotated sample:

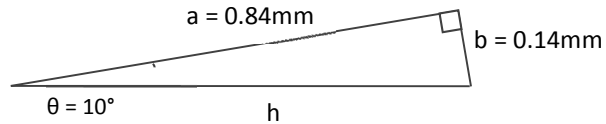


Wavelength of the pixel stepping:

$$1 \text{ pixel} = 0.14\text{mm}$$

$$a = 6 \times 0.14 = 0.84\text{mm}$$

$$b = 1 \times 0.14 = 0.14\text{mm}$$



Using Pythagoras' Theorem:

$$h^2 = a^2 + b^2$$

$$h^2 = 0.84^2 + 0.14^2 = 0.73$$

$$h = \mathbf{0.85\text{mm}} = \lambda$$

Wavenumber of the pixel stepping:

$$\text{Wavenumber: } k = \frac{2\pi}{\lambda}$$

$$\text{Scaled wavenumber: } k^* = \frac{k}{2\pi} = \frac{1}{\lambda} = \frac{1}{0.85} = \mathbf{1.18 \text{ cycles/mm}}$$

Figure 8-10: Calculating expected periodicity of the pixel stepping on a 10° sample

As previously stated, pixel-stepping profiles for the other samples are not necessarily a simple pattern repetition. A straight line can only be approximated by a pixelated representation. For any particular angle, the position of the pixels used to represent it may not follow a simple recurring pattern. The pattern will be periodic, but as a group of different length steps rather than a consistent single step: a 'non-uniform stepping', as shown in Figure 8-11.

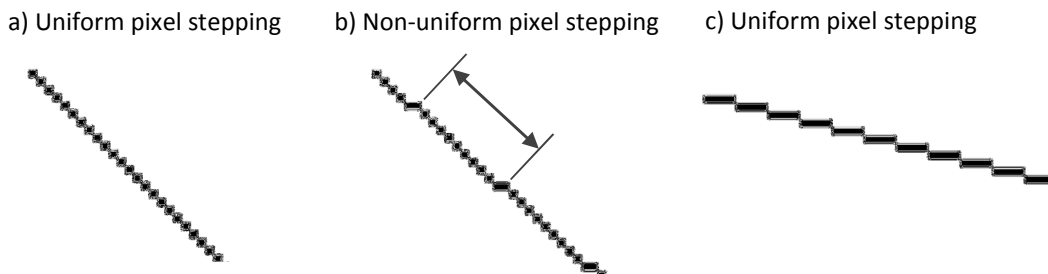


Figure 8-11: Uniform and non-uniform pixel stepping when approximating a straight, angled line

As such, there are multiple wavenumbers to identify that can be attributed to pixel stepping on several of the samples. Figure 8-12 illustrates the periodic stepping patterns apparent on each of the samples and the calculated wavenumbers as detailed in Figure 8-10.

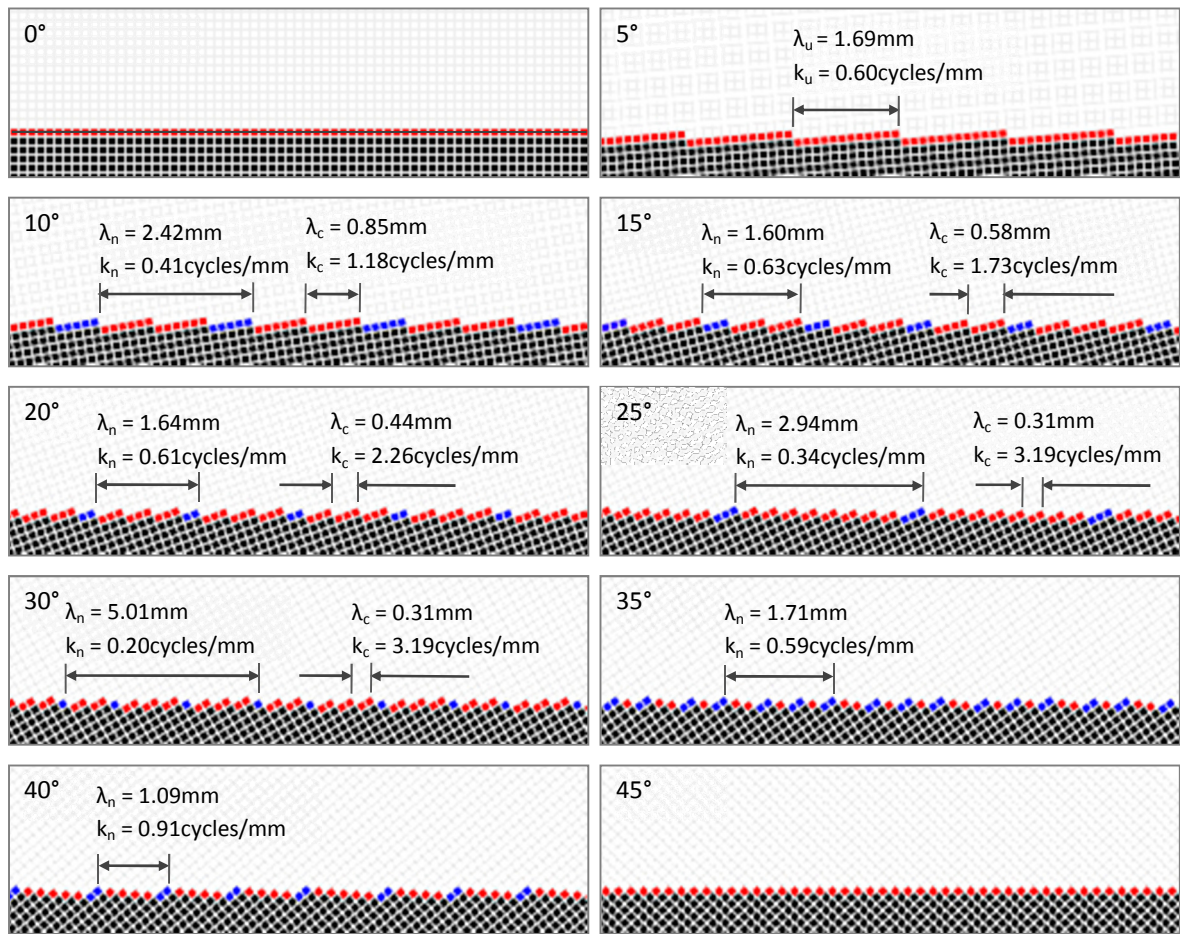


Figure 8-12: Identified pixel stepping patterns

For the sake of clear quantification, this figure shows actual pixel patterns rather than the slightly different traced boundaries. In Figure 8-12 and any following figures where wavelength or wavenumber are presented, the subscript 'u' refers to a uniform stepping pattern, 'n' refers to a non-uniform stepping and 'c' the major repeating component of that particular pattern.

For the 5° sample, only a simple uniform stepping pattern is observed. For the 10° to 40° samples, a non-uniform stepping is shown as a pattern of different length red and blue steps. Most of the non-uniform stepping patterns include a significant repeating component themselves that may be identified by the Fourier analysis, the wavenumber of which is also calculated. The 0° sample has no pixel stepping and while the 45° sample has a simple pixel stepping, when the profile is traced the result is a smooth line. As such, there are no wavenumbers to search for in the Fourier analysis of the 45° samples.

In fact, any profile that has elements of the same pattern observed in the 45° sample (i.e. a step after every pixel) will also have segments of a completely smooth line at those locations. This is shown in Figure 8-13: the 40° sample has segments of the 45° pattern and as such has a straight line component. This results in a traced boundary profile similar to that of the 5° sample, despite the very different pixel profiles.

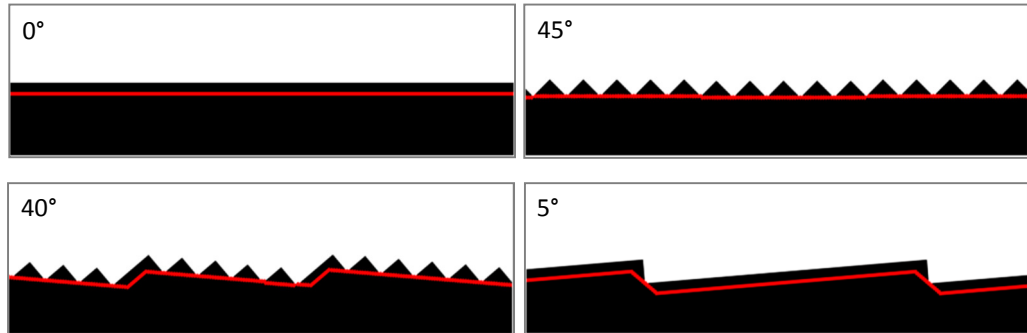
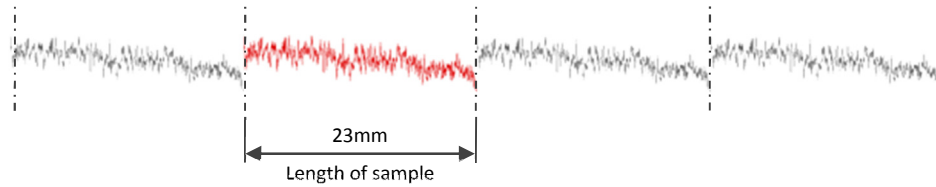


Figure 8-13: Differences between pixel profiles and traced profiles

When the sample profiles are transformed into the frequency domain (as shown previously for the 10° sample in Figure 8-6) a low frequency, high amplitude peak is observed that does not correspond to those expected from Figure 8-12 if stepping is significant. This unexpected frequency is actually a consequence of the finite length of the sample. Fourier analysis implicitly assumes that the input profile is periodic: that it is a section of an infinitely repeating signal [176-178]. Figure 8-14 shows the 23mm length, 10° sample repeated as Fourier analysis assumes it to be. As the figure demonstrates, there is an obvious discrepancy between the end of one profile and the start of the next.

Fourier analysis assumes input signal repeats into infinity:



Scaled wavenumber of sample repetition:

$$k^* = \frac{1}{\lambda} = \frac{1}{23} = 0.04 \text{ cycles/mm}$$

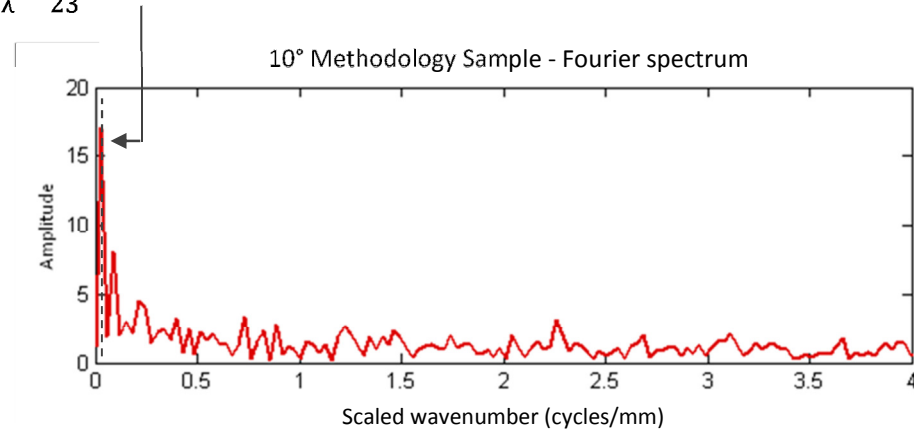


Figure 8-14: Edge effects - the source of low frequency peaks

This periodic discrepancy emerges in the frequency domain as a low frequency, high magnitude peak and is termed an 'edge effect' [178,179]. It is this edge effect that is the cause of the identified peak. Fourier theory states that viewing a profile in either the spatial or frequency domain are just two different ways of viewing the same data [173]. By applying a windowing function to the original profile, the edge effects apparent in the frequency domain can be mitigated. A windowing function will also reduce the effects of spectral leakage also caused by the discrepancy between the start and end of the profiles [180].

A window is a weighting function that promotes certain features of a profile - examples of windowing functions and their uses are shown in Table 8-1. [173,181]. There are many more types of windowing functions designed for specific signal processing applications which are not likely to be appropriate for this study.



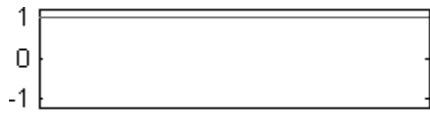
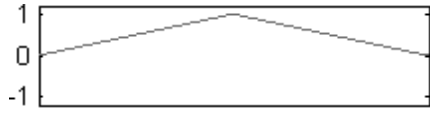
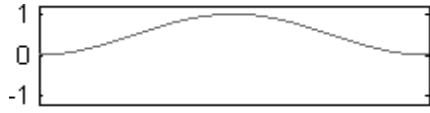
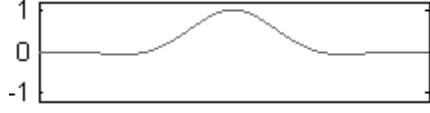
Windowing Function		Uses
Rectangular (no window)		Synchronous sampling, where start and end of profile meet seamlessly.
Bartlett		Random profiles
Hann		Random profiles
Flat Top		Sinusoids

Table 8-1: Selection of windowing functions (modified from [177,182])

Both the Bartlett and Hann windows are suitable for supposedly random profiles [177,182]. The samples tested in this study can be considered as having irregular surface profiles due to the manufacturing process. After some experimentation with applying the different windows to the data, the Hann function was considered to yield the clearest results. Figure 8-15 shows how the Hann window modifies the original 10° sample profile and compares the windowed and non-windowed samples in the frequency domain.

The window function, when applied to the original profile, artificially reduces the profile to zero at its start and end. This essentially formats the profile for the Fourier analysis: the majority of the profile is left intact by the window, while the smoothing of the ends allows the profile to seamlessly repeat into infinity.

As shown in Figure 8-15, the high amplitude peak attributed to edge effects has been substantially diminished in the Hann-windowed profile. The sharp disparity between repeated profiles is removed by the windowing function. However, there are still no particularly significant frequencies appearing above the general noise. There are no significant high amplitude peaks to match the expected wavenumber of either the non-uniform stepping pattern (0.41 cycles/mm) or the significant repeating component of that pattern (1.18 cycles/mm).

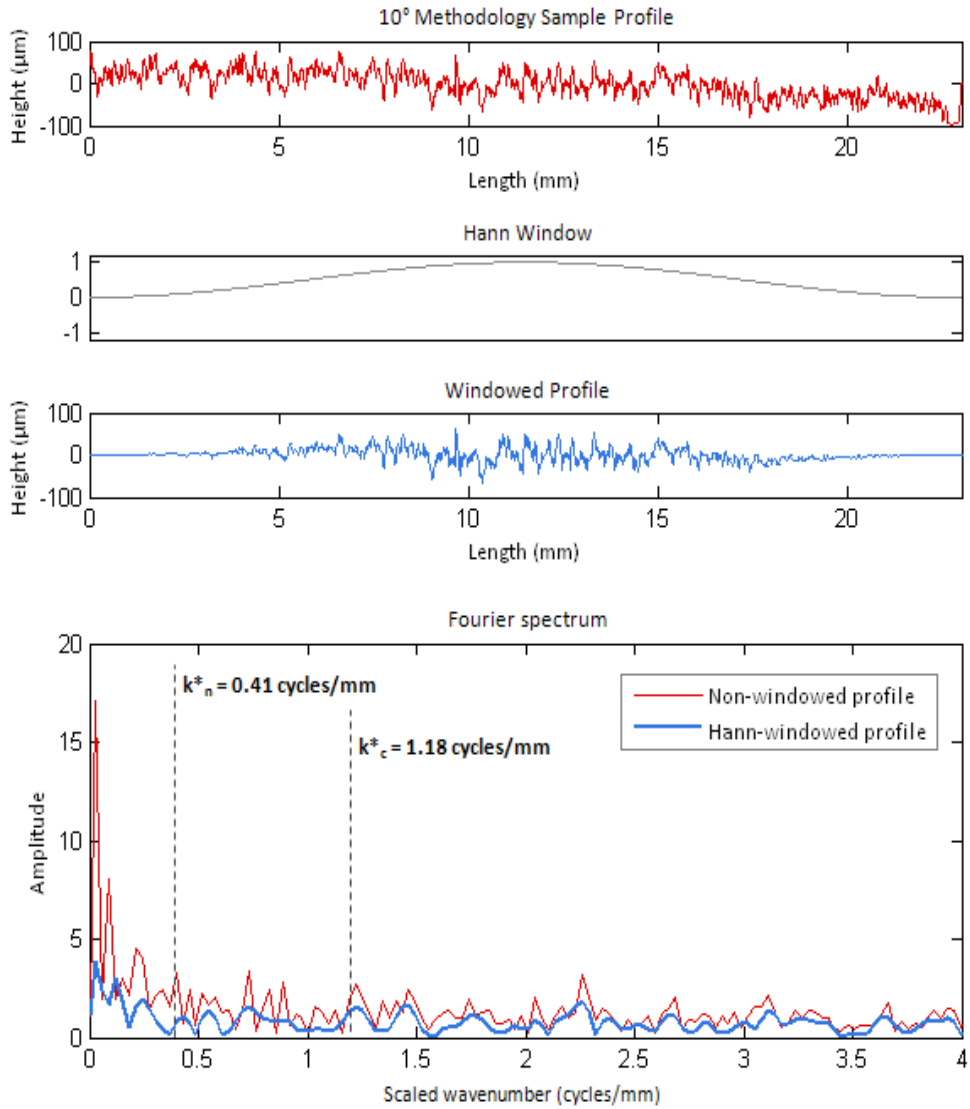


Figure 8-15: Comparing original signal with Hann windowed signal for 10° method sample

From this it can be determined that - for a 10° angle - the pixel stepping surface artefacts produced by the conformal structure method are not a significant contribution to a manufactured part's surface roughness. This is supported by Figure 8-16, which compares Hann-windowed frequency domains of both 10° conformal structure method samples and both 10° conventionally generated samples. Neither of the conformal structure method samples exhibit significant peaks at any frequency, including the expected 1.18 cycles/mm; the Fourier spectra consist almost entirely of low-amplitude noise like their conventionally generated counterparts.

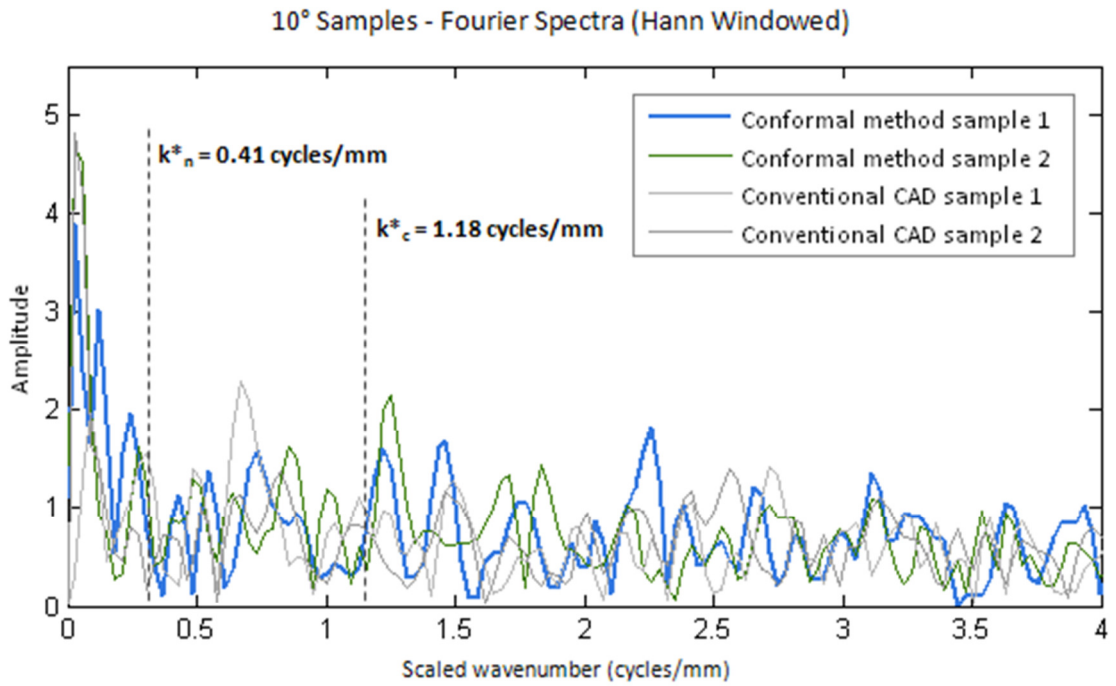


Figure 8-16: Comparing windowed samples of all 10° samples - conventionally and method generated

The only significant instance where stepping was noticeable was on the 5° samples. Less significant instances are discussed subsequently. The 5° angle was the most acute of the angles tested and exhibited a stepping pattern with the longest periodicity. The frequency spectra of the two 5° samples are shown in Figure 8-17 and clearly show this periodicity as a significant component of the manufactured samples' surface roughness, with a high amplitude peak at the expected wavenumber.

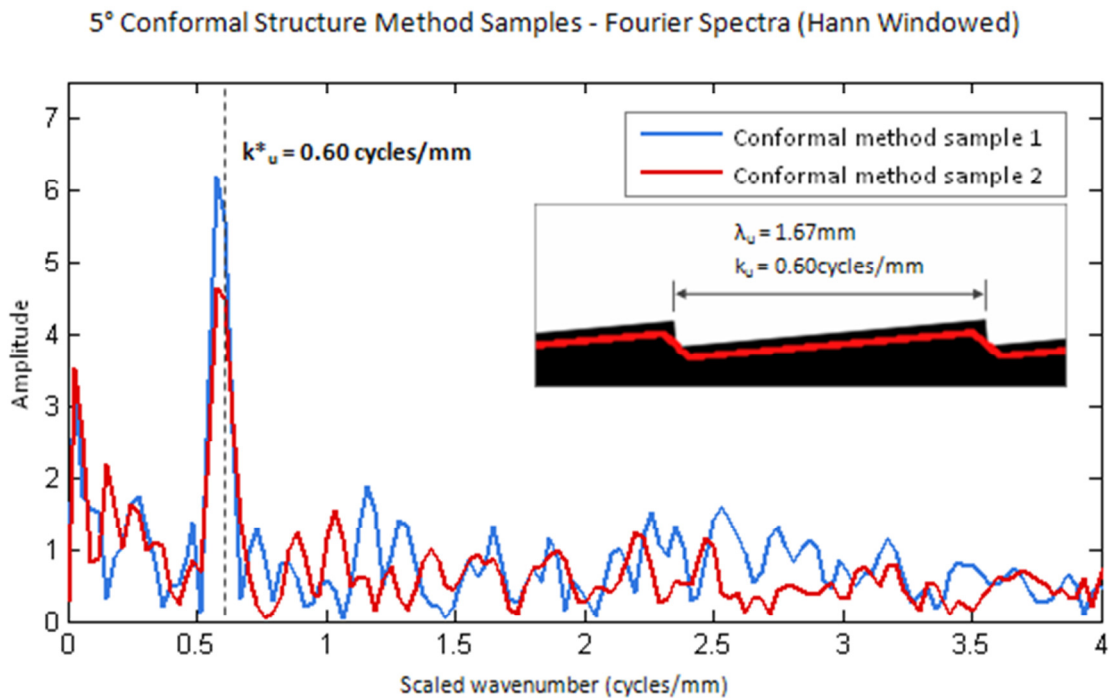


Figure 8-17: Hann windowed signal for 5° method samples

This same wavenumber is observed on both samples, which corresponds to the expected wavenumber calculated from the original stepped profile. This wavenumber does not appear on the conventionally generated 5° samples, as shown in Figure 8-18, which proves that it can only be attributed to pixel stepping.

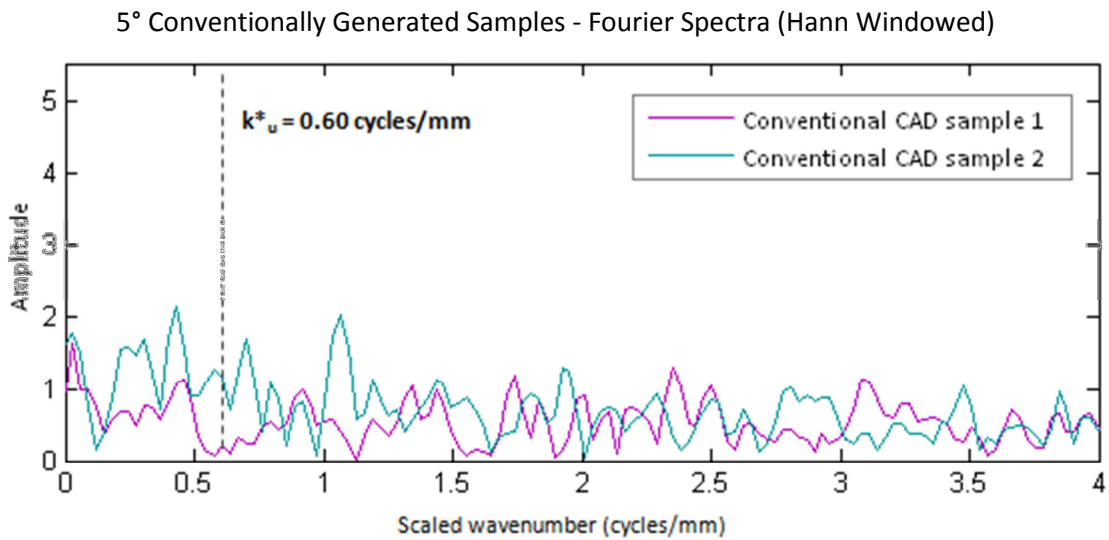


Figure 8-18: Hann windowed signal for 5° conventional samples

Figure 8-19 shows the Fourier spectra for each of the samples generated through the conformal method. For the benefit of clarity, the x-axis omits the spectra from 0 - 0.1 cycles/mm (i.e. the peaks attributed to edge effects). The expected wavenumbers (i.e. those corresponding to the pixel-stepping wavelengths as shown in Figure 8-12) are shown for each sample. As discussed, it is clear that the 5° sample has the only significant pixel stepping-produced peaks.

Two wavenumbers are highlighted with dotted lines on the 25° and 40° samples. While the amplitude of these peaks are not much higher than the general noise, the wavenumbers do correspond to the expected non-uniform stepping patterns apparent on the pixel profiles. These samples as shown in more detail in Figure 8-20 highlight these potential peaks.

There is a correlation between the expected wavenumbers and peaks in both sets of samples. For both the 25° and 40° sets each of the samples peak at the expected wavenumbers, whereas other relatively high peaks in one sample do not agree with those of the other. As previously discussed, the actual traced boundary of the 40° sample is similar to that of the 5° sample despite very different pixel stepping patterns. It is this pattern's distinct similarity that is the reason the Fourier analysis has picked up on it. Like the 5° samples, the long uniform stepping is noticeable above the general roughness of the laser sintered parts.

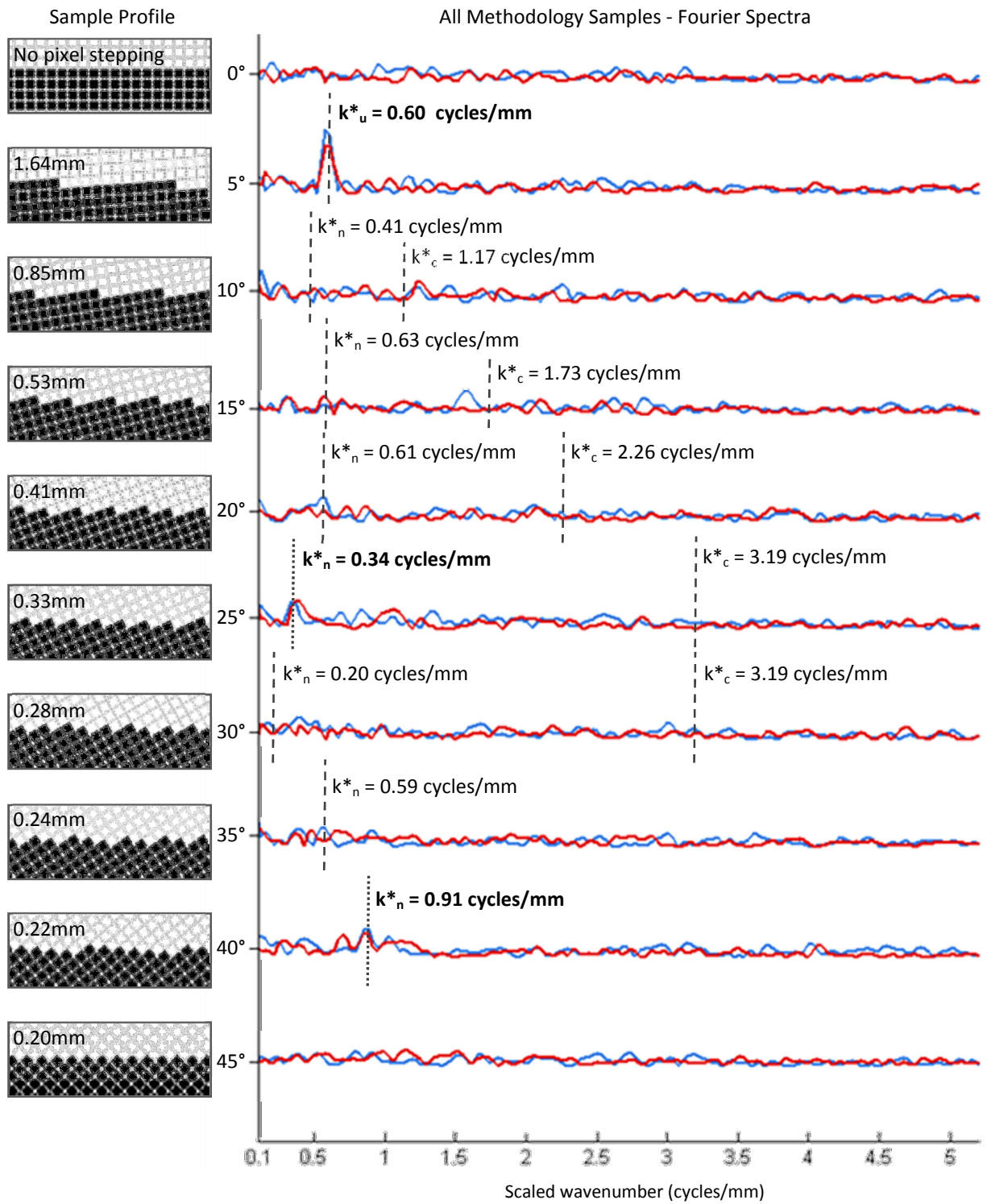


Figure 8-19: Fourier spectra for all samples generated through conformal method

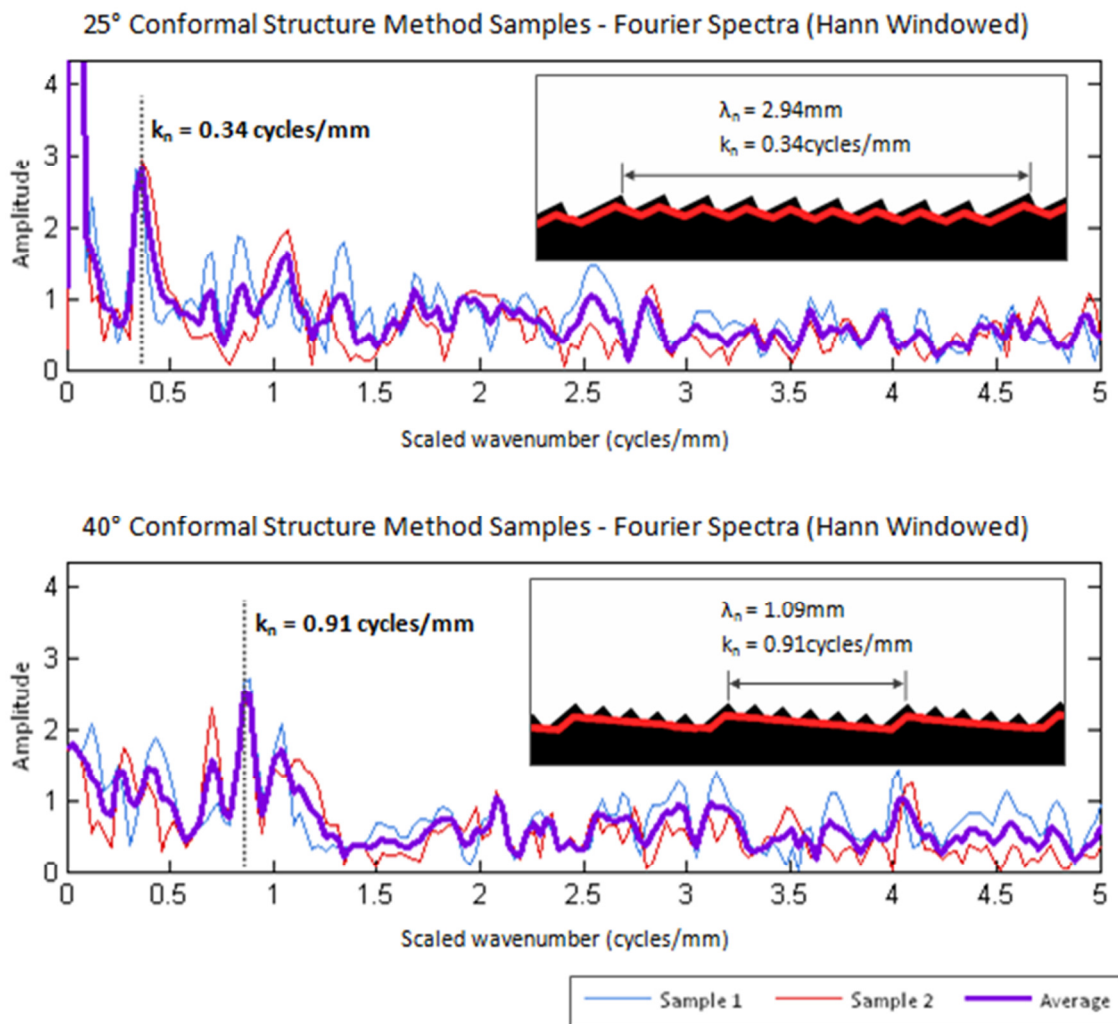


Figure 8-20: Hann windowed signal for 25° and 40° conventional samples

However, the amplitude of these particular wavenumbers (while significant over the general noise) are not nearly as significant as that found on the 5° samples discussed previously. The non-uniform stepping patterns expected on the other samples (10°, 15°, 20°, 30° and 35°) are not observed on the Fourier spectra at all. This makes the 5° sample a good measure of the effects of pixel stepping on any particular AM process at any particular method resolution. A 5° sample could be fabricated as a worst-case benchmark test piece. A 5° sample could also be used as an indicator of the effects of any methods taken to reduce pixel stepping.

#### 8.4 Methods to Reduce Pixel Stepping

To reduce the level of pixel stepping, one of two routes can be considered. By increasing the resolution of the conformal structure method, the amplitude of the stepping would be reduced. Because the conformal structure method already has the ability to vary its output resolution, a

control is already in place to reduce the effects of pixel stepping to an appropriate level. However with this approach, a compromise will always exist with the method's speed.

Another route to reduce pixel stepping would be to apply a smoothing function to the stepped profiles before parts are built, such as point averaging routines or mathematically fitting curves to the shapes. This is considered further work and out of the scope of this research. Even if pixel stepping is completely removed from the generated profiles, parts produced by additive manufacturing still exhibit stepping between layers which limits the level of overall surface smoothness that is achievable.

## 8.5 Conclusions

An experiment was carried out to determine how the pixel-stepping of parts produced by the conformal structure method affected the surface roughness of manufactured parts. Only the 5° rotated samples - with the longest uniform pixel-stepping periodicity - showed to have a significant effect on surface roughness at the resolution tested. This result could potentially be used as a means of calibrating the output of the structure trimming method to particular build parameters of AM processes. Knowing that a surface at a 5° angle in the X-Y plane will exhibit the most significant pixel stepping, test samples could be made at a range of resolutions through the structure trimming method.

Although pixel stepping is significant on the 5° sample in this experiment, it is important to note that the flat samples tested are a worst case scenario. The periodic stepping is only noticeable because it is the only feature of the surface. In practice, when the conformal structure method is generating complex, freeform lattice structures, the surface angle in any one region will rarely be a sustained 5°. Regardless, the testing showed that at the current conformal structure method's resolution

(1 pixel = 0.14mm) surface artefacts attributed to pixel stepping are apparent. To rectify this either increasing the resolution of the method or investigating smoothing algorithms has been suggested, the latter of which is considered further work.

## 9 | Conclusions and Further Work

### 9.1 Achievement of Research Aims

The work presented in this thesis was guided by two broad research aims:

1. To investigate a method to retain structural connectivity at the boundary of a trimmed structure
2. To develop a conformal structure method that:
  - a) Utilises the trimming method of conformal structure generation
  - b) Implements Research Aim 1
  - c) Develops the voxel method to fully exploit the geometric freedom of AM
  - d) Efficiently integrates into the conventional route of design to additive manufacture

Research Aim 1 was addressed with the development of a 'net skin', a method to re-connect cut struts at the boundary of the trimmed structure. Both in the B-rep concept and final voxel-based method, the process retains information regarding the connectivity of the structure. This ensures the correct construction of the net skin, always forming new struts between struts of the same cell. In the voxel-based conformal structure method, the net skin is generated by subtracting a 'hole cell' (that is aligned with the structure) from a solid skin. Although the net skin is not perfectly formed by the conformal structure method, this is considered a fair trade-off against speed.

Research Aim 2 was achieved with an advanced conformal structure method that implemented a voxel-based method to trim a selected structure type to a conformal shape. It generates a slice file to integrate into the conventional route; implements a net skin to strengthen the structure boundary and provides the foundation for functional grading of structures. Additional features to improve useability were developed, such as the ability to visualise parts and convert to the STL format if required.

### 9.2 Key Conclusions

- The voxel-based method was investigated as the basis of a method to generate conformal lattice structures. It is much more efficient than conventional CAD software at generating



conformal lattice structures and the speed at which it works is largely independent of geometric complexity.

- The novel 'net skin' method developed in this thesis has been shown to maintain connectivity at the boundaries of trimmed lattice structures, providing a major benefit over other methods of skinning. The net skin facilitates easier post-processing of conformal lattice structures.
- Although geometric complexity has minimal effect on the speed of the conformal structure method, the efficiency in which the conformal shape fits within a cuboid volume has a significant effect. This means that care must be taken to make the most efficient use of space.
- Voxel models are inherently stepped, although the coarseness of this can be controlled by changing the resolution of the voxel model. However, this has an impact on speed of the conformal structure method. Testing has shown that the default resolution set for the process (once converted to a slice file) is sufficiently fine to minimise noticeable stepping on laser sintered parts.
- The layer by layer stepping inherent in AM processes will always be a component of overall surface roughness regardless of the resolution of the method. The set resolution is adequate for the particular LS machine, although this would need to be reconsidered as the accuracy of the LS process improves in the future and indeed between different machine models and materials used.

### 9.3 Recommendations for Further Work

A grant from the East Midlands Development Agency Transport iNet has been awarded to develop the work in this thesis towards a standalone piece of software. To that end, there is still significant progress to be made to move from a Matlab-based concept to product that can be commercialised. The most substantial step would be to establish independence from Matlab by translating or re-writing the work into an actual programming language like C [183].

There are a number of improvements that must be made to the method as a whole first, to include steps of the method that are currently fulfilled by external processes. One such external step at the moment is slicing. Developing a slicing algorithm within the process enables it to be tailored to the specific needs of the method. Currently, the slicing algorithm exports a series of bitmaps, which the conformal structure method must convert to matrices for the trimming operation. An internal slicer could export a 3D matrix (the voxel model) of the sliced model directly.

Similarly, conventional CAD software is currently used to design both the conformal shape and every cell type included in the process. It is advantageous to permit the design of the conformal shape through conventional CAD which excels at representing models with 'shape complexity'. However the existing method of designing structures can be improved. Currently a separate CAD model must be sliced for each iteration of a particular cell type. For a Kelvin cell with a 15mm cell diameter, a CAD model must be constructed and sliced with a 1mm strut diameter, 2mm strut diameter, 1.1 mm etc. This is because it is difficult to modify voxel models, as there is no parametric information to label voxels with particular topological detail. The generation of a multitude of voxel cells is necessary, but it need not be a process that includes the user. The conformal structure method could include a 'cell constructor' function that would automatically construct and slice a cell given a set of dimensions. An example is shown in Figure 9-1.

The example gives the user several controls in the design of a particular cell type (based on the Kelvin cell); different cell types could be introduced as cell constructor functions are written. Cell diameter, strut diameter and helical diameter of the strut are entered. The cell constructor actually makes use of boundary representation, which - unlike voxels - can be parameterised to allow the same model to be resized as required. The result would then be sliced with the method's internal slicer to produce the voxel model (as a 3D matrix). This takes advantage of both modelling methods. B-rep is used to parametrically model a single cell, while voxels are used to efficiently represent the entire structure.

Data flow through the method would change to a situation where the required cell would be modelled and sliced at the start of the process. In contrast to the current situation which requires an ever-expanding library of every cell type and its iterations stored as voxel models.

As a method that constructs geometry at the slice level, there is currently no sufficiently developed means to visualise the generated conformal structures. For a user to have real confidence in the output of the method an interactive 3D visualisation is important. Within Matlab, isosurfaces were used to construct 3D models of the output as previously discussed, however the time taken to plot these lessened its usefulness. This is largely due to the method being run in Matlab, so migration to an actual programming language like C++ should go some way to rectifying this.

Another issue with generating geometry at the slice level, the output is specific to a particular process. The method currently only outputs slice files in the CLI format, which – while open source – is not a *de facto* standard that all AM machines are obligated to conform to. Although laser sintering was deemed the most capable AM process for fabricating lattice structures, other processes can fabricate certain structure types.

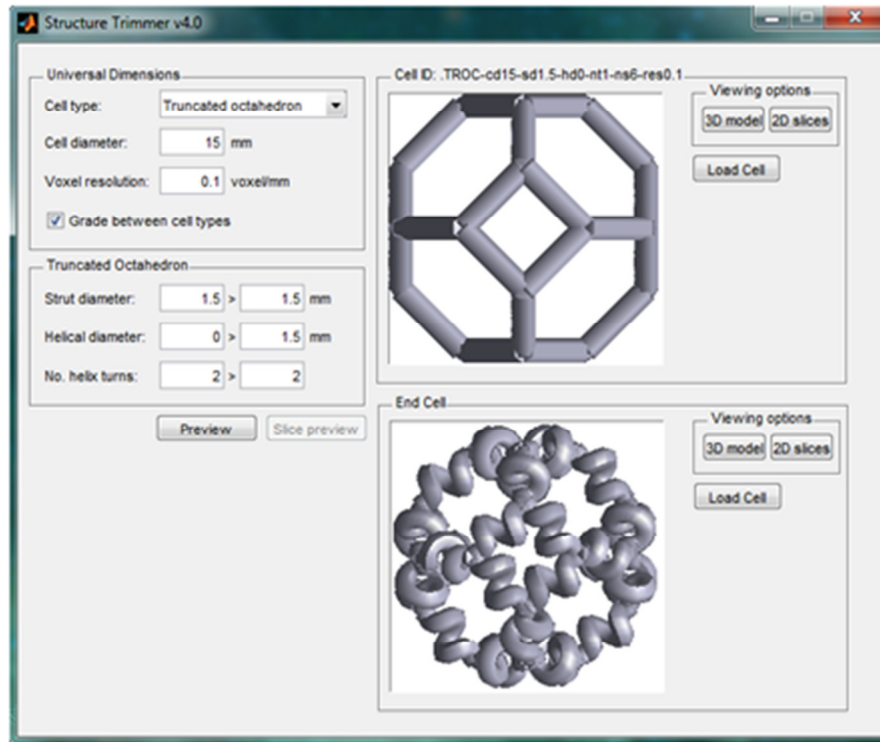


Figure 9-1: A B-rep cell constructor where cell type, cell diameter, strut diameter and helical diameter can be adjusted

Some structure geometries are self-supporting and can be built by AM processes that it would otherwise be unwise to do so. One such process is selective laser melting (SLM) from MCP – a process similar to laser sintering, but specifically for metals. The ability to construct some types of lattice structures in metal would be a useful addition to the method, so the necessity arises to obtain and integrate a file type compatible with SLM (such as the F&S format). Another potentially interesting AM process to integrate would be the jetting technologies from Objet; combining the functionally graded structures the method is capable of with the multiple material capabilities of jetting.

The capability to functionally grade between different strut types presented in this work is just a foundation to build from. A simple linear grading of structure across the length of a part has visual impact, but it is not ever likely to be useful in a real product. For a structure to be graded meaningfully, a whole new design philosophy must be developed. A structure must be graded according to regional loading and stresses. Integration with the finite element method could be one route to investigate. Stress variation according to FEA could direct functional grading. Another potential route would be the integration of the method with topological optimisation techniques. Given loading conditions to withstand, topology optimisation seeks to find the optimal location of material. Topology optimisation essentially works with voxels - material is moved around by changing the density of each voxel. These voxel densities could be linked to a specific cell type

within the conformal structure method as a means to guide functional grading. A simple example is shown in Figure 9-2.

A particular density as determined by a topology optimisation result would need to match up with a particular cell geometry. Before any method of functional grading is progressed, the actual mechanical properties of each structure must be characterised. Considering the level of control over structure geometry that is possible, as well as differences between AM processes and materials available, this is not a trivial task. But for a specific setup, it would be possible if only to prove the concept.

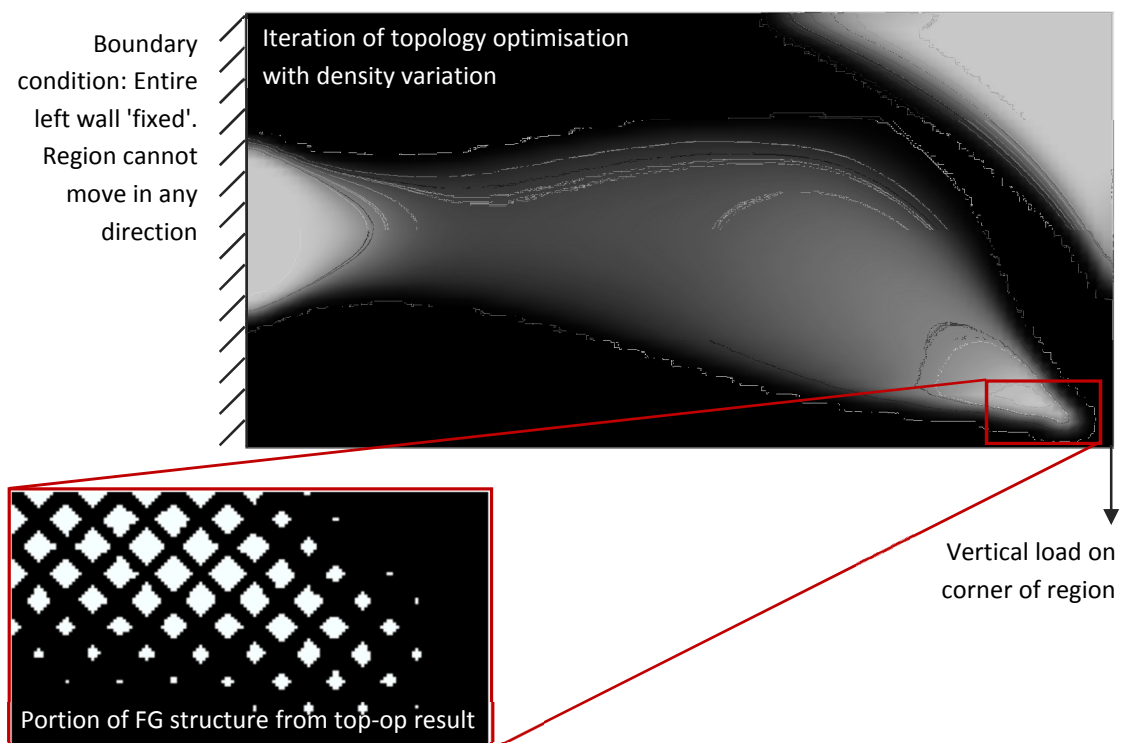


Figure 9-2: Integrating a density map from a topological optimisation approach into the conformal structure method

The pixel stepping of models as highlighted in the surface roughness testing would not exist if more advanced forms of boundary recognition tools were employed at the voxel-to-slice conversion stage. Vector graphics software such as Adobe Illustrator [184] provide tools to convert a bitmap image into 2D splines and polylines for further manipulation. The technique behind these could be investigated and implemented in the conformal structure method. However, testing did show that the effect of pixel stepping was a minimal component of surface roughness. Any bitmap to spline conversion process would need to be extremely quick, as it would be implemented for every slice of the model. Additionally the process would need to be robust - always producing a repeatable and closed-loop result to ensure manufacturability and continuity between slices.

## References

- [1] N. Hopkinson, R. J. M. Hague, and P. M. Dickens, Eds., *Rapid Manufacturing: An Industrial Revolution for the Digital Age*. Wiley, 2006.
- [2] I. Gibson, D. W. Rosen, and B. Stucker, *Additive Manufacturing Technologies: Rapid Prototyping to Direct Digital Manufacturing*. Springer, 2009.
- [3] K.-H. Grote, E. K. Antonsson, and (Eds.), *Springer Handbook of Mechanical Engineering*, vol. 10. Springer, 2009.
- [4] I. Gibson, "Rapid Prototyping: A Review," in *Virtual Modeling and Rapid Manufacturing*, P. J. Bártolo, Ed. Taylor & Francis/Balkema, 2005, pp. 7-18.
- [5] J. J. Beaman, J. W. Barlow, D. L. Bourell, R. H. Crawford, H. L. Marcus, and K. P. McAlea, *Solid Freeform Fabrication: a New Direction in Manufacturing*. Kluwer Academic Publishers, 1996.
- [6] P. C. Smith and A. E. W. Rennie, "Using Additive Manufacturing Effectively: A CAD Tool to Support Decision Making," in *Proceedings of the 36th International MATADOR Conference*, 2010, pp. 381-384.
- [7] S. Mansour and R. Hague, "Impact of rapid manufacturing on design for manufacture for injection moulding," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 217, no. 4, pp. 453-461, Jan. 2003.
- [8] R. A. Malloy, *Plastic Part Design for Injection Moulding: An Introduction*. Hanser Gardner Publications, 1994.
- [9] "Freedom Of Creation: Pioneers in 3D Printed Designs." [Online]. Available: <http://www.freedomofcreation.com/>. [Accessed: 17-Jan-2011].
- [10] "Shapeways: Passionate About Creating." [Online]. Available: <http://www.shapeways.com/>. [Accessed: 17-Nov-2011].
- [11] "Fraunhofer-Allianz Generative Fertigung: Bio-Medizin." [Online]. Available: <http://www.generativ.fraunhofer.de/index.php?l1=branchen&l2=b>. [Accessed: 20-Jan-2011].
- [12] D. Watts, "A Genetic Algorithm Based Topology Optimisation Approach for Exploiting Rapid Manufacturing's Design Freedom," Loughborough University, 2008.
- [13] K. Højbjerg, "Additive Manufacturing of Porous Metal Components," in *Proceedings of the 6th International Conference on Additive Manufacturing*, 2011.
- [14] "Festo Corporate - Bionic Handling Assistant." [Online]. Available: [http://www.festo.com/cms/en\\_corp/9655.htm](http://www.festo.com/cms/en_corp/9655.htm). [Accessed: 17-Jan-2011].
- [15] G. N. Levy, R. Schindel, and J. P. Kruth, "Rapid Manufacturing and Rapid Tooling with Layer Manufacturing (LM) Technologies, State of the Art and Future Perspectives," *CIRP Annals - Manufacturing Technology*, vol. 52, no. 2, pp. 589-609, 2003.

- [16] D. J. Brackett, I. Ashcroft, and R. J. M. Hague, "Topology Optimisation for Additive Manufacturing," in *In press*.
- [17] A. K. Kamrani and E. A. Nasr, *Collaborative Engineering: Theory and Practice*. Springer, 2008.
- [18] R. Hague, S. Mansour, and N. Saleh, "Material and design considerations for rapid manufacturing," *International Journal of Production Research*, vol. 42, no. 22, pp. 4691-4708, Nov. 2004.
- [19] C. K. Chua, K. F. Leong, and C. S. Lim, *Rapid Prototyping: Principles and Applications*, 2nd ed. World Scientific Publishing, 2004.
- [20] A. Liberman and H. Gothait, "Photopolymer material jetting in rapid prototyping," in *Virtual Modelling and Rapid Manufacturing*, P. J. Bártolo, Ed. 2005, pp. 355-360.
- [21] T. Grimm, *User's Guide to Rapid Prototyping*. Society of Manufacturing Engineers, 2004.
- [22] K. Mumtaz and N. Hopkinson, "Selective laser melting of Inconel 625 using pulse shaping," *Rapid Prototyping Journal*, vol. 16, no. 4, pp. 248-257, 2010.
- [23] "Tailored Injury Prevention and Performance Improvement for Protective Sports Garments (SCUTA)." [Online]. Available: <http://www.lboro.ac.uk/research/amrg/research/current/protective-sports-garments-scuta.html>. [Accessed: 02-May-2011].
- [24] G. Bingham, "The Generation of 3D Data for Rapid Manufactured Textiles," Loughborough University, 2007.
- [25] L. J. Gibson and M. F. Ashby, *Cellular Solids: Structure and Properties*, 2nd ed. Cambridge University Press, 1997.
- [26] N. J. Mills, C. Fitzgerald, A. Golchrist, and R. Verdejo, "Polymer foams for personal protection: cushions, shoes and helmets," *Composites Science and Technology*, vol. 63, pp. 2389-400, 2003.
- [27] J. Zhang and M. F. Ashby, "Mechanical selection of foams and honeycombs used for packaging and energy absorption," *Journal of Materials Science*, vol. 29, pp. 157-63, 1994.
- [28] Erg-Aerospace, "Duocel foam for impact absorption applications," 2007. [Online]. Available: <http://www.ergaerospace.com/foamproperties/applicationguide/energy.htm>. [Accessed: 04-Feb-2008].
- [29] C. Benning, *Plastic foams: The Physics and Chemistry of Product Performance and Process Technology*. New York: Wiley-Interscience, 1969.
- [30] G. Woods, *Flexible Polyurethane Foams: Chemistry and Technology*. Galliard (Printers) Ltd., 1982.
- [31] A. E. Simone and L. J. Gibson, "Aluminum Foams Produced by Liquid State Processes," *Acta Materialia*, vol. 46, no. 9, pp. 3109-23, 1998.
- [32] A. T. Huber and L. J. Gibson, "Anisotropy of Foams," *Journal of Materials Science*, vol. 23, pp. 3031-40, 1988.

- [33] H. X. Zhu, N. J. Mills, and J. F. Knott, "Analysis of the high strain compression of open-cell foams," *Journal of the Mechanics and Physics of Solids*, vol. 45, pp. 1875-904, 1997.
- [34] "Acoustic Properties of Metallic Foams." [Online]. Available: <http://www-diva.eng.cam.ac.uk/energy/acoustics/metalfoam.html>. [Accessed: 17-May-2011].
- [35] B. Wang, Z. Peng, Y. Zhang, and Y. Zhang, "Compressive response and energy absorption of foam EPDM," *Journal of Applied Polymer Science*, vol. 105, pp. 3462-9, 2007.
- [36] Z. Wang, H. Ma, L. Zhao, and G. Yang, "Studies on the dynamic compressive properties of open-cell aluminum alloy foams," *Key Engineering Materials*, vol. 306, pp. 905-10, 2006.
- [37] R. Verdejo and N. J. Mills, *Performance of EVA foam in running shoes*. Blackwell, 2002.
- [38] "EOS Literature." [Online]. Available: <http://www.eos.info/en/news-events/press-material/literature.html>. [Accessed: 22-Aug-2011].
- [39] J. Rakow and A. Waas, "Size Effects in Metal Foam Cores for Sandwich Structures," *AIAA Journal*, vol. 42, no. 7, pp. 1331-1337, Jul. 2004.
- [40] A. M. Kraynik, "The Structure of Random Foam," *Advanced Engineering Materials*, vol. 8, no. 9, pp. 900-6, 2006.
- [41] W. Thompson, "On the division of space with minimum partitional area," *Philosophical Magazine*, vol. 24, no. 151, p. 503, 1887.
- [42] K. Li, X. L. Gao, and G. Subhash, "Effects of cell shape and strut cross-sectional area variations on the elastic properties of three-dimensional open-cell foams," *Journal of the Mechanics and Physics of Solids*, vol. 54, pp. 783-806, 2006.
- [43] L. Gong, S. Kyriadekes, and N. Triantafyllidis, "On the stability of Kelvin cell foams under compressive loads," *Journal of the Mechanics and Physics of Solids*, vol. 53, no. 771-94, 2005.
- [44] N. J. Mills, "The high strain mechanical response of the wet Kelvin model for open-cell foams," *International Journal of Solids and Structures*, vol. 44, pp. 51-65, 2007.
- [45] L. Gong, S. Kyriakides, and W. Jang, "Compressive response of open-cell foams. Part I: Morphology and elastic properties," *International Journal of Solids and Structures*, vol. 42, no. 5-6, pp. 1355-1379, Mar. 2005.
- [46] W. E. Warren and A. M. Kraynik, "Linear elastic behavior of a low-density Kelvin foam with open cells," *Journal of Applied Mechanics*, vol. 64, pp. 787-94, 1997.
- [47] M. Janus-Michalska and R. B. Pecherski, "Macroscopic Properties of Open-Cell Foams Based on Micromechanical Modelling," *Technische Mechanik*, vol. 23, no. 2-4, pp. 234-44, 2003.
- [48] J. P. J. Brennan-Craddock, G. A. Bingham, R. J. M. Hague, and R. D. Wildman, "Impact Absorbent Rapid Manufactured Structures ( IARMS )," in *Proceedings of the Solid Freeform Fabrication Symposium*, 2008, pp. 266-277.

- [49] S. Tsopanos et al., "The Influence of Processing Parameters on the Mechanical Properties of Selectively Laser Melted Stainless Steel Microlattice Structures," *Journal of Manufacturing Science and Engineering*, vol. 132, no. 4, p. 041011, 2010.
- [50] H. Wang, Y. Chen, and D. W. Rosen, "A Hybrid Geometric Modeling Method for Large Scale Conformal Cellular Structures," *25th Computers and Information in Engineering Conference, Parts A and B*, vol. 3, pp. 421-427, 2005.
- [51] a Evans, "The topological design of multifunctional cellular metals," *Progress in Materials Science*, vol. 46, no. 3-4, pp. 309-327, 2001.
- [52] D. Dutta, F. B. Prinz, D. Rosen, and L. Weiss, "Layered Manufacturing: Current Status and Future Trends," *Journal of Computing and Information Science in Engineering*, vol. 1, no. 1, p. 60, 2001.
- [53] L. Mullen, R. C. Stamp, W. K. Brooks, E. Jones, and C. J. Sutcliffe, "Selective Laser Melting: a regular unit cell approach for the manufacture of porous, titanium, bone in-growth constructs, suitable for orthopedic applications.," *Journal of biomedical materials research. Part B, Applied biomaterials*, vol. 89, no. 2, pp. 325-34, May. 2009.
- [54] S. Yang, K.-F. Leong, Z. Du, and C.-K. Chua, "The design of scaffolds for use in tissue engineering. Part II. Rapid prototyping techniques," *Tissue engineering*, vol. 8, no. 1, pp. 1-11, Feb. 2002.
- [55] M. a Wettergreen, B. S. Bucklen, W. Sun, and M. a K. Liebschner, "Computer-aided tissue engineering of a human vertebral body," *Annals of biomedical engineering*, vol. 33, no. 10, pp. 1333-43, Oct. 2005.
- [56] C. Lam, "Scaffold development using 3D printing with a starch-based polymer," *Materials Science and Engineering: C*, vol. 20, no. 1-2, pp. 49-56, May. 2002.
- [57] M. W. Naing, C. K. Chua, K. F. Leong, and Y. Wang, "Fabrication of customised scaffolds using computer-aided design and rapid prototyping techniques," *Rapid Prototyping Journal*, vol. 11, no. 4, pp. 249-259, 2005.
- [58] H. N. G. Wadley, "Multifunctional periodic cellular metals," *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, vol. 364, no. 1838, pp. 31-68, Jan. 2006.
- [59] J. Lim and K. Kang, "Mechanical behavior of sandwich panels with tetrahedral and Kagome truss cores fabricated from wires," *International Journal of Solids and Structures*, vol. 43, no. 17, pp. 5228-5246, Aug. 2006.
- [60] A. Pasko, T. Vilbrandt, O. Fryazinov, and V. Adzhiev, "Procedural Function-Based Spatial Microstructures," in *Shape Modeling International Conference*, 2010, no. c, pp. 47-56.
- [61] R. Hague, S. Mansour, and N. Saleh, "Design opportunities with rapid manufacturing," *Assembly Automation*, vol. 23, no. 4, pp. 346-356, 2003.
- [62] M. Burns, *Automated Fabrication: improving productivity in manufacturing*. Prentice Hall, 1993.



- [63] C. C. Kai, G. G. K. Jacob, and T. Mei, *Interface between CAD and rapid prototyping systems. Part 1: a study of existing interfaces*, vol. 13. 1997, pp. 566-570.
- [64] H. Zhu, "B-Rep model simplification by automatic fillet/round suppressing for efficient automatic feature recognition," *Computer-Aided Design*, vol. 34, no. 2, pp. 109-123, Feb. 2002.
- [65] C. Schroeder, W. C. Regli, A. Shokoufandeh, and W. Sun, "Computer-aided design of porous artifacts," *Computer-aided design*, vol. 37, no. 3, pp. 339–353, Mar. 2005.
- [66] D. Baldwin, "Surface Reconstruction from Constructive Solid Geometry for Interactive Visualization," in *Advances in Visual Computing: Third International Symposium, ISVC 2007, Lake Tahoe, NV, USA, November 2007. Proceedings, Part 1*, 2007, pp. 321-330.
- [67] D. Marsh, *Applied Geometry for Computer Graphics and CAD*, 2nd ed. Springer, 2004.
- [68] S. Ghali, *Introduction to Geometric Computing*. Springer, 2008.
- [69] C. R. Alavala, *Computer Graphics*. PHI Learning Private Ltd., 2009.
- [70] J. Vuoskoski, "Exchange of Product Data between CAD systems and a Physics Simulation Program," Tampere University of Technology, 1996.
- [71] C. Chu, G. Graf, and D. W. Rosen, "Design for Additive Manufacturing of Cellular Structures," *Computer-Aided Design and Applications*, vol. 5, no. 5, pp. 686-696, 2008.
- [72] M. K. Agoston, *Computer Graphics and Geometric Modeling: Implementation and Algorithms*. Springer, 2005.
- [73] E. A. Nasr and A. K. Kamrani, *Computer-Based Design and Manufacturing: an Information-Based Approach*. Springer, 2007.
- [74] D. E. LaCourse, *Handbook of Solid Modeling*. McGraw-Hill, 1995.
- [75] M. S. Tawfik, "An efficient algorithm for CSG to b-rep conversion," in *Proceedings of the first ACM symposium on Solid modeling foundations and CAD/CAM applications*, 1991, pp. 99–108.
- [76] U. Roy and C. R. Liu, "Feature-Based Representational Scheme of a Solid Modeler for Providing Dimensioning and Tolerancing Information," *Robotics & Computer-Integrated Manufacturing*, vol. 4, no. 3, pp. 335-345, Feb. 1988.
- [77] A. J. P. Gomes and J. G. Teixeira, "Form feature modelling in a hybrid CSG/BRep scheme," *Computers & Graphics*, vol. 15, no. 2, pp. 217-229, 1991.
- [78] D. Ushakov, *Variational Direct Modeling: How to Keep Design Intent in History-Free CAD (white paper)*. 2008.
- [79] G. Chu, G. Brady, W. Miao, J. Halloran, S. Hollister, and D. Brei, "Ceramic SFF by direct and indirect stereolithography," in *Materials Research Society Symposium Proceedings*, 1999, vol. 542, no. 7, pp. 119–124.

- [80] C. M. Langton, M. a Whitehead, D. K. Langton, and G. Langley, "Development of a cancellous bone structural model by stereolithography for ultrasound characterisation of the calcaneus.," *Medical engineering & physics*, vol. 19, no. 7, pp. 599-604, Oct. 1997.
- [81] C. M. Cheah, C. K. Chua, K. F. Leong, and S. W. Chua, "Development of a Tissue Engineering Scaffold Structure Library for Rapid Prototyping. Part 2: Parametric Library and Assembly Program," *The International Journal of Advanced Manufacturing Technology*, vol. 21, no. 4, pp. 302-312, Feb. 2003.
- [82] V. R. Gervasi and D. C. Stahl, "Design and fabrication of components with optimized lattice microstructures," in *Proceedings of the Solid Freeform Fabrication Symposium*, 2004, pp. 838-844.
- [83] V. Chandru and S. Manohar, "Voxel-based modeling for layered manufacturing," *IEEE Computer Graphics and*, vol. 15, no. 6, pp. 42-47, 1995.
- [84] M. Mäntylä, *An Introduction to Solid Modeling*. Computer Science Press, 1988.
- [85] H. Wang and D. W. Rosen, "Parametric Modeling Method for Truss Structures," in *Proceedings of DETC'02 2002 ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2002, pp. 759-767.
- [86] M. Bhandarkar, "STEP-based feature extraction from STEP geometry for Agile Manufacturing," *Computers in Industry*, vol. 41, no. 1, pp. 3-24, Jan. 2000.
- [87] C. Elanchezhian, T. S. Selwyn, and G. S. Sundar, *Computer Aided Manufacturing*, 2nd ed. Laxmi Publications (P) Ltd., 2007.
- [88] Y. Chen, "3D Texture Mapping for Rapid Manufacturing," *Computer-Aided Design & Applications*, vol. 4, no. 6, pp. 761-771, 2007.
- [89] H. T. Yau, C. C. Kuo, and C. H. Yeh, "Extension of surface reconstruction algorithm to the global stitching and repairing of STL models," *Computer-Aided Design*, vol. 35, no. 5, pp. 477-486, Apr. 2003.
- [90] W. Zhu, "A Visibility Sphere Marching algorithm of constructing polyhedral models for haptic sculpting and product prototyping," *Robotics and Computer-Integrated Manufacturing*, vol. 21, no. 1, pp. 19-36, Feb. 2005.
- [91] D.-xing Wang, D.-ming Guo, Z.-yuan Jia, and H.-wen Leng, "Slicing of CAD models in color STL format," *Computers in Industry*, vol. 57, pp. 3-10, Oct. 2005.
- [92] "Materialise: replace solid parts with metal lattice structures." [Online]. Available: <http://www.materialise.com/materialise/view/en/1979905-Metal+Structures.html>. [Accessed: 17-Jan-2011].
- [93] "Marcam Engineering -Software solutions for Rapid Technologies - AutoFab." [Online]. Available: <http://www.marcam.de/cms/autofab.82.en.html>. [Accessed: 17-Jan-2011].
- [94] Y. Chen, "A Mesh-based Geometric Modeling Method for General Structures," in *2006 ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Philadelphia, PA*, 2006.

- [95] V. R. Gervasi, "Net Shape Composites using SLA TetraCast Patterns," in *Solid Freeform Fabrication Symposium*, 1997, pp. 149-157.
- [96] "Rapid Prototyping Research - Milwaukee School of Engineering." [Online]. Available: [http://www.msoe.edu/academics/research\\_centers/rpc/research.shtml](http://www.msoe.edu/academics/research_centers/rpc/research.shtml). [Accessed: 18-Jan-2011].
- [97] H. Chow, S. Tan, and W. Sze, "Layered Modeling of Porous Structures with Voronoi Diagrams," *Computer-Aided Design & Applications*, vol. 4, no. 1-4, pp. 321-330, 2007.
- [98] S.-H. Huang, L.-C. Zhang, and M. Han, "An Effective Error-Tolerance Slicing Algorithm for STL Files," *The International Journal of Advanced Manufacturing Technology*, vol. 20, no. 5, pp. 363-367, Sep. 2002.
- [99] N. Alves and P. Bartolo, "Integrated computational tools for virtual and physical automatic construction," *Automation in Construction*, vol. 15, no. 3, pp. 257-271, May. 2006.
- [100] V. Kumar, "An assessment of data formats for layered manufacturing," *Advances in Engineering Software*, vol. 28, no. 3, pp. 151-164, Apr. 1997.
- [101] C. C. Kai, G. G. K. Jacob, and T. Mei, "Interface between CAD and Rapid Prototyping Systems. Part 2: LMI - An Improved Interface," *The International Journal of Advanced Manufacturing Technology*, vol. 13, no. 8, pp. 571-576, 1997.
- [102] X. Wu, "Voxel-based model and its application in advanced manufacturing," *Proceedings of SPIE*, vol. 5444, pp. 383-388, 2004.
- [103] "Common Layer Interface (CLI): Version 2.0 Specification." .
- [104] M. Vatani, A. Rahimi, F. Brazandeh, and A. Sanati Nezhad, "An enhanced slicing algorithm using nearest distance analysis for layer manufacturing," *Proceedings of World Academy Science, Engineering and Technology*, vol. 15, pp. 721-726, 2009.
- [105] T. Plachetka, "POV Ray: Persistence of Vision Parallel Raytracer," in *Proc. of Spring Conf. on Computer Graphics, Budmerice, Slovakia*, 1998, pp. 123-129.
- [106] G. M. Ovidiu, T.-T. Mirela, P. Radu, and V. Dinu, "Influence of the Lattice Structures on the Mechanical Behavior of Hip Endoprostheses," *2010 Advanced Technologies for Enhancing Quality of Life*, pp. 6-11, Jul. 2010.
- [107] "netfabb: Selective Space Structures." [Online]. Available: <http://www.netfabb.com/structure.php>. [Accessed: 17-Jan-2011].
- [108] W. Brooks, C. Sutcliffe, W. Cantwell, P. Fox, J. Todd, and R. Mines, "Rapid design and manufacture of ultralight cellular materials," in *Proceedings of the Solid Freeform Fabrication Symposium*, 2005, pp. 231-241.
- [109] L. Mullen, R. C. Stamp, P. Fox, E. Jones, C. Ngo, and C. J. Sutcliffe, "Selective laser melting: a unit cell approach for the manufacture of porous, titanium, bone in-growth constructs, suitable for orthopedic applications. II. Randomized structures.," *Journal of biomedical materials research. Part B, Applied biomaterials*, vol. 92, no. 1, pp. 178-88, Jan. 2010.

- [110] D. W. Hutmacher, "Scaffolds in tissue engineering bone and cartilage," *Biomaterials*, vol. 21, no. 24, pp. 2529-43, Dec. 2000.
- [111] M. Too et al., "Investigation of 3D non-random porous structures by fused deposition modelling," *The International Journal of Advanced Manufacturing Technology*, vol. 19, no. 3, pp. 217-223, 2002.
- [112] K. C. Ang, K. F. Leong, C. K. Chua, and M. Chandrasekaran, "Investigation of the mechanical properties and porosity relationships in fused deposition modelling-fabricated porous structures," *Rapid Prototyping Journal*, vol. 12, no. 2, pp. 100-105, 2006.
- [113] R. Stamp, P. Fox, W. O'Neill, E. Jones, and C. Sutcliffe, "The development of a scanning strategy for the manufacture of porous biomaterials by selective laser melting," *Journal of materials science. Materials in medicine*, vol. 20, no. 9, pp. 1839-48, Sep. 2009.
- [114] G. Jacob, "Development of a new rapid prototyping interface," *Computers in Industry*, vol. 39, no. 1, pp. 61-70, Jun. 1999.
- [115] I. Stroud and P. Xirouchakis, "STL and Extensions," *Advances in Engineering Software*, vol. 31, no. 2, pp. 83-95, Feb. 2000.
- [116] J. D. Hiller and H. Lipson, "STL 2.0: A Proposal for a Universal Multi-Material Additive Manufacturing File Format," in *Solid Freeform Fabrication Symposium (SFF'09)*, 2009, no. 1, pp. 266-278.
- [117] W. Chiu and S. Tan, "Multiple material objects: from CAD representation to data format for rapid prototyping," *Computer-aided design*, vol. 32, no. 12, pp. 707-717, Oct. 2000.
- [118] S. J. Rock and M. J. Wozny, "A flexible file format for solid freeform fabrication," in *Solid Freeform Fabrication Symposium Proceedings, University of Texas, Austin, TX*, 1991, pp. 1-12.
- [119] "Standard Specification for Additive Manufacturing File Format (AMF) Draft 0.45," *Structure*.
- [120] S. Gibson, "Beyond Volume Rendering: Visualization, Haptic Exploration, and Physical Modeling of Voxel-based Objects," *Proceedings of Eurographics Workshop on Visualization in Scientific Computing*. pp. 10-24, 1995.
- [121] A. E. Kaufman, "Volume visualization of the ascending thoracic aorta using isotropic MDCT data: protocol optimization," *ACM Computing Surveys (CSUR)*, vol. 28, no. 1, pp. 165-167, Nov. 1996.
- [122] H. Jones, *Computer Graphics through Key Mathematics*. Springer, 2001.
- [123] T. R. Jackson, W. Cho, N. M. Patrikalakis, and E. M. Sachs, "Memory Analysis of Solid Model Representations for Heterogeneous Objects," *Journal of Computing and Information Science in Engineering*, vol. 2, no. 1, p. 1, 2002.
- [124] M. Chen, A. Kaufman, and R. Yagel, *Volume Graphics*. Springer, 2000.

- [125] F. Guo, L. Wang, and D. Dong, "Human Head 3D Dimensions Measurement for the Design of Helmets," in *Digital Human Modeling: Second International Conference, ICDHM 2009, 2009*, pp. 624-631.
- [126] S. J. Hollister, R. A. Levy, T. M. Chu, J. W. Halloran, and S. E. Feinberg, "An image-based approach for designing and manufacturing craniofacial scaffolds," *International Journal of Oral & Maxillofacial Surgery*, vol. 29, no. 1, pp. 67-71, 2000.
- [127] E. Verges, D. Ayala, S. Grau, and D. Tost, "3D Reconstruction and Quantification of Porous Structures," *Computers & Graphics*, vol. 32, no. 4, pp. 438-444, Aug. 2008.
- [128] S. Vanis, O. Rheinbach, a Klawonn, O. Prymak, and M. Epple, "Numerical computation of the porosity of bone substitution materials from synchrotron micro computer tomographic data," *Materialwissenschaft und Werkstofftechnik*, vol. 37, no. 6, pp. 469-473, Jun. 2006.
- [129] W. Sun, B. Starly, J. Nam, and A. Darling, "Bio-CAD modeling and its applications in computer-aided tissue engineering," *Computer-Aided Design*, vol. 37, no. 11, pp. 1097-1114, Sep. 2005.
- [130] J. D. Foley, V. Dam, Feiner, and Hughes, *Computer Graphics: Principles and Practice*, 2nd ed. Addison-Wesley, 1997.
- [131] T. A. Galyean, "Sculpting: An interactive volumetric modeling technique," *Computer Graphics*, vol. 25, no. 4, pp. 267-274, 1991.
- [132] T. Vilbrandt, E. Malone, H. Lipson, and A. Pasko, "Universal Desktop Fabrication," *Heterogeneous objects modelling and applications: collection of papers on foundations and practice*, p. 259, 2008.
- [133] I. Zied, *CAD/CAM Theory and Practice*. McGraw-Hill, 1991.
- [134] S. W. Wang and A. E. Kaufman, "Volume-Sampled 3D Modeling," *IEEE Computer Graphics and Applications*, vol. 14, no. 5, pp. 26-32, 1994.
- [135] "3D-Coat - Voxel sculpting, Retopology, UV-mapping, Texture painting." [Online]. Available: <http://www.3d-coat.com/>. [Accessed: 22-Jan-2011].
- [136] "Welcome to Voxelogic." [Online]. Available: <http://www.voxellogic.com/>. [Accessed: 22-Jan-2011].
- [137] A. Pasko, V. Adzhiev, A. Sourin, and V. Savchenko, "Function representation in geometric modeling: concepts, implementation and applications," *The Visual Computer*, vol. 11, no. 8, pp. 429-446, Aug. 1995.
- [138] Q. Liu and A. Sourin, "Function-based representation of complex geometry and appearance," in *Proceedings of the tenth international conference on 3D Web technology*, 2005, vol. 1, no. 212, pp. 123-134.
- [139] O. Fryazinov and A. Pasko, "Interactive ray shading of FRep objects," in *WSCG 2008 Communications Papers Proceedings*, 2008, pp. 145-152.
- [140] "HyperFun." [Online]. Available: <http://www.hyperfun.org/wiki/doku.php?id=main>. [Accessed: 18-Jan-2011].

- [141] A. Pasko and V. Adzhiev, "Constructive function-based modeling in multilevel education," *Communications of the ACM*, vol. 52, no. 9, pp. 118–122, 2009.
- [142] G. Y. Gardner, "Simulation of natural scenes using textured quadric surfaces," *ACM SIGGRAPH Computer Graphics*, vol. 18, no. 3, pp. 11–20, 1984.
- [143] J. P. Lewis, "Algorithms for solid noise synthesis," *ACM SIGGRAPH Computer Graphics*, vol. 23, no. 3, pp. 263–270, Jul. 1989.
- [144] J. Bloomenthal, "Polygonization of implicit surfaces," *Computer Aided Geometric Design*, vol. 5, no. 4, pp. 341–355, Nov. 1988.
- [145] A. S. Glassner, Ed., *An Introduction to Ray Tracing*. Morgan Kaufmann, 2002.
- [146] M. Dickinson and R. Nay, *Modulus Mapping of a Metal Oxide Film*. .
- [147] D. W. Rosen, "Computer-aided design for additive manufacturing of cellular structures," *Computer-Aided Design & Applications*, vol. 4, no. 5, pp. 585–594, 2007.
- [148] G. E. Farin, *Curves and surfaces for CAGD: a practical guide*, 5th ed. Academic Press, 2002.
- [149] J. Fish and T. Belytschko, *A First Course in Finite Elements*. John Wiley & Sons, 2007.
- [150] C. T. F. Ross, *Finite Element Methods in Structural Mechanics*. Ellis Horwood Ltd, 1985.
- [151] V. Adams, *A Designer's Guide to Simulation with Finite Element*. NAFEMS, 2008.
- [152] D. M. Yan, W. Wang, B. Lévy, and Y. Liu, "Efficient Computation of 3D Clipped Voronoi Diagram," *Advances in Geometric Modeling and Processing*, pp. 269–282, 2010.
- [153] F. Nielsen and R. Nock, "Hyperbolic Voronoi Diagrams Made Easy," *2010 International Conference on Computational Science and Its Applications*, pp. 74–80, 2010.
- [154] F. Aurenhammer, "Voronoi diagrams - a survey of a fundamental geometric data structure," *ACM Computing Surveys (CSUR)*, vol. 23, no. 3, pp. 345–405, Sep. 1991.
- [155] L. P. Chew and R. L. (Scot) Dyrsdale, "Voronoi diagrams based on convex distance functions," *Proceedings of the first annual symposium on Computational geometry - SCG '85*, vol. 75, pp. 235–244, 1985.
- [156] O. Watanabe, H. M. Zbib, and E. Takenouchi, "Crystal plasticity: micro-shear banding in polycrystals using Voronoi tessellation," *International Journal of Plasticity*, vol. 14, no. 8, pp. 771–788, 1998.
- [157] F. Fritzen, T. Böhlke, and E. Schnack, "Periodic three-dimensional mesh generation for crystalline aggregates based on Voronoi tessellations," *Computational Mechanics*, vol. 43, no. 5, pp. 701–713, Oct. 2008.
- [158] F. Dupuis, J.-F. Sadoc, and J.-P. Mornon, "Protein secondary structure assignment through Voronoi tessellation.," *Proteins*, vol. 55, no. 3, pp. 519–28, May. 2004.

- [159] W. Brostow, M. Chybicki, R. Laskowski, and J. Rybicki, "Voronoi polyhedra and Delaunay simplexes in the structural analysis of molecular-dynamics-simulated materials," *Physical Review B*, vol. 57, no. 21, pp. 13448-13458, Jun. 1998.
- [160] C. Rycroft, G. Grest, J. Landry, and M. Bazant, "Analysis of granular flow in a pebble-bed nuclear reactor," *Physical Review E*, vol. 74, no. 2, pp. 1-16, Aug. 2006.
- [161] S. Mulders, "Archive of Voronoi for Maya." [Online]. Available: <http://www.digisan.nl/dse/?cat=11>. [Accessed: 17-Jan-2011].
- [162] "MATLAB - The Language of Technical Computing." [Online]. Available: <http://www.mathworks.com/products/matlab/>. [Accessed: 23-Jan-2011].
- [163] H. J. Weber and G. B. Arfken, *Essential Mathematical Methods for Physicists*. Academic Press, 2003.
- [164] R. Goldman, *Pyramid Algorithms: A Dynamic Programming Approach to Curves and Surfaces for Geometric Modeling*. Morgan Kaufmann, 2003.
- [165] L. O'Gorman, M. J. Sammon, and M. Seul, *Practical Algorithms for Image Analysis*, 2nd ed. Cambridge University Press, 2008.
- [166] "Freesteel." [Online]. Available: <http://www.freesteel.co.uk/wpblog/frontpage/>. [Accessed: 23-Jan-2011].
- [167] I. Gibson and D. Shi, "Material properties and fabrication parameters in selective laser sintering process," *Rapid Prototyping Journal*, vol. 3, no. 4, pp. 129-136, 1997.
- [168] G. M. Fadel and C. Kirschman, "Accuracy issues in CAD to RP translations," *Rapid Prototyping Journal*, vol. 2, no. 2, pp. 4-17, 1996.
- [169] J. M. Coupland and J. Lobera, "Measurement of Steep Surfaces Using White Light Interferometry," *Strain*, vol. 46, no. 1, pp. 69-78, Feb. 2010.
- [170] "EOS Plastic Laser Sintering Systems." [Online]. Available: <http://www.eos.info/en/products/systems-equipment/plastic-laser-sintering-systems.html>. [Accessed: 17-Jan-2011].
- [171] R. N. Bracewell, "The Fourier Transform," *Scientific American*, pp. 86-95.
- [172] D. Sundararajan, *The Discrete Fourier Transform: Theory, Algorithms and Applications*. World Scientific Publishing, 2001.
- [173] S. Gade, N. Thrane, H. Konstantin-Hansen, and J. Wismer, "Time Windows." Brüel & Kjær.
- [174] R. W. Ramirez, *The FFT Fundamentals and Concepts*. Prentice Hall, 1985.
- [175] EOS, "Material data sheet PA 2200." .
- [176] J. Ramanathan, *Methods of Applied Fourier Analysis*. Birkhäuser Boston, 1998.
- [177] A. F. Harvey and M. Cerna, "The Fundamentals of FFT-Based Signal Analysis and Measurement in LabVIEW and LabWindows." National Instruments, 1993.

- [178] K. C. Tan, H. Lim, and B. Tan, "Windowing techniques for image restoration," *CVGIP: Graphical Models and Image Processing*, vol. 53, no. 5, pp. 491–500, 1991.
- [179] Y. Ge, Q. Cheng, and S. Zhang, "Reduction of edge effects in spatial information extraction from regional geochemical data: a case study based on multifractal filtering technique," *Computers & Geosciences*, vol. 31, no. 5, pp. 545-554, Jun. 2005.
- [180] J. Arrillaga and N. R. Watson, *Power system harmonics*, 2nd ed. John Wiley & Sons, 2003.
- [181] H. Hauser, E. Groller, and T. Theussl, "Mastering Windows: Improving Reconstruction," *2000 IEEE Symposium on Volume Visualization (VV 2000)*, pp. 101-108, Oct. 2000.
- [182] "Understanding FFT Windows." LDS Dactron, 2003.
- [183] R. Albert and T. Breedove, *C++: An Active Learning Approach*. Jones and Bartlett Publishers, 2009.
- [184] "Adobe." [Online]. Available: <http://www.adobe.com/>. [Accessed: 22-Jan-2011].