NURBS-compatible subdivision surfaces

Thomas J. Cashman

February 2010

University of Cambridge
Computer Laboratory
Queens' College

This dissertation is submitted for the degree of Doctor of Philosophy

delivered by
EUROGRAPHICS DIGITAL LIBRARY
www.eg.org diglib.eg.org

Abstract

Two main technologies are available to design and represent freeform surfaces: Non-Uniform Rational B-Splines (NURBS) and subdivision surfaces. Both representations are built on uniform B-splines, but they extend this foundation in incompatible ways, and different industries have therefore established a preference for one representation over the other. NURBS are the dominant standard for Computer-Aided Design, while subdivision surfaces are popular for applications in animation and entertainment. However there are benefits of subdivision surfaces (arbitrary topology) which would be useful within Computer-Aided Design, and features of NURBS (arbitrary degree and non-uniform parametrisations) which would make good additions to current subdivision surfaces.

I present NURBS-compatible subdivision surfaces, which combine topological freedom with the ability to represent any existing NURBS surface exactly. Subdivision schemes that extend either non-uniform *or* general-degree B-spline surfaces have appeared before, but this dissertation presents the first surfaces able to handle both challenges simultaneously. To achieve this I develop a novel factorisation of knot insertion rules for non-uniform, general-degree B-splines.

Many subdivision surfaces have poor second-order behaviour near singularities. I show that it is possible to bound the curvatures of the general-degree subdivision surfaces created using my factorisation. Bounded-curvature surfaces have previously been created by 'tuning' uniform low-degree subdivision schemes; this dissertation shows that general-degree schemes can be tuned in a similar way. As a result, I present the first general-degree subdivision schemes with bounded curvature at singularities.

Previous subdivision schemes, both uniform and non-uniform, have inserted knots indiscriminately, but the factorised knot insertion algorithm I describe in this dissertation grants the flexibility to insert knots selectively. I exploit this flexibility to preserve convexity in highly non-uniform configurations, and to create locally uniform regions in place of non-uniform knot intervals. When coupled with bounded-curvature modifications, these techniques give the first non-uniform subdivision schemes with bounded curvature.

I conclude by combining these results to present NURBS-compatible subdivision surfaces: arbitrary-topology, non-uniform and general-degree surfaces which guarantee high-quality second-order surface properties.

Acknowledgements

While working towards the PhD degree I have been spurred on by friends, colleagues and mentors; some people, to their credit, have been all three at once. I am indebted to my supervisor, Neil Dodgson, who has always been available for guidance, and who gave me the confidence to publish my ideas. With Ursula Augsdörfer I have shared many conversations on subdivision surfaces but many more games of chess, and I am grateful for both. I have also been fortunate to work with Malcolm Sabin, who is a source of knowledge and inspiration on any subject with a link to geometry, as well as a model of scientific curiosity and humility. I thank all three for fruitful conversations and sound advice.

Christian Richardt has cheerfully shared an office with me for several years, and others in the Rainbow research group have been only too willing to lend their time and experience. These kind souls include Richard Southern, Ian Davies and Alan Blackwell, all of whom helped me to prepare materials that demonstrate NURBS-compatible subdivision surfaces in action. I also thank Daniel Bates, who helped me to discover an error in Table 5.3 and Figure 5.4 that I was able to correct before they appeared here (or in the SIGGRAPH paper for which they were headed at the time).

This research would not have been possible without the financial support of the EPSRC. I am also grateful for the Steven Thomas studentship provided by Queens' College, which allowed me to attend several conferences to present this work to the international subdivision community. That community has provided useful feedback; I particularly thank the anonymous authors of twenty reviews which I received as a result of submitting papers on the work presented here. Their comments have enriched the presentation and analysis of my ideas.

Lastly, I thank my family and friends, who have been a constant support and encouragement. Above all, this dissertation belongs to Sarah, who loved me unreservedly through it all, and never stopped believing that I could do it.

For Sarah
In honour of Spirograph and metal cheese

Copyright material

I am grateful to the following copyright holders for allowing their work to appear in this dissertation:

Figure 1.1 © Todd Randall Jordan
http://www.flickr.com/photos/tojosan/128627213/
Used under a Creative Commons Attribution-Noncommercial-Share Alike 2.0 License (http://creativecommons.org/licenses/by-nc-sa/2.0/)

Figure 2.1 © Edson International
http://www.westlawn.edu/news/index.asp?displayfile=EdsonSplineWgtsWEB.htm
Used with permission

Figure 2.7 © Blender Foundation
http://www.bigbuckbunny.org
Used under a Creative Commons Attribution 3.0 License (http://creativecommons.org/licenses/by/3.0/)

Publications

This dissertation presents research that has also been published in the following papers:

T. J. Cashman, U. H. Augsdörfer, N. A. Dodgson and M. A. Sabin. NURBS with Extraordinary Points: High-degree, Non-uniform, Rational Subdivision Schemes. *ACM Transactions on Graphics*, 28(3):#46, 1–9, 2009.

U. H. Augsdörfer, T. J. Cashman, N. A. Dodgson and M. A. Sabin. Numerical Checking of C^1 for Arbitrary Degree Subdivision Schemes based on Quadrilateral Meshes. In *13th IMA Conference on the Mathematics of Surfaces*, volume 5654 of *Lecture Notes in Computer Science*, pages 45–54. Springer, 2009.

T. J. Cashman, N. A. Dodgson and M. A. Sabin. Selective knot insertion for symmetric, non-uniform refine and smooth B-spline subdivision. *Computer Aided Geometric Design*, 26(4):472–479, 2009.

T. J. Cashman, N. A. Dodgson and M. A. Sabin. A symmetric, non-uniform, refine and smooth subdivision algorithm for general degree B-splines. *Computer Aided Geometric Design*, 26(1):94–104, 2009.

T. J. Cashman, N. A. Dodgson and M. A. Sabin. Non-uniform B-Spline Subdivision Using Refine and Smooth. In *12th IMA Conference on the Mathematics of Surfaces*, volume 4647 of *Lecture Notes in Computer Science*, pages 121–137. Springer, 2007.

Contents

List of Figures **11**

List of Tables **13**

1 Introduction **15**
- 1.1 NURBS and subdivision surfaces . . . 16
- 1.2 An arbitrary topology superset of NURBS . . . 17
- 1.3 Thesis . . . 18

2 Background **19**
- 2.1 B-splines . . . 19
  - 2.1.1 Knot vectors: uniform and non-uniform . . . 21
  - 2.1.2 NURBS . . . 22
  - 2.1.3 Uniform B-spline subdivision . . . 23
  - 2.1.4 Knot insertion . . . 23
- 2.2 Uniform subdivision surfaces . . . 23
  - 2.2.1 Generalising knot insertion . . . 24
  - 2.2.2 Schemes based on B-splines . . . 25
  - 2.2.3 Schemes not based on B-splines . . . 25
  - 2.2.4 Refine and smooth . . . 26
- 2.3 Non-uniform subdivision surfaces . . . 28
  - 2.3.1 NURSS . . . 28
  - 2.3.2 Specialised applications . . . 28
- 2.4 Summary . . . 29

3 Non-uniform refine and smooth **31**
- 3.1 Limitations and requirements . . . 31
- 3.2 Polar form . . . 33
  - 3.2.1 Knot insertion . . . 35
  - 3.2.2 Schaefer's algorithm . . . 35
- 3.3 Symmetric algorithm . . . 36
  - 3.3.1 Problem statement . . . 37
  - 3.3.2 Overview . . . 37
  - 3.3.3 The refine stage . . . 38
  - 3.3.4 A smoothing stage . . . 39
  - 3.3.5 Examples . . . 41

3.3.6 Proof . . . 41
3.3.7 End conditions . . . 44
3.4 Summary . . . 45

4 Extraordinary vertices **47**
4.1 Incorporating knot intervals . . . 48
4.2 Generalised knot insertion . . . 49
4.3 Knot insertion strategy . . . 51
4.3.1 Subdividing large intervals first . . . 53
4.3.2 Creating uniform extraordinary regions . . . 55
4.4 Summary and discussion . . . 59

5 Bounded curvature **61**
5.1 Continuity of subdivision surfaces . . . 61
5.1.1 Proving C^1 . . . 63
5.1.2 Bounded curvature . . . 64
5.1.3 C^2 schemes . . . 65
5.1.4 Tuning . . . 66
5.2 Existing high-degree schemes . . . 66
5.3 Bounded curvature for saddle components . . . 68
5.4 Bounded curvature for cup component . . . 72
5.4.1 Negative results . . . 73
5.4.2 Additional smoothing stage . . . 76
5.5 Polar artifacts . . . 77
5.6 Proving C^1 continuity . . . 79
5.7 Summary . . . 81

6 Conclusion **83**
6.1 Contributions . . . 83
6.2 Applications . . . 85
6.3 Limitations . . . 87
6.4 Even degrees . . . 89
6.5 Future work . . . 91

Bibliography **93**

List of Figures

1.1 The 'Cloud Gate' sculpture in Millennium Park, Chicago 15
1.2 Example NURBS surface 16
1.3 The incompatibility between NURBS and existing subdivision surfaces 17

2.1 A traditional spline used for drawing smooth curves 19
2.2 Uniform B-spline basis functions from degree 1 to 6 20
2.3 Cubic B-spline basis functions for two different knot vectors 21
2.4 Cubic B-splines using the same control points, but two different knot vectors 21
2.5 Chaikin's algorithm to construct a uniform quadratic B-spline 22
2.6 A variant of Catmull-Clark subdivision acting on a cube 24
2.7 Animated characters created using Catmull-Clark subdivision surfaces 25
2.8 Regular refinement patterns for important subdivision schemes 27
2.9 An overview of subdivision surfaces based on B-splines 29

3.1 The constraints placed by property B on quintic knot insertion 34
3.2 The polar form of a B-spline 34
3.3 The need for a symmetric algorithm 36
3.4 Knot insertion order for non-uniform symmetric refine and smooth 38
3.5 Polar arguments for a point through the sequence of refine and smooth stages 39
3.6 The four smoothing actions that form a point P_i^σ 40
3.7 An example factorisation for a non-uniform quintic B-spline with multiple knots . . 42
3.8 The example from §3.3.5 in full; see Figure 3.7 for an example application 42
3.9 Refine-and-smooth factorisation for odd d in the unselective case 43
3.10 Refine-and-smooth factorisation for even d in the unselective case 44

4.1 Example refinement patterns for irregularities in the control mesh 47
4.2 Local knot vectors for a mesh containing extraordinary vertices 49
4.3 Calculating univariate smoothing stages one edge at a time 50
4.4 Calculating bivariate smoothing stages one face at a time 50
4.5 An example refine stage for degree 3 50
4.6 Failure of the direct midpoint knot insertion strategy 52
4.7 Removing the problem highlighted by Figure 4.6 52
4.8 Knot vectors can be independent of knot intervals in vertex-connected faces 53
4.9 The effect of three different knot insertion strategies 54
4.10 The knot insertion strategy in §4.3 for an example non-uniform configuration 56
4.11 The interaction of multiple extraordinary vertices while creating uniform regions . . 57

4.12 An equivalence class contains extraordinary vertices connected by emanating rays . . 57
4.13 Algorithm to assign each extraordinary vertex to an equivalence class 58
4.14 Pseudocode for non-uniform general-degree subdivision 58

5.1 Eigenbasis functions for the centroid-averaging schemes at degree 9 and valency 8 . . 67
5.2 Five different bounded-curvature settings for β_8^5 and γ_8^5 69
5.3 Finding an approximation to a stable natural configuration 70
5.4 Bounded-curvature solutions for α_n^d, β_n^d and γ_n^d 71
5.5 Problems that arise when trying to use α to satisfy $\lambda^2 = \mu_0$ for $n = 3$ 73
5.6 Modifying the refine stage to achieve bounded curvature for degree 5 and valency 3 . 74
5.7 Final affine combination used to position 3-valent vertices for bounded curvature . . 76
5.8 The quadratic natural configuration for bounded-curvature subdivision schemes . . . 78
5.9 Characteristic rings resulting from bounded-curvature modifications 79
5.10 Surfaces defined by a control mesh with 12-valent vertices 80

6.1 The same control mesh subdivided in three different configurations 86
6.2 Non-uniform knot intervals affect the projection of a surface's parametrisation . . . 86
6.3 The Catmull-Clark surface defined on the control mesh used in Table 6.1 88

List of Tables

5.1 The ratios μ_0/λ^2 and μ_2/λ^2 for centroid-averaging schemes 67
5.2 β_n^d and γ_n^d at a selection of valencies and with d ranging from 3 to 9 70
5.3 α_n^d at a selection of valencies and with d ranging from 3 to 9 72
5.4 Values for δ_d to achieve bounded curvature for valency three 77

6.1 Surfaces with three different knot configurations at degrees 3 to 9 84
6.2 Natural configurations for both odd- and even-degree bounded-curvature schemes . 90

Introduction 1

This dissertation is concerned with the design of *freeform surfaces*. These surfaces are typically smooth, but also have a shape which can be controlled by a designer. They are therefore more flexible than surfaces such as the natural quadrics (spheres, planes, circular cylinders and cones), but can also be smoother than surfaces which are formed as a composition of smaller primitives. Freeform surfaces have become increasingly important over the last fifty years, and are now used in fields as diverse as automotive and aeronautical engineering, ship-building, consumer product design, animated films and special effects, and even architecture and sculpture (see Figure 1.1).

One of the key challenges for Computer-Aided Design (CAD) is to provide ways of designing and representing freeform surfaces. In other words, to take some instructions from a designer that specify a shape, and then generate a smooth surface that meets the specification. The designer can

© Todd Randall Jordan (see page 6 for details)

Figure 1.1: The 'Cloud Gate' sculpture in Millennium Park, Chicago. This is a freeform surface which has been physically realised in highly-reflective metal, highlighting the smoothness of the surface.

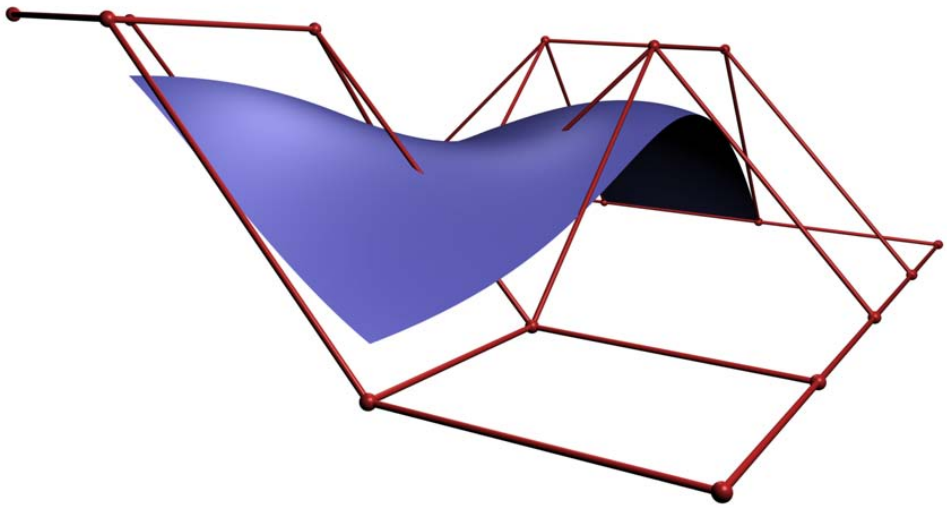


Figure 1.2: Example NURBS surface. NURBS and subdivision surfaces are both designed using an interface where the smooth surface (shown here in blue) approximates a given control mesh (in red).

specify a shape in several ways, but one that has become popular is using *control points* connected together as a *mesh* (see Figure 1.2). The designed surface approximates the given control mesh, whilst guaranteeing a certain level of smoothness. In some cases the required level of smoothness can also be part of the surface specification. For freeform surfaces which are designed in this way, there are two main competing representations: *Non-Uniform Rational B-Splines* (NURBS) and *subdivision surfaces*.

NURBS and subdivision surfaces 1.1

Both of these representations were first described in the late 1970s, and are now mature technologies which are used extensively in current applications. However, they were developed to offer a designer different freedoms. In short, NURBS offer control over the surface parametrisation and smoothness, while subdivision surfaces give freedom from topological constraints (see Chapter 2 for details). This has led different industries to establish a preference for one of the two representations; NURBS are the dominant standard for CAD and engineering applications, while subdivision surfaces are very popular for use in films and computer games.

Subdivision surfaces did not become widely used until the 1990s, by which time NURBS were already established as the industry standard for CAD. For purposes such as character animation [20], however, subdivision surfaces provide significant gains over a NURBS representation, as they remove the need for time-consuming 'stitching' of separate surface patches. It is therefore natural to ask whether CAD applications could also benefit from using subdivision surfaces. In fields such as product design, the need for the flexibility of subdivision has only grown more urgent in recent years, as products have taken on smoother and more freeform shapes.

Unfortunately, subdivision surfaces present an unwelcome compatibility hurdle for CAD vendors. In contrast to the younger entertainment industry, CAD packages must provide access to a huge volume of existing and historical data, almost all of which uses NURBS to represent freeform surfaces. Until the work described in this dissertation, however, subdivision surfaces

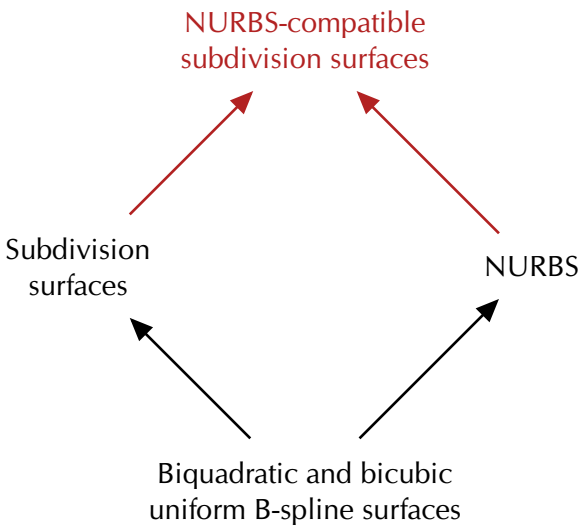


Figure 1.3: The incompatibility between NURBS and existing subdivision surfaces. In this figure arrows represent subset relations, showing that NURBS and the most popular subdivision schemes extend uniform biquadratic and bicubic B-splines in different ways. This dissertation describes NURBS-compatible subdivision surfaces, the first superset of NURBS without topological constraints.

have always been incompatible with NURBS, by which I mean that they are unable to represent a general NURBS surface without approximation. We can visualise this incompatibility by considering the class of surfaces which *are* contained in both representations (see Figure 1.3): existing subdivision surfaces are a superset (providing added flexibility) of only a subset of NURBS (in general the uniform, low-degree subset).

An arbitrary topology superset of NURBS 1.2

This dissertation shows that the compatibility hurdle can be overcome, by demonstrating a way to construct NURBS-compatible subdivision surfaces. This is a class of surfaces that contains NURBS as a proper subset, and can therefore represent any existing NURBS surface exactly. However, the new surfaces are also free of topological constraints, which is the key flexibility enabled by a subdivision surface representation. In discussing the potential for subdivision surfaces within CAD, Ma [48] anticipates exactly this development:

> "Ultimately a further generalization like NURBS for the CAD and graphics community and further standardization are expected. Such a unified generalization should cover all what [sic] we can do with NURBS, including the exact definition of regular shapes such as sphere, cylinder, cone, and various general conical shapes and rotational geometry."

There are two main respects in which subdivision surfaces have failed to represent the full generality of NURBS:

- they are uniform, rather than the more general non-uniform, and
- they are not defined for arbitrary degree.

These therefore become the two main challenges for us to overcome in constructing NURBS-compatible subdivision surfaces. In addition to Ma, Gonsor and Neamtu [32] explore the usefulness of subdivision to CAD, and their conclusions include a call for subdivision surfaces that allow non-uniform parametrisations. The work described here is therefore rooted firmly in the demands of industry.

Thesis 1.3

The fine detail of a standardised subdivision representation, such as the one that Ma expects, may not match the schemes in this dissertation. Indeed, there are likely to be several places where this new representation requires further analysis and research. However, I will demonstrate that

- it is possible to create subdivision schemes that generalise NURBS, thereby extending NURBS to arbitrary topologies, and that
- the generalisation can produce surfaces with a similar quality to the state of the art in subdivision representations.

Apart from presenting a solution to current problems in industry, I believe the generalisation is also interesting from a theoretical point of view. NURBS and subdivision surfaces have existed for over thirty years and have their foundations in exactly the same B-spline theory, but this work is the first time we are able to combine, comprehensively, the benefits of the two. In addition to this unification I also make the following contributions:

- a new factorisation of non-uniform B-spline knot insertion rules (§3.3): the first such factorisation to allow selective knot insertion,
- a new understanding of the link between the subdominant eigenvalue of a bounded-curvature subdivision scheme and the influence that an extraordinary vertex exerts on the limit surface (§5.5),
- a hypothesis proposing a connection between the stability of a subdivision scheme's natural configuration, and the quality of surfaces produced by that scheme (§5.3),
- the first arbitrary-degree subdivision schemes with bounded curvature (Chapter 5),
- a strategy for inserting knots into non-uniform subdivision surfaces (§4.3) which, when combined with the previous result, gives the first non-uniform subdivision schemes with bounded curvature.

These contributions draw heavily on previous work, in particular where subdivision schemes have made a step towards NURBS-compatibility by tackling each of the main challenges from §1.2 separately. In the next chapter I describe the B-spline background, and the previous work, in greater detail.

Background 2

This chapter describes the invention of NURBS and subdivision surfaces, starting from the common foundation of *B-splines*. Despite the importance of B-splines, the development of these surface representations is still only a small part of the history of CAD. For a broader view of this history, see Farin's summary [26].

B-splines 2.1

The word 'spline' originally referred to a thin strip of wood which was controlled using heavy spline weights (see Figure 2.1). To specify a particular curve, a designer used the weights to force points along the spline to pass through certain positions. The resistance of the spline to this deformation produces a smooth curve: the shape with minimal bending energy given the constraints set by the weights. This curve could then be traced onto a design lying underneath the whole arrangement.

Even before the computerisation of this type of design work, there were many applications for mathematical methods to smoothly interpolate data, just as the traditional spline interpolates the positions set by the spline weights. Schumaker [86] provides several references to early work in

© Edson International (see page 6 for details)

Figure 2.1: A traditional spline used for drawing smooth curves. The wooden spline is bent into shape using spline weights.

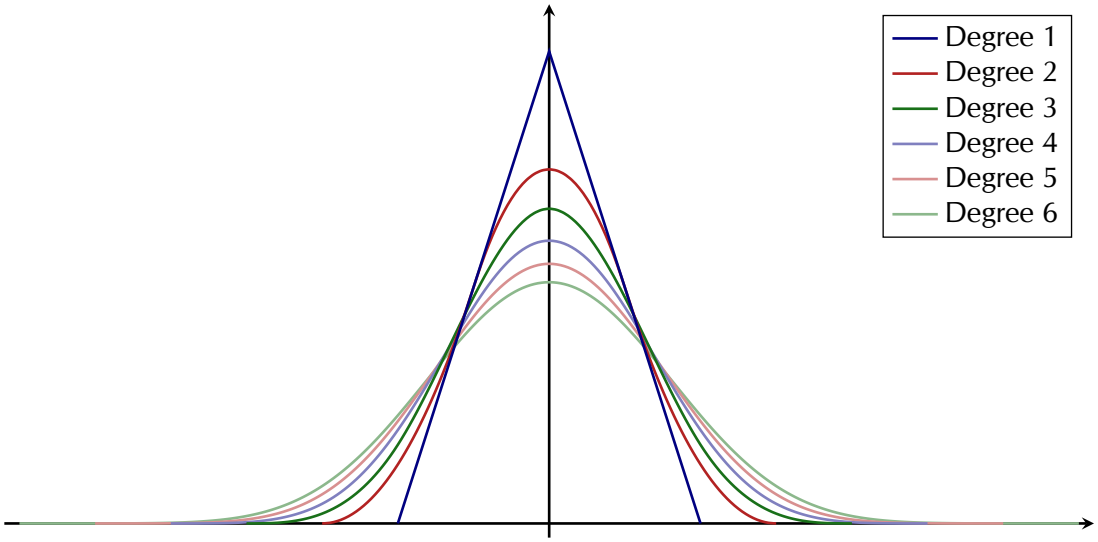


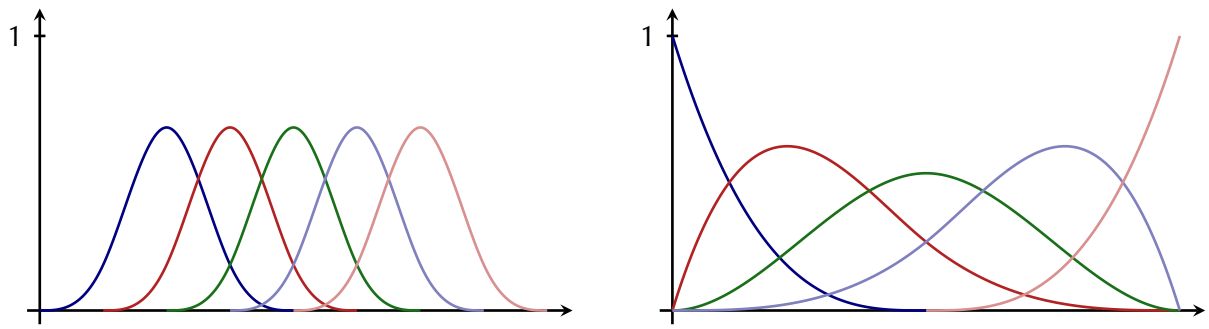
Figure 2.2: Uniform B-spline basis functions from degree 1 to 6. As degree increases, the basis functions become smoother and have a larger support, but each basis function still has only local influence over a curve or surface.

this field. However, Schoenberg [85] was the first to explicitly draw the analogy with a wooden spline when he coined the name 'B-spline' to stand for *basis spline*. Schoenberg used this name to refer to a class of basis functions (see Figure 2.2 for some examples), which can be linearly combined to form a smooth *spline function*. In fact within CAD, 'B-spline' has come to refer to an element in the span of this basis, which means we have to use the redundant name 'B-spline basis' for the basis itself.

B-splines are *piecewise polynomial*[1]: they consist of separate sections of polynomial joined together at positions called *knots*. The joins are constructed to be as smooth as possible: degree d B-splines have $d-1$ continuous derivatives across each knot. Given these constraints, the basis functions are uniquely defined by the property of minimal support (see, e.g. de Boor [16]), which in practice means that they make it possible to construct a long, smooth curve while still allowing a designer to modify only a small region at a time. This property is known as *local control*. As computers started to be used for the design of curves and surfaces, the smoothness and support properties made B-splines the subject of considerable theoretical interest. Local control, for example, was not available from the Bézier-Bernstein basis which was also popular at the time. However, B-splines were initially difficult to use in practical applications because the available evaluation algorithms were numerically unstable. Cox [13] and de Boor [14] independently addressed this shortcoming by providing a stable evaluation algorithm which bears their names, and Riesenfeld [72] recognised that this allowed B-splines to become a powerful method for representing freeform shapes.

Within CAD, B-splines are used to construct *parametric*[1] curves and surfaces, which is far more flexible than using functions defined directly on an axis line or plane. We therefore obtain a shape as a function $\phi : \mathbb{R}^p \to \mathbb{R}^n$ where p is equal to 1 for curves and 2 for surfaces. Each basis function is associated with a coefficient in $\mathbb{R}^n$, and the resulting position in space is known as a

[1]Ramshaw [67] points out that each of the adjectives *parametric*, *piecewise* and *polynomial* represents a design decision which can be separately motivated.



(a) The uniform case. Here each basis function is an identical translated copy of all the others.

(b) An example non-uniform case. In this example every basis function is unique.

Figure 2.3: Cubic B-spline basis functions for two different knot vectors. Figure 2.4 shows example B-spline curves defined on these bases.

(a) The uniform case.

(b) An example non-uniform case.

Figure 2.4: Cubic B-splines using the same control points, but two different knot vectors. Each B-spline consists of two segments of cubic polynomial. The basis functions associated with the control points are shown in Figure 2.3.

control point. This setup gives a mathematical framework for designing smooth shapes where, in the curve case, the control points act as a close analogue of the traditional spline weights. The main difference is that, in general, the resulting shapes *approximate* their control points rather than *interpolate* them. This can be motivated by the fact that B-splines hold desirable mathematical properties, such as possessing non-negative basis functions, which are not available to interpolating splines.

Knot vectors: uniform and non-uniform 2.1.1

The collection of knots for a B-spline is known as its knot vector. To continue the analogy with a wooden spline, this corresponds to the list of positions for spline weights when measured along the spline. A *non-uniform* B-spline can have these knots almost arbitrarily positioned, whereas for a *uniform* B-spline, they must be equally spaced. All knot vectors with equal spacing are shifts and scales of each other. The effect of shifting or scaling every knot is to transform the parameter space which is used as the domain of ϕ, but has no effect on its image, and describing a B-spline as uniform is therefore sufficient to characterise its knot vector completely. Figure 2.3 shows example uniform and non-uniform B-spline bases.

Non-uniform knot vectors are useful for several practical applications. In Figure 2.4(b), for example, non-uniform knots have been used to make a B-spline interpolate its control polygon's end points. This setup is popular as it allows a designer to accurately position a B-spline's boundary, but the required knot vector uses multiple (i.e. coincident) knots, which is only possible

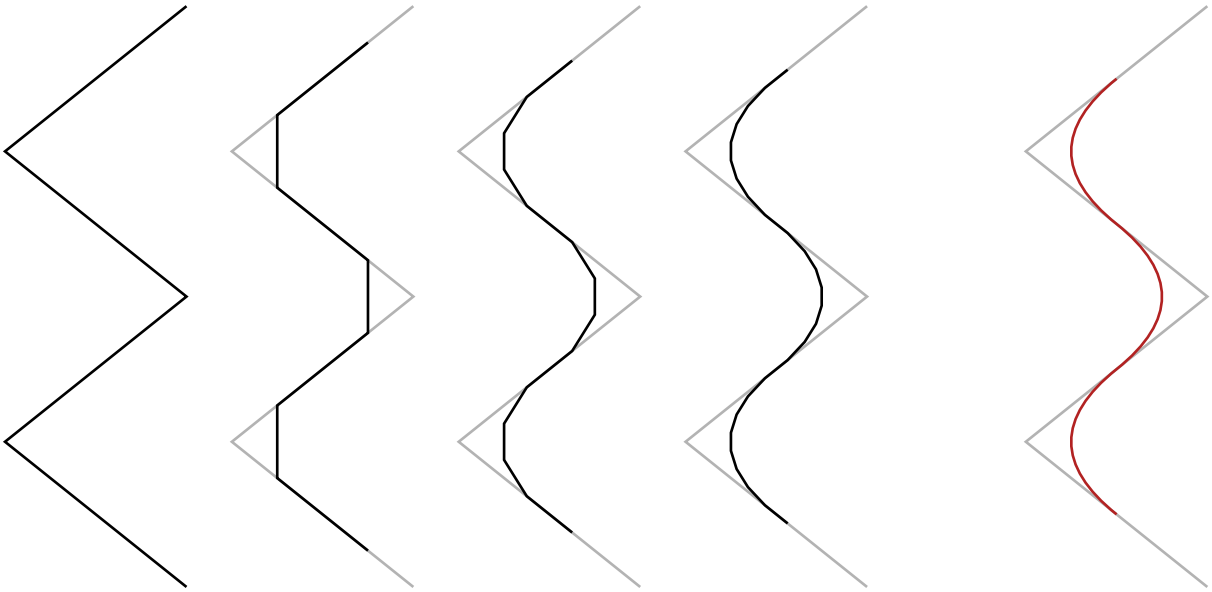


Figure 2.5: Chaikin's algorithm to construct a uniform quadratic B-spline. Each step of Chaikin's algorithm 'cuts the corners' of the control polygon. This figure shows three steps of Chaikin's algorithm followed by the limit curve: a uniform quadratic B-spline.

with a non-uniform knot vector. Multiple knots can also be used in the interior of a curve or surface, where the result is that continuity is reduced (by a specified amount) at a particular knot. Non-uniform knot vectors also allow a parametrisation to be more evenly distributed over freeform geometry, by allocating larger amounts of parameter space to longer sections of a curve or surface. This can be important when fitting B-splines to measured values which are not evenly spaced [58]. In a similar way, where short-wavelength features are created as part of a long-wavelength shape, the representation is more efficient and more easily modifiable if the control mesh is denser in areas of greater curvature variation. In this case, higher quality results are obtained by using a non-uniform knot vector that respects the control mesh spacing [32].

NURBS 2.1.2

By the 1970s, CAD was an important part of the design process used in industry, but there was no standardised representation for freeform curves and surfaces. This made it difficult to transfer designs between different systems. Boeing, for example, used two systems with completely incompatible representations: one based on B-splines and the other based on conic sections [26]. This could make it frustratingly difficult for different parts of a design team to work together.

Versprille [97] noticed that *rational* B-splines provided a way of unifying the majority of these different representations, as rational polynomials are able to reproduce conic sections without approximation. His ideas were developed by Piegl and Tiller [95] and the unification became known as NURBS: 'Non-Uniform Rational B-Splines'. NURBS were therefore created for the same reason as their generalisation described in this dissertation: to bring a wide range of freeform surfaces together in a standard representation. This vision was fulfilled, as IGES (the 'Initial Graphics Exchange Specification') made NURBS the lingua franca for freeform curves and surfaces in 1983 [57].

Uniform B-spline subdivision 2.1.3

Just as B-splines were starting to find their place within CAD, Chaikin [11] found a fresh way of generating and analysing them. He showed that a tangent-continuous freeform curve could be generated from a control polygon by a recursive 'corner cutting' procedure (Figure 2.5) which, unknown to Chaikin, de Rham [19] had investigated nearly thirty years earlier. Riesenfeld [73] and Forrest realised that this process generated a uniform quadratic B-spline, which was already well understood. However, Chaikin's fresh perspective was that these curves, which held useful properties, could be generated without any reference to the closed-form representation at all.

In fact the subsequent analysis showed that Chaikin's algorithm was generating the familiar control polygons for a given B-spline curve, but on a denser set of knots at each iteration. It became clear that inserting knots into a B-spline knot vector resulted in simple geometric rules that found a new control polygon from the existing one, and that it was possible to use the limit of this sequence of polygons as a way of defining the curve itself. This was immediately useful for practical applications, and is the idea that launched subdivision surfaces several years later (§2.2).

Knot insertion 2.1.4

Once Chaikin had shown that *uniform* B-splines could be subdivided, increasing the density of knots in the knot vector, there was a natural gap for the same result on *non-uniform* B-splines. This problem became known as 'knot insertion', but could equally be called 'non-uniform B-spline subdivision', as it is completely analogous to the uniform case. Furthermore, these non-uniform knot insertion rules are the foundation of non-uniform subdivision surfaces (§2.3), just as the uniform knot insertion rules (§2.1.3) are the starting point for the creation of uniform subdivision surfaces (§2.2).

Given a control polygon, a knot vector and a refined knot vector, a knot insertion algorithm returns the new control polygon that defines exactly the same B-spline but on the denser knot vector. This problem was tackled several times in different ways: Sablonniere [82] showed how to compute the required change of basis in specific scenarios, and Boehm's algorithm [8] allows one knot to be inserted at a time using a minimal amount of computation [9]. Cohen et al. independently developed the Oslo algorithm [12], which considers the general case where an arbitrary number of knots are inserted at once.

Uniform subdivision surfaces 2.2

The previous section described how B-splines, with associated tools like knot insertion, came to be the dominant standard for freeform curves and surfaces. The main remaining sticking point was that B-spline surfaces are topologically limited. The generalisation from B-spline curves to surfaces uses a tensor-product construction, resulting in a rectangular parameter space and a surface which is topologically equivalent to a plane (e.g. Figure 1.2). Closing (i.e. looping) the parameter space makes it possible to create a surface that is topologically an open cylinder or a torus, but a single B-spline patch cannot represent a surface of any higher genus.

This nullifies one of the main advantages of B-splines: the ability to handle large surfaces that would require separate pieces using the earlier technologies of the conic section or Bézier basis. The limitation on B-spline topology means that multiple patches are needed to represent surfaces

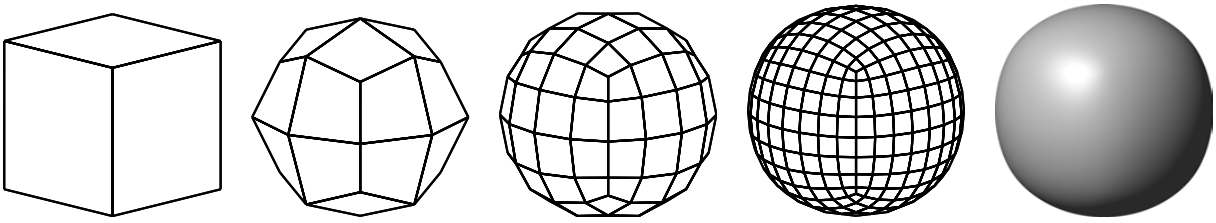


Figure 2.6: A variant of Catmull-Clark subdivision acting on a cube. This figure shows the control mesh after the first three subdivision steps, and the smooth limit surface.

of the complexity which is required in practice. Where separate B-spline surfaces meet, they must be manually 'stitched' together, and the seams do not have the same continuity guarantees which B-splines provide for the rest of the surface. In the specific cases where the surfaces have compatible knot vectors, these guarantees can instead be achieved by careful positioning of the control points on either side of each seam. However, numerical inaccuracy means that the composite surfaces are often not even continuous at the seams, let alone smooth.

The popularity of subdivision surfaces stems from their ability to remove these topological limitations. This section gives an overview of the key developments in this field; for a more comprehensive history see Sabin [76].

Generalising knot insertion 2.2.1

Subdivision surfaces work by viewing knot insertion as the primary definition of a surface, which is exactly the idea behind Chaikin's algorithm (§2.1.3). This means that the surface is defined by an iterative process; a control mesh is replaced with a denser mesh by calculating new points as a function of the old points[2]. This denser mesh is subdivided in turn, and the infinite sequence of meshes converges to a smooth *limit surface* (see Figure 2.6): the subdivision surface defined by the original control mesh.

It might seem at first that we have not gained anything by using this definition. If the function that calculates new points from old ones uses the subdivision rules derived from B-spline knot insertion, for example, then we already know an analytic form of the limit surface; at each step, knot insertion gives the control mesh for exactly the same B-spline surface as we started with, just on a denser knot vector. However, the freedom we have introduced is that we can allow the subdivision rules to vary where the connectivity of the control mesh differs from the regular case. Allowing irregular connectivity is the key to creating surfaces of arbitrary topology.

We can therefore understand subdivision schemes in terms of two properties:

- the rules that are used to insert points where the mesh is regular (which are often derived from knot insertion),
- how those rules are generalised to allow for meshes with irregular connectivity.

The first of these is the most important for understanding the broad behaviour of a scheme, and the rest of this section introduces subdivision schemes for important types of regular refinement. We shall consider the second property more carefully in Chapters 4 and 5.

[2]For the majority of subdivision schemes this function is linear in the control points, but nonlinear subdivision is also an active area of research. See, for example, Sabin and Dodgson [80], Wallner and Dyn [99], Schaefer et al. [84] and Dyn et al. [23]. In this dissertation, however, I focus exclusively on the linear setting.

© Blender Foundation (see page 6 for details)

Figure 2.7: Animated characters created using Catmull-Clark subdivision surfaces.

Schemes based on B-splines 2.2.2

The first subdivision surfaces generalised knot insertion rules for a simple but important class of B-splines: those defined on uniform knot vectors, for low degrees. Doo [22] was the first to consider modifications to Chaikin's algorithm to create uniform biquadratic subdivision surfaces, and Catmull and Clark [10] considered similar modifications to knot insertion rules for uniform bicubic B-splines (Figure 2.6). At the same time, Doo and Sabin [21] described how subdivision surfaces can be analysed using diagonalisation, so these early subdivision schemes became known as 'Catmull-Clark' and 'Doo-Sabin'.

Subdivision surfaces remained largely an academic curiosity for nearly two decades, but the late 1990s saw a surge of interest because the size of computer memories and the speed of processors made it tractable to compute the subdivided control meshes directly[3]. Subdivision surfaces were also used in a successful high-profile experiment at Pixar by De Rose et al. [20], which resulted in Pixar converting all their modelling and animation tools to use a subdivision surface representation [106]. Other animation companies soon followed, and Catmull-Clark subdivision surfaces are now the standard representation for animated characters used in films (Figure 2.7).

Schemes not based on B-splines 2.2.3

The Doo-Sabin and Catmull-Clark subdivision schemes are based on tensor-product B-splines and therefore operate on quadrilateral meshes. They refine a control mesh as shown in Figures 2.8(a)

[3] In fact as a result of the increased interest in subdivision, Stam [90] showed that subdivision surfaces can be evaluated efficiently (without an exponential cost), so this explanation now appears spurious. However, computational considerations were widely believed to count against subdivision surfaces before Stam's insight.

and (b) respectively. Nearly ten years after these first schemes were introduced, Loop [46] filled a natural gap for a subdivision scheme which instead uses triangular control meshes. The refinement pattern for Loop's scheme is shown in Figure 2.8(e), and Dyn et al. [25] used the same type of refinement to create the first surface subdivision scheme where the limit surface interpolates a given control mesh rather than approximating it.

Catmull-Clark and Loop continue to be the most important subdivision schemes for use in practice; both create surfaces that are C^2 continuous apart from isolated singularities which arise from irregularities in the control mesh , where both schemes are C^1. It might appear that these early schemes cover the feasible space of regular grid refinement, but researchers have also found interesting subdivision schemes with refinement patterns that incorporate a rotation of the grid directions. In particular, this includes the Simplest scheme by Peters and Reif [53], the $\sqrt{3}$ scheme by Kobbelt [41] and the 4-8 scheme by Velho and Zorin [96].

The schemes shown in Figures 2.8(c) to (f) are based on a variety of surface types: Loop, Simplest and 4-8 all generalise knot insertion rules for box splines, a class of smooth functions which is closely related to B-splines[4]. The Butterfly scheme generalises an important interpolatory subdivision scheme for curves [24], and the $\sqrt{3}$ scheme creates a novel surface with C^2 continuity in regular regions but a fractal support for each basis function [35]. However, none of these surfaces can be represented by NURBS, so no modifications to these schemes would be able to create NURBS-compatible subdivision surfaces. In §2.2.4 and the rest of this dissertation, we return to subdivision schemes that generalise tensor-product B-splines.

Refine and smooth 2.2.4

At the same time as the first research on knot insertion (§2.1.4), which produced the Oslo algorithm [12] and Boehm's algorithm [8] for inserting knots into non-uniform B-splines, Lane and Riesenfeld [42] showed that in a specific uniform case, knot insertion is susceptible to an important and useful factorisation. Consider Chaikin's algorithm, which is a special case of the Lane-Riesenfeld algorithm, and which we can write as

$$P_{2i} = \frac{3}{4}Q_i + \frac{1}{4}Q_{i+1} \quad , \quad P_{2i+1} = \frac{1}{4}Q_i + \frac{3}{4}Q_{i+1} \quad \text{for } i \in \mathbb{Z}$$

Here a new set of points P_i is constructed from the old set Q_i. Lane and Riesenfeld showed that this calculation can be broken into two stages:

$$P^1_{2i} = Q_i \quad , \quad P^1_{2i+1} = \frac{1}{2}Q_i + \frac{1}{2}Q_{i+1} \tag{2.1}$$

followed by

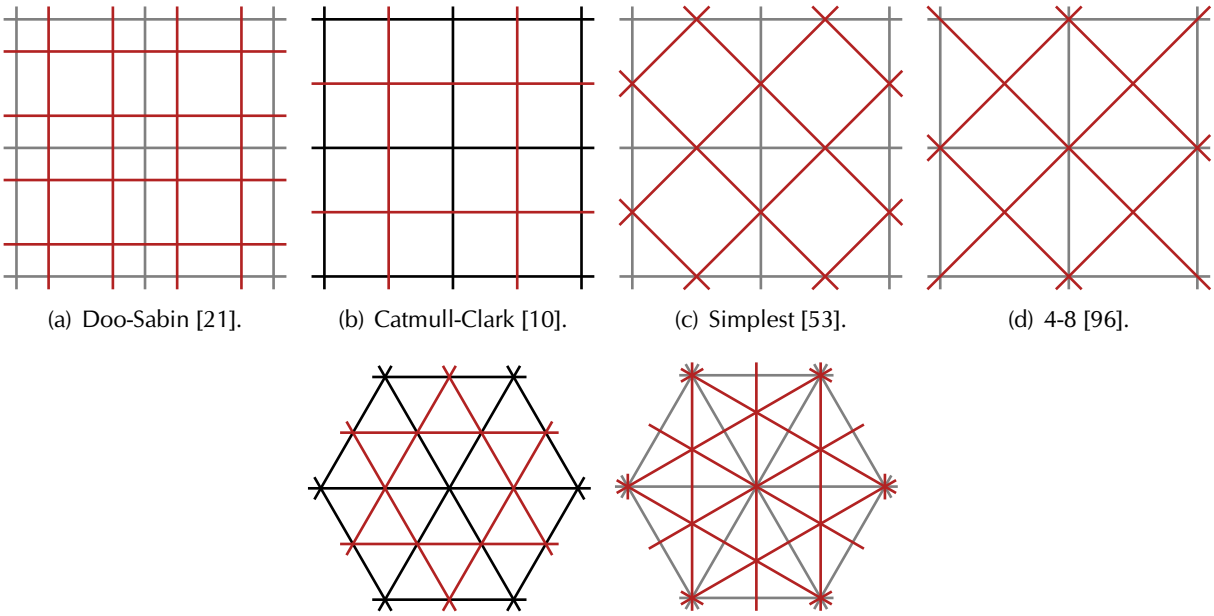
$$P^2_i = \frac{1}{2}P^1_i + \frac{1}{2}P^1_{i+1}$$

In fact by replacing this last expression with the recurrence

$$P^{d+1}_i = \frac{1}{2}P^d_i + \frac{1}{2}P^d_{i+1} \tag{2.2}$$

the Lane-Riesenfeld algorithm can compute a uniform subdivision step, that inserts a new knot into the centre of every existing knot interval, for B-splines of any degree d. Taubin [94] called

[4] The '4-3' scheme by Peters and Shiue [55] is also based on box splines.



(a) Doo-Sabin [21]. (b) Catmull-Clark [10]. (c) Simplest [53]. (d) 4-8 [96].

(e) Loop [46], Butterfly [25]. (f) $\sqrt{3}$ [41].

Figure 2.8: Regular refinement patterns for important subdivision schemes. Original edges are drawn in black where they form part of the refined pattern, or in grey where they do not.

these *refinement* and *smoothing* stages[5] and Vouga and Goldman [98], among others, note that the refinement stage (2.1) can be replaced by

$$P^0_{2i} = Q_i \quad , \quad P^0_{2i+1} = Q_i$$

since (2.1) is then given by one application of the smoothing stage (2.2).

The Lane-Riesenfeld algorithm is a very efficient way of inserting knots in this uniform, global case; for tensor-product surfaces, unfactorised knot insertion has complexity quadratic in degree, whereas computation required for the Lane-Riesenfeld algorithm grows only linearly. However, for subdivision surfaces that allow irregular control meshes, the fact that the Lane-Riesenfeld algorithm keeps all computation *local* also confers an important benefit. The generalisation to irregular connectivity discussed in §2.2.1 requires that we consider all possible mesh configurations, and without the Lane-Riesenfeld algorithm the number of cases to consider grows exponentially with degree. By using repeated applications of the fixed-width affine combination given in (2.2), however, the number of special cases is kept constant.

Several researchers have created arbitrary-degree subdivision surfaces by generalising the Lane-Riesenfeld algorithm and making use of this property. Prautzsch [62] and Warren and Weimer [103] described the natural 'midpoint' generalisation, and Zorin and Schröder [105] showed that the resulting subdivision surfaces are C^1 at singularities for degrees ≤ 9. Stam [91] and Stewart and Foisy [93] addressed some practical considerations by describing variants where the topology of the mesh is invariant under smoothing, and Prautzsch and Chen [63] proved C^1

[5] Taubin [94] actually uses the terms refinement and smoothing *steps*, while other researchers (e.g. Bajaj et al. [4]) have called the same operations 'splitting' and 'averaging' rules. I follow Taubin's terminology, except for calling the factorised components *stages* to distinguish them from the subdivision *step* that they are part of.

continuity at all degrees ≥ 2. In regular regions, all of these schemes generate tensor-product B-splines of any specified degree d, and are therefore C^{d-1}.

Non-uniform subdivision surfaces 2.3

By supporting arbitrary-topology surfaces, subdivision schemes remove the need for the error-prone 'stitching' of B-spline patches discussed in §2.2. However, the schemes that I have discussed so far can only do so in the limited context of a *uniform* parametrisation. These schemes therefore have none of the benefits of *non-uniform* knot vectors described in §2.1.1. NURBS, on the other hand, do support non-uniform parametrisations, and so any method purporting to remove the need for NURBS stitching must also be able to represent non-uniform B-splines.

Some researchers have analysed non-uniform subdivision schemes which are capable of generating not only the B-splines, but also a wide class of other curves. De Boor [15], for example, proved convergence for an arbitrary 'corner cutting' procedure, and Gregory and Qu [33] gave conditions that produce smooth (C^1) curves from the same general framework. Warren [102] pushed this even further, to consider C^k continuity for arbitrary k. However, we do not need this level of generality for compatibility with NURBS, and knot insertion for non-uniform B-splines is already well understood (§2.1.4). The difficulty lies in generalising these knot insertion rules to surfaces defined on irregular meshes: exactly the question raised in §2.2.1.

NURSS 2.3.1

Sederberg et al. [89] were the first to describe non-uniform subdivision surfaces that tackle this problem, using schemes they named 'NURSS': *Non-Uniform Recursive Subdivision Surfaces*. They gave knot insertion rules that specialise to the Doo-Sabin and Catmull-Clark rules in the uniform case, but can also represent non-uniform biquadratic or bicubic B-splines exactly. Qin and Wang [66] pointed out that the resulting subdivision surfaces have some shortcomings around vertices with high valency[6], at least for the biquadratic case, but Wang et al. [100, 101] attempted to provide efficient evaluation methods for the NURSS schemes nevertheless. Müller et al. [51] also addressed evaluation of a non-uniform Catmull-Clark scheme by using a different generalisation, which makes it possible to evaluate the limit surface of their scheme at any given vertex.

Sederberg et al. [88] developed the NURSS construction even further by allowing a control mesh to contain T-junctions. They called the resulting surfaces *T-splines*. Like the work in this dissertation and the earlier NURSS, T-splines maintain backwards-compatibility with NURBS [87], but T-junctions are not necessary for that compatibility as they are not supported by NURBS. In fact the combination with T-splines required a restriction on the NURSS knot vectors (see §4.1 for details), but compatibility with bicubic NURBS was, again, unaffected by this restriction.

Specialised applications 2.3.2

Karčiauskas and Peters [39] also created a non-uniform scheme for subdivision surfaces, but for a very specific purpose: modifying the speed at which the subdivision process converges towards an extraordinary vertex. Their scheme is otherwise identical to Catmull-Clark, in the regular regions of the surface, and is therefore able to represent only *uniform* B-splines.

[6] The valency of a vertex is the number of edges connected to it.

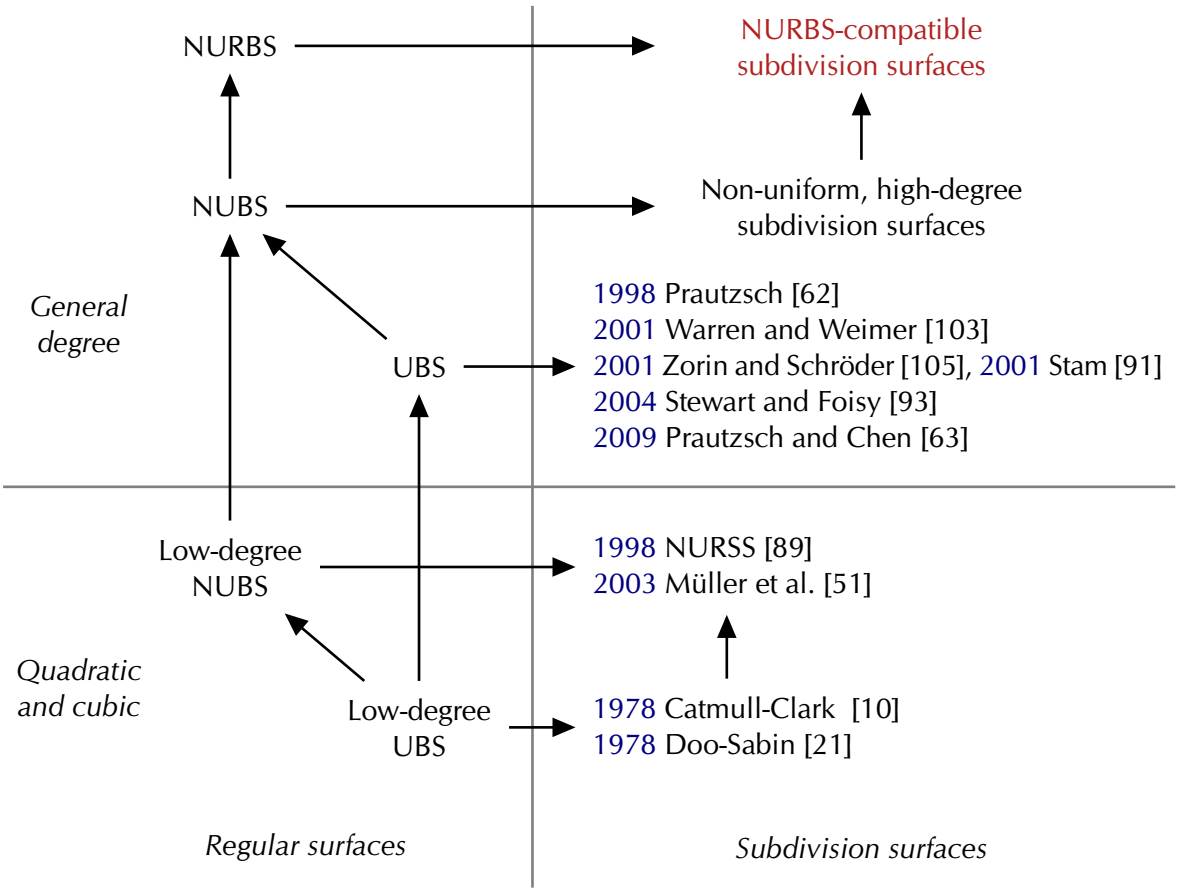


Figure 2.9: An overview of subdivision surfaces based on B-splines. This diagram shows classes of surfaces with subset relations between them (where ⟶ represents ⊂). The acronym NURBS is shortened to *NUBS*: 'Non-Uniform B-Splines' and *UBS*: 'Uniform B-Splines'. No previous subdivision scheme provides an arbitrary-topology superset of NURBS.

Summary 2.4

Figure 2.9 summarises the relationships between the main subdivision schemes that generalise B-spline surfaces. The diagram shows that all previous subdivision schemes have provided a superset (allowing arbitrary topology) of only a subset of NURBS (a subset that is restricted to be uniform, or low degree, or both). This dissertation, by contrast, introduces *NURBS-compatible subdivision surfaces*: the first arbitrary-topology superset of NURBS. The two main barriers to achieving this are:

- non-uniform parametrisations, and
- providing a general-degree representation.

These challenges have been tackled separately in previous work, but a superset of NURBS must incorporate both at the same time. The other requirement for a superset of NURBS is a *rational* representation (see §2.1.2), but this is straightforward [89]: rational control points can be projected into $\mathbb{R}^4$, subdivided, and then projected back into $\mathbb{R}^3$. The following chapters therefore address the two main challenges, above, by describing a subdivision scheme that can handle them both simultaneously.

Non-uniform refine and smooth 3

This chapter presents research that has also been published in the following papers:

> T. J. Cashman, N. A. Dodgson and M. A. Sabin. Selective knot insertion for symmetric, non-uniform refine and smooth B-spline subdivision. *Computer Aided Geometric Design*, 26(4):472–479, 2009.
>
> T. J. Cashman, N. A. Dodgson and M. A. Sabin. A symmetric, non-uniform, refine and smooth subdivision algorithm for general degree B-splines. *Computer Aided Geometric Design*, 26(1):94–104, 2009.
>
> T. J. Cashman, N. A. Dodgson and M. A. Sabin. Non-uniform B-Spline Subdivision Using Refine and Smooth. In *12th IMA Conference on the Mathematics of Surfaces*, volume 4647 of *Lecture Notes in Computer Science*, pages 121–137. Springer, 2007.

The rules for subdivision surfaces are derived from knot insertion (§2.2.1), and NURBS-compatible subdivision surfaces must be both non-uniform and defined for arbitrary degree. The rules for NURBS-compatible schemes must therefore generalise non-uniform general-degree B-spline knot insertion. This is exactly the work described in §2.1.4: both Boehm [8] and the Oslo algorithm [12] give rules for B-spline knot insertion in this general setting.

However, recall from §2.2.4 that for high degrees, it is not feasible to use an unfactorised knot insertion algorithm directly, as there are too many mesh configurations to consider; the number of cases grows exponentially with degree. A refine-and-smooth factorisation keeps this number of cases constant, and is more efficient: for surfaces, the computation required for knot insertion rises quadratically with degree for unfactorised algorithms but linearly for refine and smooth. To create non-uniform general-degree subdivision schemes, a refine-and-smooth factorisation is therefore essential. Other subdivision schemes based on arbitrary-degree B-splines [62, 63, 91, 93, 103, 105] generalise the Lane-Riesenfeld algorithm [42], but this is restricted to *uniform* knot insertion. For a scheme that can represent all NURBS surfaces, and hence must handle *non-uniform* knot vectors, we need a similar refine-and-smooth factorisation but for non-uniform knot insertion.

Previous work has tackled this problem for specific non-uniform configurations. Goldman and Warren [30] modified the Lane-Riesenfeld algorithm for knots in geometric sequence (whereas a uniform knot vector has knots in arithmetic sequence), and Plonka [60] found a factorisation for knots at uniform positions but with a given multiplicity. However, we need an algorithm that is not restricted to any specific arrangement of knots. This is the problem I address in this chapter.

Limitations and requirements 3.1

Before attempting to find a non-uniform analogue of the Lane-Riesenfeld algorithm, we need to establish our requirements for the factorisation. This is important, as this section will show that a

natural set of expectations from the uniform case leads to an overconstrained problem, and we must therefore impose fewer properties on the non-uniform algorithm than in the uniform case. First I define the notation that we will need to describe these requirements precisely.

We have a B-spline of degree d that we wish to subdivide. The B-spline is defined by its knot vector, $\boldsymbol{t}$, and its control points. We must specify a new knot vector $\boldsymbol{u}$ for the subdivided version, and the knot insertion algorithm then determines the location of the new control points. Let the subdivision matrix of degree d that transforms B-splines on $\boldsymbol{t}$ into B-splines on $\boldsymbol{u}$ be $\boldsymbol{S}^d$. Knot insertion is simply a change of basis [47], and if $B_{n,d,\boldsymbol{\gamma}}(x)$ is the nth B-spline basis function of degree d on knot vector $\boldsymbol{\gamma}$, then $\boldsymbol{S}^d$ is the basis transformation matrix that gives the co-ordinates of each $B_{j,d,\boldsymbol{t}}(x)$ relative to the $B_{i,d,\boldsymbol{u}}(x)$:

$$B_{j,d,\boldsymbol{t}}(x) = \sum_i S^d_{ij} B_{i,d,\boldsymbol{u}}(x).$$

A knot insertion algorithm therefore calculates new control points as a weighted sum of existing control points. S^d_{ij} is the weight[7] that multiplies the jth control point in contribution to the ith new control point.

We are seeking to preserve some properties of the Lane-Riesenfeld algorithm from the uniform case, but will need to relax others to allow for non-uniform knot vectors. So consider the following list of properties, all of which apply to the Lane-Riesenfeld algorithm, and which we might consider retaining in a non-uniform analogue:

A. For any degree d, there are smoothing matrices $\boldsymbol{M}^0, \boldsymbol{M}^1, \ldots, \boldsymbol{M}^{d-1}$ such that

$$\boldsymbol{S}^d = \boldsymbol{M}^{d-1}\,\boldsymbol{M}^{d-2}\,\ldots\,\boldsymbol{M}^0\,\boldsymbol{S}^0.$$

B. Furthermore, for each $0 < \delta < d$, $\boldsymbol{M}^{\delta-1}\,\boldsymbol{M}^{\delta-2}\,\ldots\,\boldsymbol{M}^0\,\boldsymbol{S}^0 = \boldsymbol{S}^\delta$.

C. Each $\boldsymbol{M}$ is a band matrix of bandwidth two.

Although these three properties apply in the uniform case, there is an immediate problem in applying them to a non-uniform analogue, which is the phase shift in a knot vector when d changes from odd to even or vice versa. When d is odd (the primal case), control points are aligned with *knots*, but when d is even (the dual case), control points are aligned with *knot intervals*. So to be meaningful for non-uniform knot vectors, we must amend these properties to stay in either the primal or dual case respectively. We can do so by combining pairs of smoothing stages together:

A. For any degree d, there are smoothing matrices $\boldsymbol{N}^{d \bmod 2}, \boldsymbol{N}^{2+d \bmod 2}\,\ldots, \boldsymbol{N}^{d-2}$ such that

$$\boldsymbol{S}^d = \boldsymbol{N}^{d-2}\,\boldsymbol{N}^{d-4}\,\ldots\,\boldsymbol{N}^{d \bmod 2}\,\boldsymbol{S}^{d \bmod 2}.$$

B. Furthermore, for each $0 < \delta < d$ such that $d - \delta$ is even,

$$\boldsymbol{N}^{\delta-2}\,\boldsymbol{N}^{\delta-4}\,\ldots\,\boldsymbol{N}^{\delta \bmod 2}\,\boldsymbol{S}^{\delta \bmod 2} = \boldsymbol{S}^\delta.$$

C. Each $\boldsymbol{N}$ is a band matrix of bandwidth three.

[7] The original description of the Oslo algorithm [12] uses the notation $\alpha_{jk}(i)$ for S^{k-1}_{ij}.

Of these, properties A and C are important for the factorisation to be useful: A states that the algorithm must compute the same knot insertion result as the unfactorised algorithm, and C gives the local affine combinations that were the reason for seeking a factorisation in the first place. Property B is less crucial, but is nevertheless an elegant property of the Lane-Riesenfeld algorithm that we might hope to maintain in a non-uniform analogue. This property states that to compute knot insertion for quintic B-splines, for example, we first compute the linear case ($d = 1$), and then the cubic ($d = 3$), and finally the control points for the quintic case ($d = 5$).

To study the implications of property B, consider a very limited case: knot insertion on a uniform knot vector, but with one new knot positioned arbitrarily in each knot interval. Let r_i give the position of the new knot in the interval $[\, t_i \, t_{i+1} \,]$, i.e. $r_i = (u_{2i+1} - t_i)/(t_{i+1} - t_i)$ and let $\overline{r_i} = 1 - r_i$. Then the section of $\boldsymbol{S}^3$ which computes the three points centred around u_{2i+1} is

$$\tilde{\boldsymbol{s}}^3 = \frac{1}{6}\begin{pmatrix} (1+\overline{r_{i-1}})\overline{r_i} & 4+2\overline{r_{i-1}}r_i & r_{i-1}(1+r_i) & 0 \\ 0 & 2+2\overline{r_i} & 2+2r_i & 0 \\ 0 & (1+\overline{r_i})\overline{r_{i+1}} & 4+2\overline{r_i}r_{i+1} & r_i(1+r_{i+1}) \end{pmatrix}$$

Property B requires that $\boldsymbol{N}^3$ takes an affine combination of these three rows to form a single row of $\boldsymbol{S}^5$:

$$\tilde{\boldsymbol{s}}^5 = \frac{1}{60}\begin{pmatrix} (2+\overline{r_{i-1}}) \times & 3(\overline{r_{i-1}}\overline{r_i}r_{i+1} + \overline{r_i}r_{i+1} + & 3(\overline{r_{i-1}}r_ir_{i+1} + \overline{r_{i-1}}r_i + & (2+r_{i+1}) \times \\ (1+\overline{r_i})\overline{r_{i+1}} & 2\overline{r_{i-1}} + 2\overline{r_i} + 2\overline{r_{i+1}} + 6) & 2r_{i-1} + 2r_i + 2r_{i+1} + 6) & (1+r_i)r_{i-1} \end{pmatrix}$$

That is, we require $\boldsymbol{N}^3$ to have a 1×3 block $\tilde{\boldsymbol{n}}$ such that $\tilde{\boldsymbol{s}}^5 = \tilde{\boldsymbol{n}}\tilde{\boldsymbol{s}}^3$ and $\sum_i \tilde{n}_i = 1$.

In this case, $\tilde{\boldsymbol{n}}$ is fixed by the pattern of zeros in $\tilde{\boldsymbol{s}}^3$: $\tilde{n}_1$ must be $\tilde{s}^5_1/\tilde{s}^3_{11}$ and $\tilde{n}_3$ must be $\tilde{s}^5_4/\tilde{s}^3_{34}$. The remaining coefficient, $\tilde{n}_2$, is then determined by the requirement for $\tilde{\boldsymbol{n}}$ to be an affine combination. It is not guaranteed, however, that this value for $\tilde{\boldsymbol{n}}$ will indeed give $\tilde{\boldsymbol{s}}^5 = \tilde{\boldsymbol{n}}\tilde{\boldsymbol{s}}^3$. If we examine either the second or the third column (if the relationship holds for one then it must also for the other, since all rows sum to 1), then we find the following restriction on the r_i:

$$4r_i^3 - 3r_i^2(r_{i-1} + 1 + r_{i+1}) + 2r_i(2r_{i-1} + 1 + r_{i-1}r_{i+1}) - r_{i-1}r_{i+1} - 2r_{i-1} = 0 \qquad (3.1)$$

Therefore if a factorisation satisfies property B, then in this case setting the position of two new knots, which fixes r_{i-1} and r_i, means that r_{i+1} is already determined as the solution to (3.1). This constraint is shown in Figure 3.1, which shows that for some values of (r_{i-1}, r_i), r_{i+1} is not merely fixed, but has no solution at all such that $0 \le r_{i+1} \le 1$.

This simple example shows that we should not expect to be able to maintain property B in the non-uniform case. In Chapter 4, we will need the flexibility to be able to insert knots at any given position, and this example shows that property B is too tight a constraint for this level of freedom. The remainder of this chapter therefore considers non-uniform analogues of the Lane-Riesenfeld algorithm which hold only properties A and C of the three.

Polar form 3.2

A powerful tool for analysing B-splines is the *polar form* or 'blossom'. This form will prove instrumental in finding a refine-and-smooth factorisation of non-uniform B-spline knot insertion.

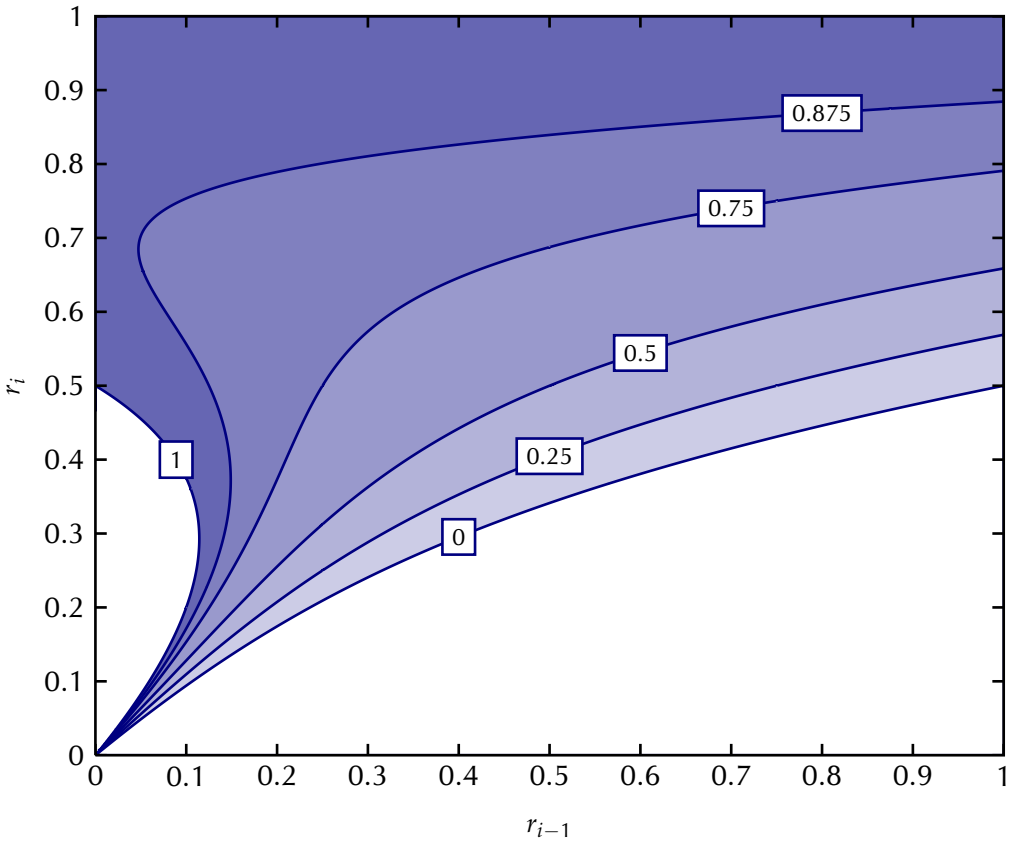


Figure 3.1: The constraints placed by property B on quintic knot insertion. This contour plot shows the value of r_{i+1}, as constrained by r_{i-1} and r_i, if we seek a non-uniform refine-and-smooth factorisation that preserves property B from page 32. The shaded region gives $0 \leq r_{i+1} \leq 1$ and therefore shows the possible values for (r_{i-1}, r_i) for a factorisation which is constrained in this way.

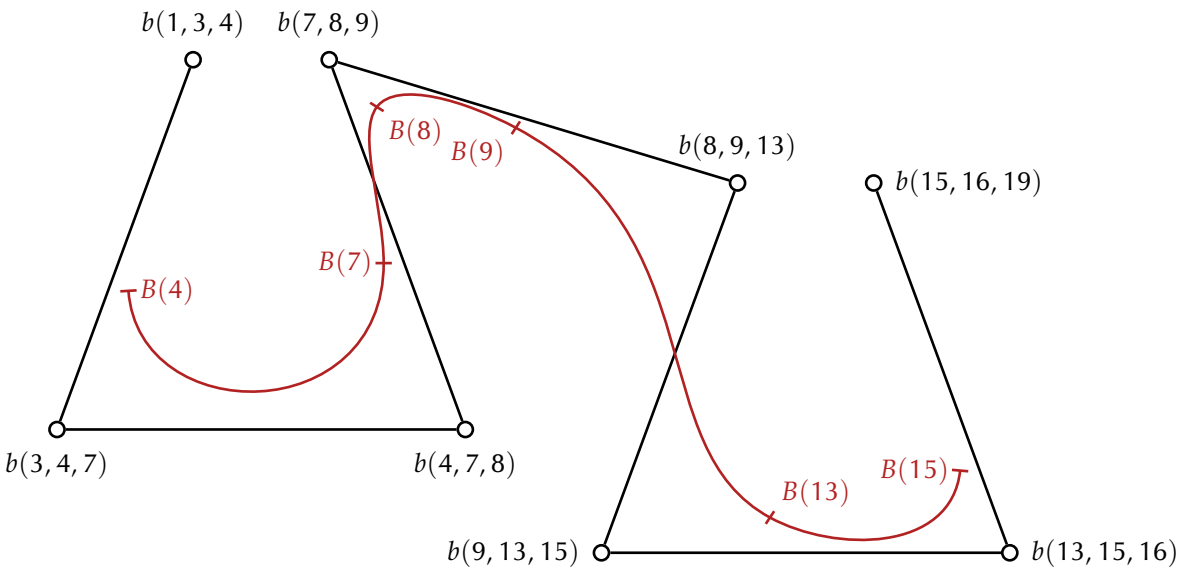


Figure 3.2: The polar form of a B-spline. This figure shows a cubic non-uniform B-spline, B, defined on the knot vector [1, 3, 4, 7, 8, 9, 13, 15, 16, 19]. The control points are given by the polar form, b, where the arguments (of which there are three, because B is cubic) are consecutive values from the knot vector.

The term blossom was introduced by Ramshaw [67], who described many implications of viewing a polynomial with this framework, although he later discovered [68] that both de Casteljau [18] and de Boor [17] had independently arrived at the same concept.

Vouga and Goldman [98] provide the following summary. The polar form of a degree-d polynomial $B(t)$ is the unique polynomial in d variables[8] $b(v_1, v_2, \ldots, v_d)$ which satisfies these three properties:

- symmetric:
  $b(v_{\sigma 1}, v_{\sigma 2}, \ldots, v_{\sigma d}) = b(v_1, v_2, \ldots, v_d)$ for any permutation, σ, of $\{1, 2, \ldots, d\}$.
- multiaffine:
  $b((1-\alpha)v_{11} + \alpha v_{12}, v_2, \ldots, v_d) = (1-\alpha)b(v_{11}, v_2, \ldots, v_d) + \alpha b(v_{12}, v_2, \ldots, v_d)$.
- diagonal:
  $b(t, t, \ldots, t) = B(t)$.

As a consequence of satisfying these three properties, the polar form also holds the property illustrated in Figure 3.2: if P_i is the ith control point of a B-spline $B(t)$ with knot vector $\boldsymbol{t}$, then $P_i = b(t_{i+1}, \ldots, t_{i+d})$. Vouga and Goldman [98] called this the *dual function* property, as it can be derived [43] using the dual functionals developed by de Boor and Fix [16].

Knot insertion 3.2.1

The polar form allows us to reduce B-spline knot insertion to a graph reachability problem. We have a set of starting nodes: the control points of a B-spline. By the dual function property, these points are given by the polar form evaluated on consecutive knots in the original knot vector $\boldsymbol{t}$. We also have a set of end nodes: the control points of the same B-spline, but on a subdivided knot vector. Again, using the dual function property, we know that these points are given by the polar form evaluated on consecutive knots in the *refined* knot vector $\boldsymbol{u}$. Using the multiaffine property we can combine any two nodes which share $d-1$ polar arguments to generate new nodes which evaluate the polar form at new positions. The symmetry property means that it does not matter *which* $d-1$ arguments are shared. The goal of this process is to provide a path, by taking multiaffine combinations, from the start to the end nodes.

Goldman [31] shows that many knot insertion algorithms (including Boehm's algorithm [8] and the Oslo algorithm [12]) fit into this common framework. The polar form is not the only way of deriving knot insertion rules, however: Vouga and Goldman [98] show that in general the Lane-Riesenfeld algorithm does *not* take a path through polar evaluations in the way described above. The polar form has, nevertheless, proved to be a useful and powerful way to analyse knot insertion, particularly in the non-uniform case.

Schaefer's algorithm 3.2.2

We are now ready to return to the problem posed at the start of this chapter: finding a refine-and-smooth factorisation of knot insertion rules for non-uniform general-degree B-splines. We know that we should not expect the factorisation to hold property B from §3.1, and the polar

[8] Ramshaw [68] introduced the term *polar arguments* for the d values at which a polar form is evaluated.

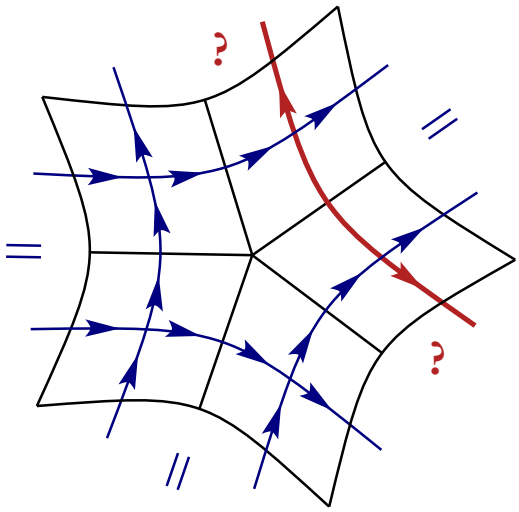


Figure 3.3: The need for a symmetric algorithm. An asymmetric algorithm requires a consistent orientation for mesh edges, but this is impossible to achieve for some vertex valencies. In this example, no orientation for the red edge is consistent with its parallel edges.

form provides us with a framework in which to search for a solution. In fact Schaefer and Goldman [83] recently described a knot insertion algorithm, derived using the polar form, which fits this specification exactly. However, Schaefer's algorithm is *asymmetric*: smoothing stages are dependent on the direction in which control points are indexed. The process of knot insertion is independent of this direction, and therefore so is the end result of this factorisation. Using Schaefer's approach, however, the results of intermediate smoothing stages are dependent on an arbitrary orientation.

To operate directly on a mesh, Schaefer's algorithm therefore requires each edge to have an orientation which is consistent with parallel edges. Figure 3.3 shows this is not possible to achieve for a mesh with arbitrary connectivity. In addition to the requirements we found in §3.1, we therefore need to add one more: the factorisation must be symmetric (independent of orientation), so that it is unaffected by this orientation problem.

Symmetric algorithm 3.3

The remainder of this chapter describes my solution to this set of requirements. The algorithm I developed has the additional benefit of allowing *selective* knot insertion, where knots are inserted in some knot intervals but not others. The earlier factorisations (Schaefer's algorithm [83] for non-uniform knot insertion, and Lane-Riesenfeld [42] for the uniform case) both require a new knot to be inserted in every existing knot interval. Where original knots are multiple, the subdivision process therefore increases multiplicity, which is undesirable. The factorisation described here avoids this problem by allowing a selection of knot intervals to remain unchanged in the subdivided B-spline.

The factorisation is described here for the univariate case, where a B-spline curve is specified using a control polygon. For a surface controlled by a mesh with *regular* connectivity, the bivariate case is simply the tensor product of the rules described here. In Chapter 4, I consider the generalisation to surfaces with *irregular* connectivity.

Problem statement 3.3.1

As before, let B be a B-spline of degree d with a knot vector $\boldsymbol{t}$. We want to calculate the control points for B on a new knot vector. To provide access to all the required knots in a single object, let $\boldsymbol{u} = (u_j), j \in Z$ be a non-decreasing sequence that contains both the old knot vector and the new[9]. $\boldsymbol{u}$ is indexed using Z, an uninterrupted interval on $\mathbb{Z}$. To provide the freedom to choose whether a new knot is inserted in any given interval, we need an explicit list of the original knots. Therefore let Y be the indexing set for $\boldsymbol{t}$, where $Y \subset Z$ and, since $\boldsymbol{t}$ is a subsequence of $\boldsymbol{u}$, $\boldsymbol{t} = (u_j), j \in Y$. For example, if we have five knots and wish to insert knots in every interval, then $Z = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ and $Y = \{1, 3, 5, 7, 9\}$. If, however, we did not want a new knot in the third of the four intervals, then $Z = \{1, 2, 3, 4, 5, 6, 7, 8\}$ and $Y = \{1, 3, 5, 6, 8\}$.

By allowing selective knot insertion, this formulation allows us to subdivide B with multiple knots in $\boldsymbol{t}$ without increasing knot multiplicity. There are, however, restrictions on Y in order to ensure that this works correctly, and there are end conditions to consider, discussion of which I defer to §3.3.7. The restrictions on Y are:

- Y has the same bounds as Z (i.e. Y and Z are either unbounded below, or share a common minimum, and likewise for the upper bound),
- at most one new knot is inserted between adjacent old knots (i.e. $\forall\, j \in Z \setminus Y$, $j + 1 \in Y$ and $j - 1 \in Y$).

This general setup allows for several specific scenarios:

- to insert a knot into every non-zero knot interval: $j, j + 1 \in Y$ if and only if $u_j = u_{j+1}$,
- to increase multiplicity: $u_j = u_k$ where $j \in Y$ and $k \notin Y$,
- to retain a non-zero knot interval: $j, j + 1 \in Y$ for $u_j \neq u_{j+1}$.

We can insert more than one knot in an interval by using more than one complete subdivision step.

Overview 3.3.2

Like Schaefer's algorithm [83], this factorisation is best expressed in terms of the polar form of a B-spline. The key idea is a recipe for the polar arguments at every point P_i^σ of every stage σ of a refine-and-smooth algorithm. σ grows by steps of 2, so for a degree d algorithm there will be $\lceil d/2 \rceil$ stages. The recipe states that the arguments should include only the new knots that fall in a σ-sized region of $\boldsymbol{u}$, centred at i (see Figures 3.4 and 3.5). The polar arguments outside this region must belong to $\boldsymbol{t}$. As σ grows to d, this region grows to include d consecutive values in the new knot vector, and so the points P_i^d are control points on the subdivided B-spline. For example, for even degree, at $\sigma = 0$, all knots are in $\boldsymbol{t}$, the original knot vector. At each stage, zero, one or two new knots are added by the process described below, until at the final stage ($\sigma = d$), all polar arguments are uninterrupted sequences on the new knot vector, $\boldsymbol{u}$.

[9] Away from boundaries, $\boldsymbol{u}$ is precisely the knot vector for the subdivided B-spline, as subdivision can only *insert* knots, not remove them. At the boundaries, however, $\boldsymbol{u}$ may contain some extra knots that do not appear in the knot vector of the subdivided B-spline. I consider this detail in §3.3.7.

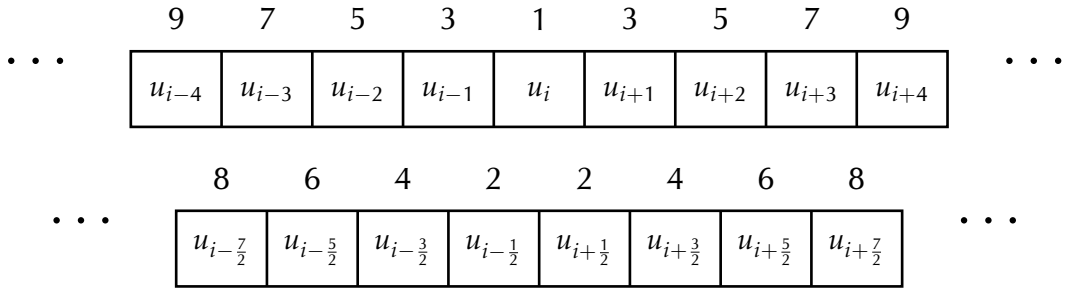


Figure 3.4: Knot insertion order for non-uniform symmetric refine and smooth. As σ increases, the polar arguments of a point P_i^σ include all the knots in an increasing region of $\boldsymbol{u}$. This is summarised above for d odd (top) and d even (bottom). A new knot ($k \notin Y$) is inserted when σ is equal to the value above the knot u_k. Figure 3.5 shows this process for an example where d is odd.

We can formalise this recipe by defining $\mathrm{cen}_i^\sigma(Z)$ to contain the σ indices in Z centred around i. Note that there are two cases here: either d is odd, σ is odd, and i is an element of Z. Or else d is even and σ is even, in which case i must lie between two elements of Z, because the control points of even-degree B-splines align with knot *intervals*, not with knots themselves. In either case, we now have that the polar arguments of P_i^σ are

$$u_j, \text{ where } j \in \mathrm{cen}_i^d(Y \cup \mathrm{cen}_i^\sigma(Z)) \tag{3.2}$$

See Figure 3.5 for an example of the polar arguments specified in this way as a point moves through the sequence of smoothing stages. In §3.3.6, I show that (3.2) is a self-consistent strategy which shares data along a polygon in the correct way, and Figures 3.8, 3.9 and 3.10 give examples which show how adjacent points interact under this algorithm; in each case the polar arguments of every point are specified by (3.2). First, however, I expand on this definition to show how the algorithm can be implemented.

The refine stage 3.3.3

The refine stage operates on the original control points Q and builds the sequence of points P_i^0 (for even d) or P_i^1 (for odd d). The two cases must be handled separately so that subsequent smoothing stages can symmetrically examine two knots at a time.

Even d

When $\sigma = 0$, $\mathrm{cen}_i^\sigma(Z) = \varnothing$ and so from (3.2), the polar arguments of P_i^0 are u_j for $j \in \mathrm{cen}_i^d(Y)$. But these knots are consecutive in $\boldsymbol{t}$, and so each point in P^0 is a control point on the original B-spline. Specifically, P_i^0 is equal to the point in Q corresponding to the knot interval which contains the interval indexed by i. After the refine stage, P^0 therefore contains two copies of any control points which correspond to knot intervals where knots are inserted.

Odd d

From (3.2) when $\sigma = 1$ we have that the polar arguments of each point P_i^1 are the d knots from $\boldsymbol{t}$ centred around i, but including u_i even if $i \notin Y$. That is, the polar arguments of P_i^1 are

$$u_j, \text{ where } j \in \mathrm{cen}_i^d(Y \cup \{\, i \,\}) \tag{3.3}$$

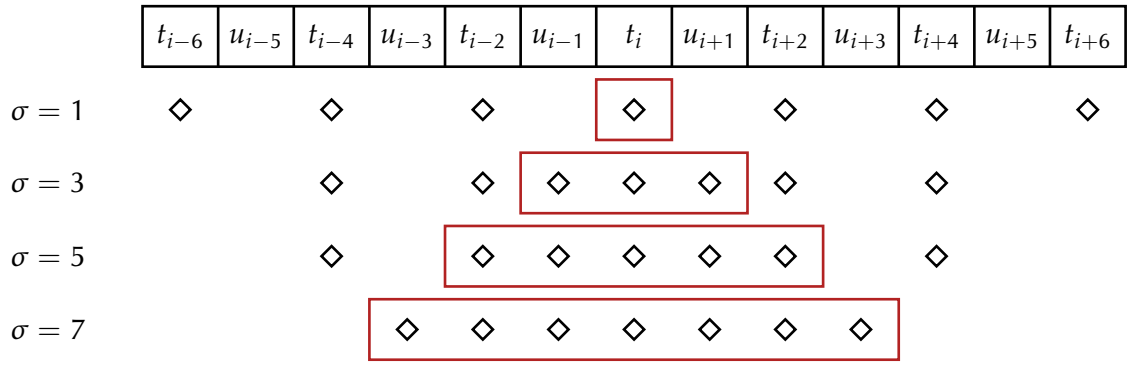


Figure 3.5: Polar arguments for a point through the sequence of refine and smooth stages. This figure shows subdivision at degree 7, where diamonds mark the knots which are included in the polar arguments of P_i^σ at each stage σ. Knots denoted t_i indicate that $i \in Y$, and red rectangles mark the region $\mathrm{cen}_i^\sigma(Z)$ from equation (3.2). This region indexes two additional knots in each stage until, when $\sigma = d$, it indexes all d knots surrounding i.

There are two cases: either $i \in Y$, or $i \notin Y$. If $i \in Y$, then (3.3) is again a collection of d consecutive knots from $\boldsymbol{t}$. For convenience, we can index the points Q such that Q_i is the point corresponding to the knot u_i. Then we directly obtain $P_i^1 = Q_i$.

If $i \notin Y$, we need an affine combination of the two points Q_{i-1} and Q_{i+1} (note that $i-1, i+1 \in Y$ because of our restrictions on Y). If $\alpha = \min(\mathrm{cen}_{i-1}^d(Y))$ and $\delta = \max(\mathrm{cen}_{i+1}^d(Y))$, then

$$P_i^1 = \frac{u_\delta - u_i}{u_\delta - u_\alpha} Q_{i-1} + \frac{u_i - u_\alpha}{u_\delta - u_\alpha} Q_{i+1}. \tag{3.4}$$

From the multiaffine property of the polar form, and the fact that Q_{i-1} and Q_{i+1} share $d-1$ polar arguments, (3.4) inserts the knot u_i as a polar argument for P_i^1, as required by (3.3).

After the refine stage, P^1 contains an additional point on every edge which corresponds to a knot interval where a knot is inserted.

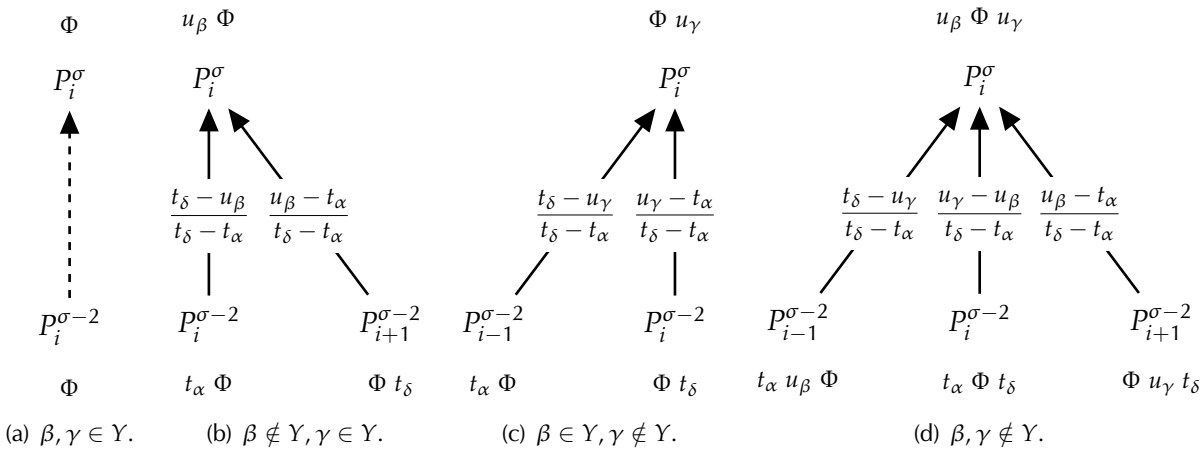
A smoothing stage 3.3.4

Each smoothing stage produces the points P^σ from the sequence of points $P^{\sigma-2}$, where the polar arguments of each point $P_i^{\sigma-2}$ include the $\sigma - 2$ knots surrounding i. In order for each P_i^σ to contain the correct set of polar arguments, we must take affine combinations of points in $P^{\sigma-2}$ to insert 0, 1, or 2 new knots. I distinguish between four possible actions, shown in Figure 3.6, where $\beta = \min(\mathrm{cen}_i^\sigma(Z))$ and $\gamma = \max(\mathrm{cen}_i^\sigma(Z))$.

Where α and δ appear, they are, respectively, the least and greatest indices of the polar arguments for a point: $\alpha = \min(\mathrm{cen}_i^d(Y \cup \mathrm{cen}_i^{\sigma-2}(Z)))$ and $\delta = \max(\mathrm{cen}_i^d(Y \cup \mathrm{cen}_i^{\sigma-2}(Z)))$. These values can either be stored for each i, or calculated on-the-fly. The four cases are:

(a) No new knots

This action is illustrated in Figure 3.6(a). If $\beta, \gamma \in Y$, then $P_i^{\sigma-2}$ already contains all the polar arguments in a σ-sized region of $\boldsymbol{u}$. We can therefore directly copy $P_i^{\sigma-2}$ to P_i^σ.



(a) $\beta, \gamma \in Y$. (b) $\beta \notin Y, \gamma \in Y$. (c) $\beta \in Y, \gamma \notin Y$. (d) $\beta, \gamma \notin Y$.

Figure 3.6: The four smoothing actions that form a point P_i^σ. Points are shown with their polar arguments, and t_α and t_δ indicate knots u_α and u_δ with $\alpha, \delta \in Y$. Within each diagram, Φ denotes the same collection of (not necessarily consecutive) knots; typically u_β and u_γ will be placed somewhere in the interior of the knots Φ. For (a), $|\Phi| = d$. For (b) and (c), $|\Phi| = d - 1$ and for (d), $|\Phi| = d - 2$.

(b) New knot with index less than i

In this scenario we have $\gamma \in Y$ but $\beta \notin Y$. The polar arguments for $P_i^{\sigma-2}$ therefore already contain u_γ, but we need an affine combination to insert the knot u_β. The coefficients that multiply $P_i^{\sigma-2}$ and $P_{i+1}^{\sigma-2}$ are shown in Figure 3.6(b).

Note that in moving from $P_i^{\sigma-2}$ to P_i^σ, the knot that is displaced from the polar arguments is t_α: the knot with the least index. This property holds for each action in Figure 3.6; where a knot is inserted, it replaces a knot from $\boldsymbol{t}$ on the same side of i. The replaced knot is also always further from i than the new knot. From this, we can conclude that inserting all the new knots in a d-sized region of $\boldsymbol{u}$ (when $\sigma = d$) results in polar arguments that are consecutive in $\boldsymbol{u}$, which is the condition for P^d to be the control points after subdivision.

(c) New knot with index greater than i

This action is illustrated in Figure 3.6(c), and is completely symmetrical to the previous case. Here we have $\beta \in Y$ but $\gamma \notin Y$, and an affine combination of $P_{i-1}^{\sigma-2}$ and $P_i^{\sigma-2}$ inserts the knot u_γ.

(d) New knots on both sides of i

Here we have β and $\gamma \notin Y$, and so the smoothing action must introduce both u_β and u_γ into the polar arguments for P_i^σ. The required affine combination is shown in Figure 3.6(d).

This calculation uses three points, and so in the notation of property C from page 32, the result is a smoothing matrix with bandwidth three. However, we can factorise action (d) into three affine combinations of two points in order to keep smoothing as local as possible. This splits a single smoothing matrix of bandwidth three into two smoothing matrices, each of bandwidth two. To do so, we choose a pivot, x, and replace

$$P_i^\sigma = \frac{t_\delta - u_\gamma}{t_\delta - t_\alpha} P_{i-1}^{\sigma-2} + \frac{u_\gamma - u_\beta}{t_\delta - t_\alpha} P_i^{\sigma-2} + \frac{u_\beta - t_\alpha}{t_\delta - t_\alpha} P_{i+1}^{\sigma-2}$$

with the calculations

$$D = \frac{t_\delta - u_\gamma}{t_\delta - x} P_{i-1}^{\sigma-2} + \frac{u_\gamma - x}{t_\delta - x} P_i^{\sigma-2}$$

$$E = \frac{x - u_\beta}{x - t_\alpha} P_i^{\sigma-2} + \frac{u_\beta - t_\alpha}{x - t_\alpha} P_{i+1}^{\sigma-2}$$

$$P_i^\sigma = \frac{t_\delta - x}{t_\delta - t_\alpha} D + \frac{x - t_\alpha}{t_\delta - t_\alpha} E.$$

To maintain non-negative weights, x must lie between u_β and u_γ. For example we could set x equal to the mean of the knot interval indexed by i (when d is even), or u_i (when d is odd).

Examples 3.3.5

To illustrate the above algorithm, consider the example shown in Figure 3.7, where $d = 5$. I will continue to write $b(\cdot)$ for the polar form of B, and use $Z = \{3, \ldots, 17\}$, with

$Y =$	3	4		6	7	8		10		12		14	15		17
$Z \setminus Y =$			5				9		11		13			16	
$\boldsymbol{u} =$	0	0	2	4	4	4	8	12	14	16	17	18	18	19	20

Since d is odd, the refine stage produces P^1. Figure 3.8 shows the algorithm in full, and here I consider two example points. $8 \in Y$, so $P_8^1 = Q_8 = b(u_6, u_7, u_8, u_{10}, u_{12})$. On the other hand, $11 \notin Y$, so P_{11}^1 is an affine combination of Q_{10} and Q_{12}:

$$\begin{aligned} P_{11}^1 &= \frac{u_{15} - u_{11}}{u_{15} - u_7} Q_{10} + \frac{u_{11} - u_7}{u_{15} - u_7} Q_{12} \\ &= b(u_8, u_{10}, u_{11}, u_{12}, u_{14}) \end{aligned}$$

As these two points progress through the smoothing stages, we find that

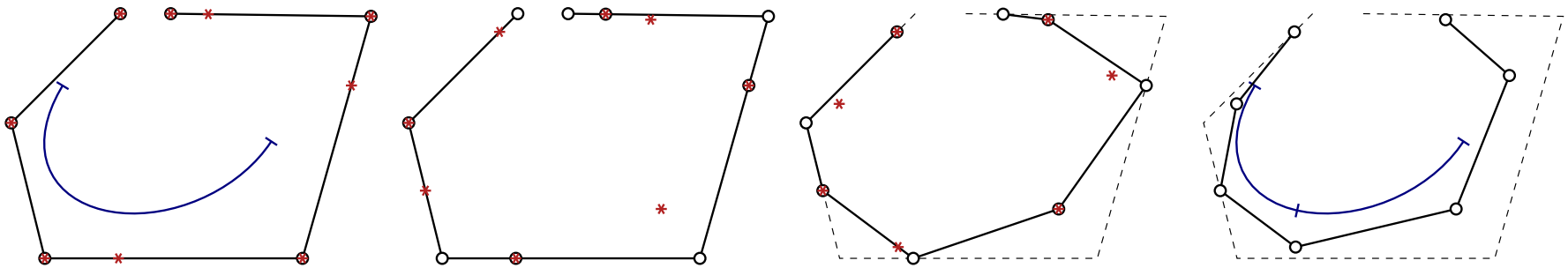
$$P_8^3 = b(u_6, u_7, u_8, u_9, u_{10}),$$

inserting u_9 using the affine combination in Figure 3.6(c). Since P_8^3 has polar arguments consecutive in Z, we know that it is already a control point on the subdivided B-spline, and so inevitably $P_8^5 = P_8^3$ as in Figure 3.6(a). For the point centred on u_{11}, we have $P_{11}^3 = P_{11}^1 = b(u_8, u_{10}, u_{11}, u_{12}, u_{14})$ since $10, 12 \in Y$, also using the action from Figure 3.6(a). For P_{11}^5, however, we find that both 9 and $13 \notin Y$, so $P_{11}^5 = b(u_9, u_{10}, u_{11}, u_{12}, u_{13})$ using the action in Figure 3.6(d). This also results in polar arguments with indices that are consecutive in Z, as required.

Figures 3.9 and 3.10 show the patterns that arise in the unselective case where a new knot is inserted in every existing knot interval. Where d is odd, as in Figure 3.9, the algorithm uses only the actions shown in Figure 3.6(a) and (d). Conversely when d is even, as in Figure 3.10, we only require the actions shown in Figure 3.6(b) and (c).

Proof 3.3.6

In §3.3.3 and §3.3.4, we considered a point P_i^σ as σ increases to d, and showed that the refine and smooth stages result in a control point on the subdivided B-spline. So far, however, I have not



(a) Refine stage. (b) First smoothing stage. (c) Second smoothing stage. (d) Subdivided B-spline.

Figure 3.7: An example factorisation for a non-uniform quintic B-spline with multiple knots. This figure shows the example from §3.3.5; points marked with stars are passed to the next stage. A knot is inserted in every non-zero interval: before subdivision, the knot vector is [0, 0, 4, 4, 4, 12, 16, 18, 18, 20] and afterwards, it is [0, 2, 4, 4, 4, 8, 12, 14, 16, 17, 18] (the truncation is explained in §3.3.7).

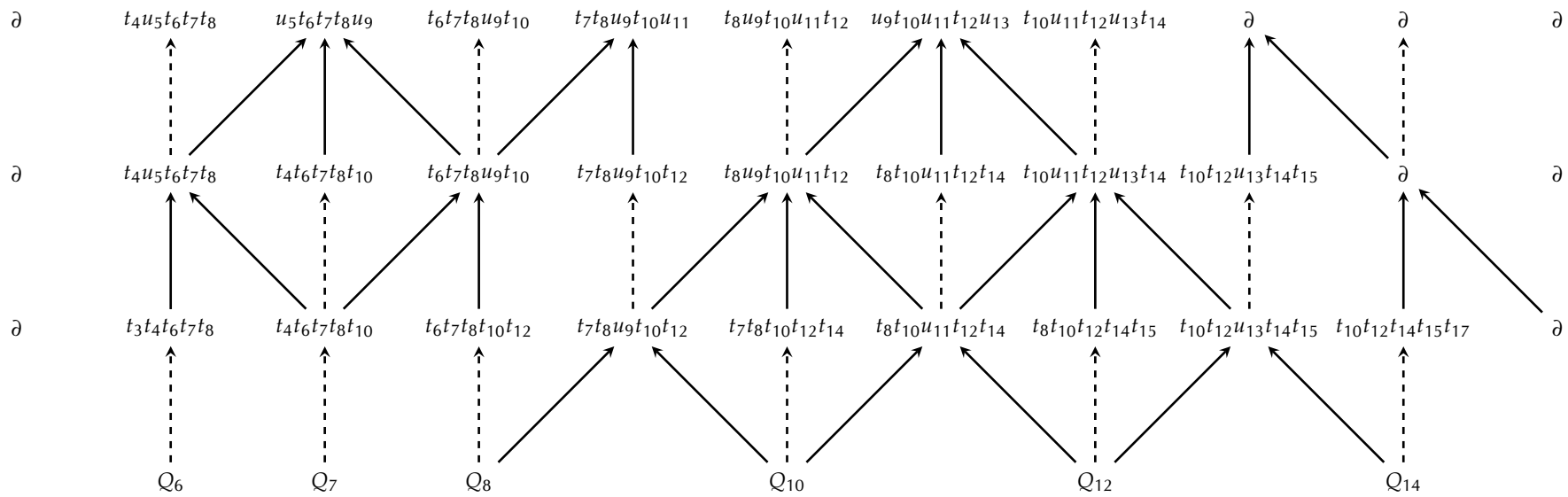


Figure 3.8: The example from §3.3.5 in full; see Figure 3.7 for an example application. At the bottom are the input points Q, above which are P^1, P^3, and finally the output P^5. Points are represented as a list of arguments to the polar form b of B.

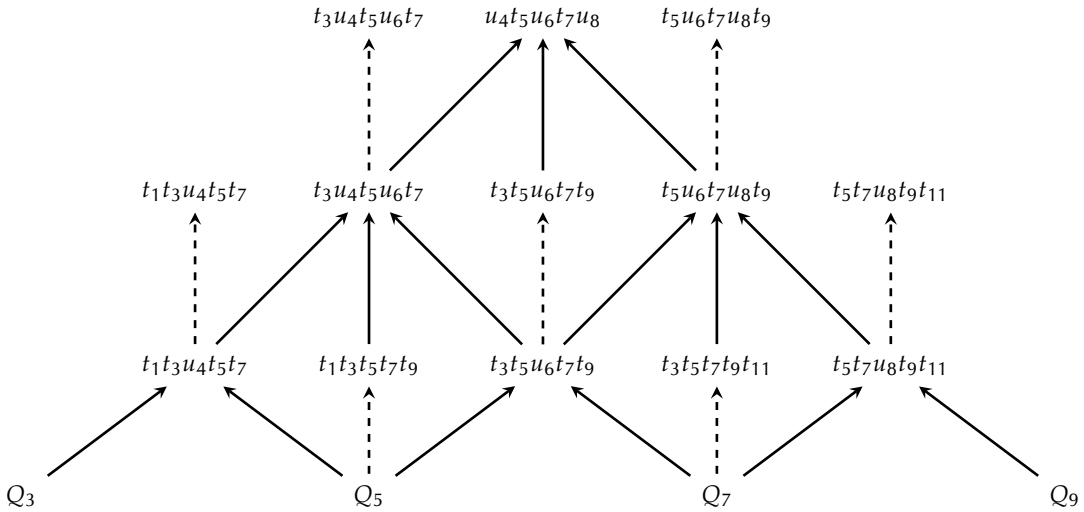


Figure 3.9: Refine-and-smooth factorisation for odd d in the unselective case. This figure shows the computation of three new points when $d = 5$ (quintic), ignoring end conditions.

justified the polar arguments for $P_{i-1}^{\sigma-2}$ and $P_{i+1}^{\sigma-2}$ shown in Figure 3.6. In this section I address the omission, thereby proving that the actions described in §3.3.4 mesh together correctly as i varies along a polygon.

Assume that the polar arguments of all points $P_i^{\sigma-2}$ contain the $\sigma - 2$ knots surrounding i. §3.3.4 showed that this property holds for P^σ, with a region of size σ, as long as we can show that the polar arguments take the form shown in Figure 3.6. There is nothing to prove for Figure 3.6(a), and we will consider just the cases in Figure 3.6(b) and (d). The proof for Figure 3.6(c) is completely symmetrical to that for 3.6(b).

New knot with index less than i

First, note that the polar arguments for $P_i^{\sigma-2}$ contain all the knots with indices in the range $\mathrm{cen}_i^{\sigma-2}(Z)$. The equivalent range for $P_{i+1}^{\sigma-2}$ is $\mathrm{cen}_{i+1}^{\sigma-2}(Z)$. These ranges overlap on $\sigma - 3$ values, and the polar arguments of both points therefore contain the knots indexed by the overlap. At the ends of these ranges are $\min(\mathrm{cen}_i^{\sigma-2}(Z))$ and $\max(\mathrm{cen}_{i+1}^{\sigma-2}(Z))$, and if either of these indices is not in Y (i.e. indexes a new knot), then that knot will not appear in the polar arguments of one of the points. On the other hand, if both $\min(\mathrm{cen}_i^{\sigma-2}(Z))$ and $\max(\mathrm{cen}_{i+1}^{\sigma-2}(Z)) \in Y$, then the knots indexed by these values will appear in both sets of polar arguments.

Now observe that, since we are performing the action in Figure 3.6(b), we already know that $\gamma \in Y$ and $\gamma = \max(\mathrm{cen}_i^{\sigma}(Z)) = \max(\mathrm{cen}_{i+1}^{\sigma-2}(Z))$. We also know that $\beta = \min(\mathrm{cen}_i^{\sigma}(Z)) \notin Y$ and so our restrictions on Y enforce that $\beta + 1 \in Y$ and $\beta + 1 = \min(\mathrm{cen}_i^{\sigma-2}(Z))$. Therefore the polar arguments of $P_i^{\sigma-2}$ and $P_{i+1}^{\sigma-2}$ overlap on $d - 1$ knots Φ, differing only in the knot with lowest index t_α, and the knot with the highest index t_δ. So the polar arguments do, in fact, take the form shown in Figure 3.6(b).

New knots on both sides of i

Here we have that $\beta \notin Y$ and $\gamma \notin Y$. $\beta = \min(\mathrm{cen}_i^{\sigma}(Z)) \in \mathrm{cen}_{i-1}^{\sigma-2}(Z)$, so u_β is certainly one of the polar arguments for $P_{i-1}^{\sigma-2}$. The same argument shows that u_γ is a polar argument for $P_{i+1}^{\sigma-2}$.

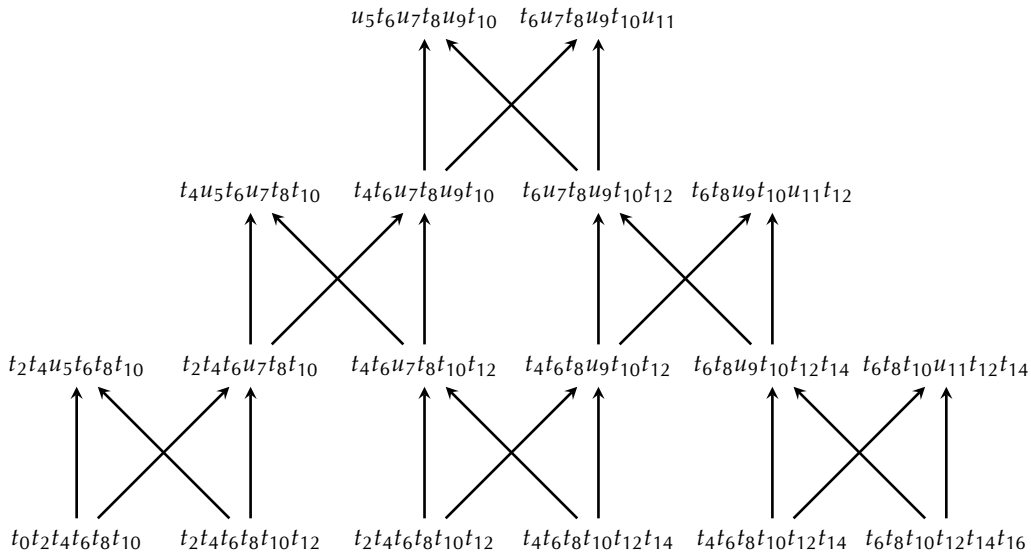


Figure 3.10: Refine-and-smooth factorisation for even d in the unselective case. This figure shows the computation of two new points when $d = 6$ (sextic), ignoring end conditions.

If we can show that the polar argument with minimum index is the same for $P_{i-1}^{\sigma-2}$ and $P_i^{\sigma-2}$, and that the polar argument with maximum index is the same for $P_i^{\sigma-2}$ and $P_{i+1}^{\sigma-2}$, then it will necessarily follow that all three sets of polar arguments share $d-2$ knots Φ.

We know from (3.2) that we can find the index of the smallest polar argument in $P_i^{\sigma-2}$ by selecting the $1+(d-\sigma)/2$ largest value in Y which is smaller than $\min(\mathrm{cen}_i^{\sigma-2}(Z))$. Furthermore, $\beta = \min(\mathrm{cen}_{i-1}^{\sigma-2}(Z)) \notin Y$. So the sets $\{y \in Y : y < \beta+1\}$ and $\{y \in Y : y < \beta\}$ are identical. Therefore the $1+(d-\sigma)/2$ largest value smaller than $\min(\mathrm{cen}_{i-1}^{\sigma-2}(Z))$ is the same as that less than $\min(\mathrm{cen}_i^{\sigma-2}(Z))$. The polar arguments for $P_{i-1}^{\sigma-2}$ and $P_i^{\sigma-2}$ therefore share a minimum-index knot t_α. Naturally, we can apply a symmetrical argument to show a common maximum-index knot t_δ for $P_i^{\sigma-2}$ and $P_{i+1}^{\sigma-2}$.

We can therefore justify the polar arguments in Figure 3.6, which validates the affine combinations I used in §3.3.4 and proves that the algorithm works correctly if we are far enough from any bounds of Z. It remains to show that the algorithm handles end conditions correctly.

End conditions 3.3.7

The bounds on the domain of the B-spline (where they exist) are u_a and u_b, where a is the dth smallest index in Y and b is the dth largest. Therefore any point that has polar arguments with indices entirely below a, or above b, should be discarded so that the limit curve remains invariant. The knot vector of B after subdivision can therefore be shorter than $\boldsymbol{u}$. With $\boldsymbol{u}$ as in §3.3.5, for example, we have $a = 8$ and $b = 10$, with the knot vector after subdivision $\{u_4, \ldots, u_{14}\} \subset \boldsymbol{u}$.

The domain of the B-spline also allows us to specify the number of points which must be computed in the refine stage. For even degree, the refine stage builds P_i^0, where i starts by indexing the interval which begins at the value $d/2$ smallest in Y. Similarly, the final i indexes the interval which ends at the value $d/2$ largest in Y. This is the largest possible range which maintains a polar argument with index in $[a\ b]$ for all points in P^0. In the same way, when d is odd, i ranges from the $\lceil d/2 \rceil$th smallest index in Y to the $\lceil d/2 \rceil$th largest.

Although these definitions guarantee that points are valid in P^1 or P^0, we now need to establish which points are discarded after smoothing. Each smoothing stage may, potentially, result in another point at each end with polar arguments indexed outside of the valid range. Fortunately there is a simple implementation which handles this complication automatically. We start by defining a boundary marker point, ∂, and append it to the results of the refine stage. For the example from §3.3.5, the result of the refine stage is then

$$\{ \quad \partial \quad P^1_6 \quad P^1_7 \quad P^1_8 \quad P^1_9 \quad P^1_{10} \quad P^1_{11} \quad P^1_{12} \quad P^1_{13} \quad P^1_{14} \quad \partial \quad \}.$$

Now we simply define an affine combination of points, $(1-x)P_1 + xP_2$ to be equal to ∂ whenever[10] $P_1 = \partial$ or $P_2 = \partial$. For our running example, $P^3_{14} = \partial$ (see Figure 3.8), because the affine combination used to form P^3_{14} takes a contribution from $P^1_{15} = \partial$. Note that this action (shown in Figure 3.6(b)) would otherwise displace t_{10} from this point's polar arguments, which would subsequently be indexed entirely above $b = 10$ and so P^3_{14} cannot be a control point on the subdivided B-spline. After the next smoothing stage, we similarly have $P^5_{13} = \partial$, and also note that P^5_{13} is not one of the required control points.

In fact this implementation works because the actions in Figure 3.6 take a contribution from $P^{\sigma-2}_{i+1}$ *whenever* the minimum index is increased (potentially above b). If P^σ_i would have polar arguments indexed entirely above b, then the same must be true of $P^{\sigma-2}_{i+1}$, and so $P^{\sigma-2}_{i+1} = \partial$ gives $P^\sigma_i = \partial$. In the same way, the actions take a contribution from $P^{\sigma-2}_{i-1}$ whenever the maximum index in the polar arguments of P^σ_i potentially drops below a.

This factorisation therefore neatly handles multiple knots using a single framework without any special cases. In particular, this includes Bézier end conditions (where there are knots of multiplicity d terminating $\boldsymbol{t}$ and $\boldsymbol{u}$), which are widely used for the boundary of B-spline curves and surfaces.

Summary 3.4

The factorisation described in §3.3 provides a solution to the problem posed at the start of this chapter, as it allows non-uniform general-degree knot insertion to be computed using a refine-and-smooth factorisation in the spirit of the Lane-Riesenfeld algorithm. Unlike the Lane-Riesenfeld algorithm, the affine combinations used by smoothing stages depend on the knot values and may therefore vary both along a polygon and between stages. Like Schaefer's algorithm, this factorisation also does not produce the control polygons for lower-degree B-splines as the result of intermediate smoothing stages. §3.1 showed that this property limits the possible knot vectors, whereas the algorithm described in §3.3 does not constrain the choice of knots. New knots may be placed at the midpoints of existing intervals, such that the knot vector tends towards piecewise uniformity, or using any other criteria. Furthermore, the intervals for insertion may be chosen as well as the new knot values.

The work in this chapter therefore lays the univariate foundations for NURBS-compatible subdivision surfaces, and is trivially extended to a tensor-product form that can be applied to regular meshes. For irregular meshes that can contain extraordinary vertices, we need to consider generalisations of this tensor-product form. This is the focus of Chapters 4 and 5.

[10] This is true for any x, including $x = 0$ or $x = 1$, as the boundaries are not affected by the *values* of the knots in $\boldsymbol{t}$, but only by their *indices* in Y.

Extraordinary vertices

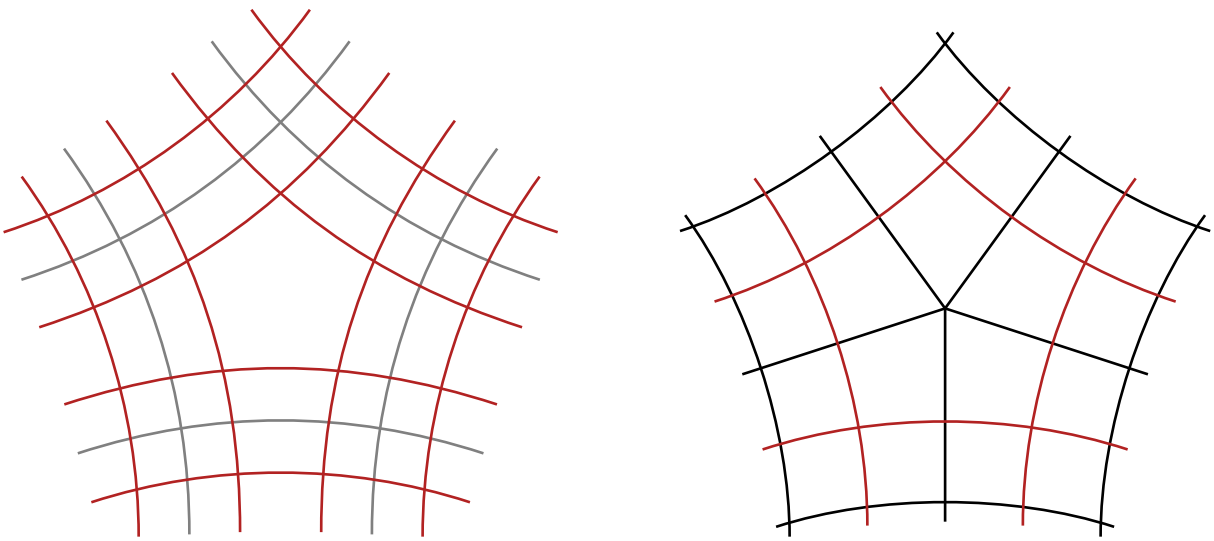
4

This chapter presents research that has also been published in the paper:

> T. J. Cashman, U. H. Augsdörfer, N. A. Dodgson and M. A. Sabin. NURBS with Extraordinary Points: High-degree, Non-uniform, Rational Subdivision Schemes. *ACM Transactions on Graphics*, 28(3):#46, 1–9, 2009.

Chapter 3 described a refine-and-smooth factorisation for B-spline knot insertion in a non-uniform, general-degree setting. This factorisation makes it possible, for the first time, to create non-uniform general-degree subdivision surfaces, thereby extending NURBS to arbitrary topologies. To do so, this chapter generalises the knot insertion rules I presented in §3.3 to take account of irregular mesh connectivity.

The refinement patterns for even- and odd-degree B-splines, shown in Figures 2.8(a) and (b) respectively, lead to the two different types of irregularity shown in Figure 4.1. The dual refinement, where degree is even, leads to *extraordinary faces*, which have fewer or greater than four edges. The primal refinement, where degree is odd, leads to *extraordinary vertices*, which are connected to fewer or greater than four edges. In both cases, the extraordinary element is preserved by each subdivision step, and is therefore present in the limit surface as a *singularity* surrounded by regular spline surface [54]. In practice, the primal case is the most important of the two, as odd degrees are used more often than even [91]. There are several factors which contribute to this preference:



(a) Even degree. (b) Odd degree.

Figure 4.1: Example refinement patterns for irregularities in the control mesh. As in Figure 2.8, original edges are drawn in black where they form part of the refined pattern, or in grey where they do not.

- many applications evaluate a function of surface normals; reflections in a surface are one example. For such first-order functionals to give a smooth result, the surface itself must be curvature continuous, and cubic is the lowest degree at which this continuity is available.
- primal refinement aligns B-spline patches with the control mesh, which can simplify the handling of boundaries.
- primal subdivision schemes are a better fit with graphics cards and displays, which can seamlessly handle high-valency vertices but require many-sided polygons to be tessellated.
- designers are likely to be more familiar with the primal case from using the popular cubic B-splines, and might therefore choose to stay in the primal setting where possible.

This chapter and the next will therefore focus solely on NURBS-compatible subdivision schemes for odd degrees[11]. This dissertation does not contain a complete solution for the even-degree case, but I describe some preliminary work on this subproblem in §6.4.

The Catmull-Clark subdivision scheme [10] generalises bicubic B-splines and therefore uses the primal refinement pattern shown in Figure 4.1(b), which preserves extraordinary *vertices* rather than extraordinary *faces*. However, Catmull-Clark subdivision allows extraordinary faces to be included in a control mesh by using the first subdivision step to insert an extraordinary vertex in the interior of each extraordinary face; the new extraordinary vertices are then preserved by subdivision instead. Primal schemes with extraordinary faces do not fit well with the generalisation of knot vectors I present in §4.1, however, and also lead to inferior surface quality [92]. Furthermore, extraordinary faces are not necessary to achieve the goal of NURBS-compatible subdivision surfaces, as NURBS control meshes use only four-sided faces, and so extraordinary faces are not needed to represent existing NURBS surfaces exactly. Extraordinary faces are not required to create arbitrary-topology surfaces, either, as extraordinary *vertices* are sufficient to grant topological freedom. Therefore I do not consider extraordinary faces further; §6.5 describes the future work that would be required to include them.

In summary, this dissertation makes two restrictions on NURBS-compatible subdivision schemes, by considering only:

- subdivision at odd degrees, rather than even,
- control meshes without extraordinary faces.

As a result, the subdivision schemes use exactly the refinement pattern shown in Figure 4.1(b): the control mesh is permitted to include extraordinary vertices, but all faces must be four-sided. This chapter generalises the knot insertion rules described in §3.3 for control meshes of this type.

Incorporating knot intervals 4.1

The knot insertion algorithm described in §3.3 requires an original knot vector and a refined version as part of its input. In the bivariate case, we need to provide these knot vectors for each of the two orthogonal directions in which the surface is subdivided. Before we can generalise the knot insertion rules to irregular meshes, we therefore need a way to specify these knot vectors on a mesh containing extraordinary vertices.

[11] However, the algorithm given in §4.3 is also applicable for even degrees.

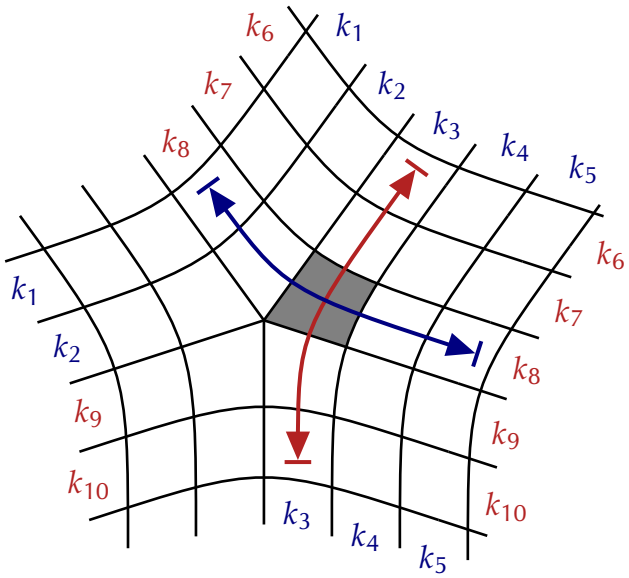


Figure 4.2: Local knot vectors for a mesh containing extraordinary vertices. Every face has a local knot vector in two directions, which we can construct by following a strip of quadrilateral faces. For the shaded face, knot spacings in the blue direction are $\{k_1, k_2, k_3, k_4, k_5\}$ and in the red direction are $\{k_6, k_7, k_8, k_9, k_{10}\}$. This example collects knot vectors at the length required for subdivision at degree 5.

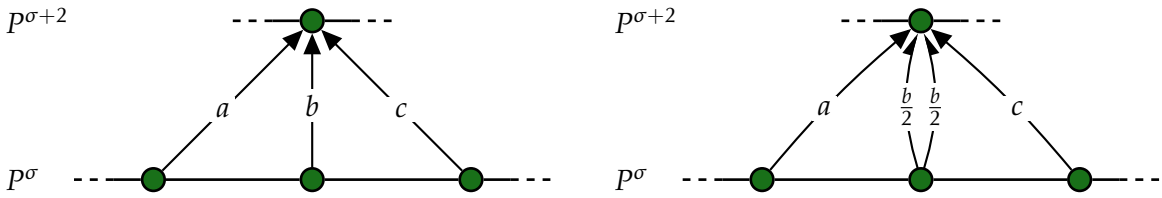
I use the same formulation as Sederberg et al. [89], which works from the observation that we can specify the knot vector for an odd-degree B-spline curve by annotating each edge of a control polygon with the *interval* between adjacent values in the knot vector. We can use the same idea to annotate edges of the control mesh with knot intervals in the bivariate case (see Figure 4.2).

If a NURBS control mesh is annotated in this way, then the tensor-product structure of the surface means that every quadrilateral face has equal knot intervals on opposite edges. Another formulation of the same property is that every face occupies a rectangle in parameter space. There have been previous attempts to create NURBS-like formalisms without this constraint. When Sederberg et al. [89] created NURSS (§2.3.1), they removed this property and instead allowed every edge to be annotated separately. The surfaces created by Müller et al. [51] offer the same freedom; they call any quadrilateral that does *not* occupy a rectangle in parameter space an *augmented face*. To create T-splines, however, Sederberg et al. [88] disallowed augmented faces in order to permit T-junctions in a control mesh.

For a superset of NURBS, we do not need the level of generality provided by NURSS, where every edge is allocated a separate knot interval. I therefore make the same restriction as T-splines and NURBS: that opposite edges of a face must be annotated with equal knot spacings. The result is that knot intervals are defined for a whole strip of quadrilateral faces rather than for a single edge. It is possible that this restriction could be lifted by future work (see §6.5), but the experience of Sederberg et al. suggests that it may introduce an unacceptable level of complexity to do so.

Generalised knot insertion 4.2

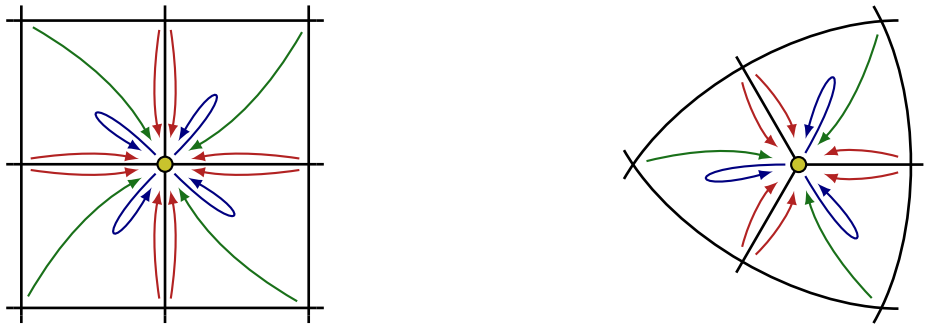
The knot insertion algorithm from §3.3 holds property C on page 32, so each affine combination takes contributions from direct neighbours only. We can therefore visualise the effect of a



(a) A smoothing stage on a single control point in the univariate knot insertion algorithm, where $a + b + c = 1$. Contributions come only from the point and its two immediate neighbours.

(b) We can consider the contributions from each side of a point separately, by using only half weights for each of the contributions from the predecessor point.

Figure 4.3: Calculating univariate smoothing stages one edge at a time. Each of the actions in Figure 3.6 can be expressed in the form shown in (a), and can therefore be calculated using the four contributions shown in (b).



(a) In regular regions.

(b) Around a vertex of valency 3.

Figure 4.4: Calculating bivariate smoothing stages one face at a time. There are contributions to a vertex ● from each of the surrounding faces. The weights in each face are taken from the tensor product of Figure 4.3(b); the pairs of contributions shown in red define the influence of edge-connected vertices and the contributions shown in blue combine to give the influence of a vertex on its successor. In a NURBS mesh, shown in (a), these 16 weights sum to 1 and compute the same affine combination as the tensor product of Figure 4.3(a).

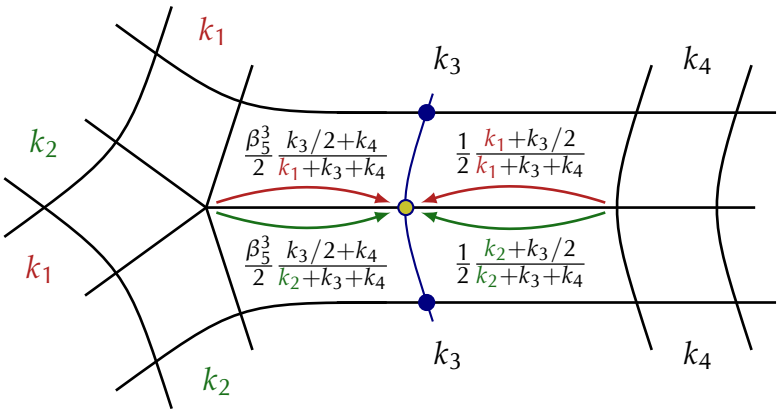


Figure 4.5: An example refine stage for degree 3. The interval k_3 has a knot inserted at its midpoint and other intervals are unmodified. The knot insertion algorithm therefore introduces vertices ●, ● along the subdivided interval, and this figure shows the unnormalised weights used to form the central vertex ●. The fractions that involve k_i are from the curve case (§3.3.3), the factors of one half are because I treat each side of each edge separately, as in Figure 4.3(b), and the multiplier, β_5^3, is required to get bounded curvature in the bivariate case, as explained in Chapter 5.

smoothing stage on a single point as shown in Figure 4.3(a), although either (or both) of the weights a and c may be zero. Figure 4.3(b) shows that we can consider the same smoothing operation *one edge at a time*. This observation allows us to generalise the factorisation to meshes that can contain extraordinary vertices, as in the surface case we can calculate the affine combinations of smoothing stages *one face at a time*. The weights used in the affine combinations are defined using the tensor product of univariate knot insertion when considered one edge at a time, as shown in Figure 4.3(b). For a NURBS control mesh, each smoothing stage calculates weights in the four faces surrounding a vertex, as shown in Figure 4.4(a). These weights combine to give the tensor product of a smoothing stage in the form shown in Figure 4.3(a), as required for NURBS knot insertion.

The same idea allows us to calculate the refine stage one face at a time, generalising the refine stage to irregular meshes, too. Again, we have the local knot vectors described in §4.1, and need to generalise the tensor product of the refine stage for odd degrees (§3.3.3). We must therefore introduce vertices on existing edges and within existing faces. In the face case, the new vertex position is given by the tensor product of univariate refinement in the two directions. In the edge case, the two univariate contributions are split into four: two on each side of the edge. Figure 4.5 shows an example of how these contributions can be applied to a bivariate case.

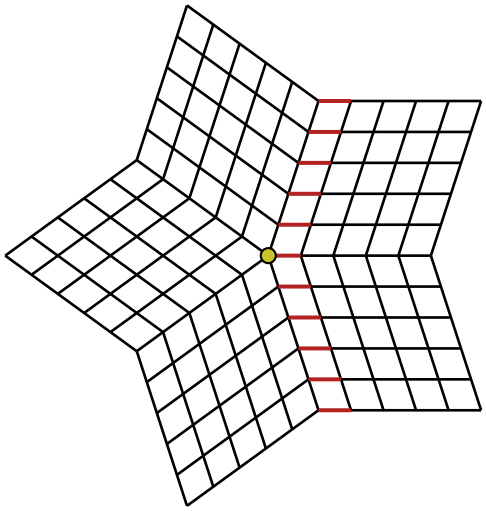
Where a mesh is irregular, this generalisation leads to points where the sum of contributing weights is not equal to one. For invariance under solid-body transformations[12], however, it is important that every combination is affine, and must therefore use weights that sum to one. Like Augsdörfer et al. [2], we can ensure this is true by normalising each affine combination using the sum of contributing weights. This is necessary at extraordinary vertices, since a vertex of valency n receives $4n$ contributions instead of 16. The multipliers I use for bounded curvature (see Chapter 5) make it necessary to normalise every affine combination that has a contribution from an extraordinary vertex. Normalisation is also sometimes necessary along the rays which emanate from extraordinary vertices, because the two sides of the ray can use different knot vectors. It is therefore easiest to normalise every affine combination, regardless of position in the mesh. This also handles faces with more than one extraordinary corner without special cases, making it easier to implement the scheme.

Knot insertion strategy 4.3

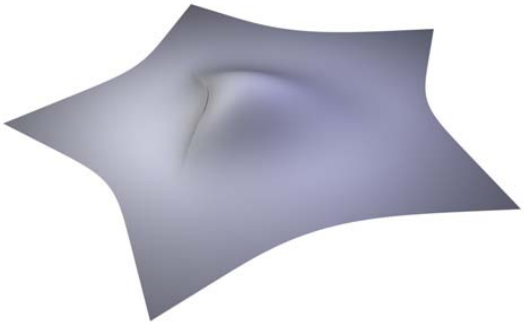
Combining the knot insertion algorithm described in §3.3 and its generalisation in §4.2 with local knot vectors from §4.1, we are able to create non-uniform general-degree subdivision schemes, thus tackling the two main challenges described in §2.4. For regular meshes, inserting knots with the resulting framework is identical to NURBS knot insertion, and therefore produces a NURBS limit surface[13]. As knot insertion is a local operation, the same is true at any *region* of a control mesh that is far enough from extraordinary vertices to be isolated from their effect. In these regions the NURBS limit surface is unique, and is therefore independent of the sequence of inserted knots.

[12] Invariance under solid-body transformations is usually the motivation for taking only affine combinations, but doing so actually grants invariance under all affine transformations, including shears.

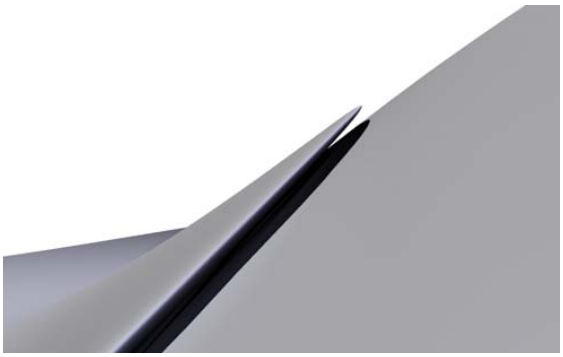
[13] Under knot insertion a regular mesh will converge to the NURBS surface, which has a known closed form, if the limiting collection of knots is dense in the parameter space. This is true for any reasonable knot insertion strategy.



(a) Projection of the control mesh onto the xy-plane. All z co-ordinates are zero apart from the single point ●, which is elevated. All knot intervals have the same size apart from the interval assigned to the edges drawn in red, which is fifty times larger than the others.

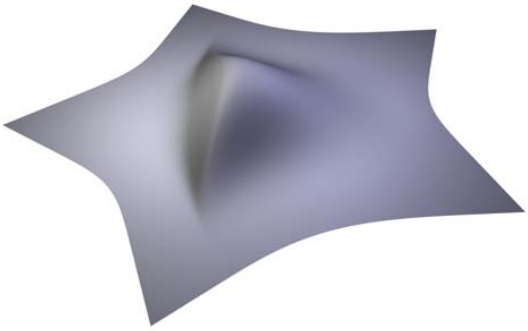


(b) Overview.

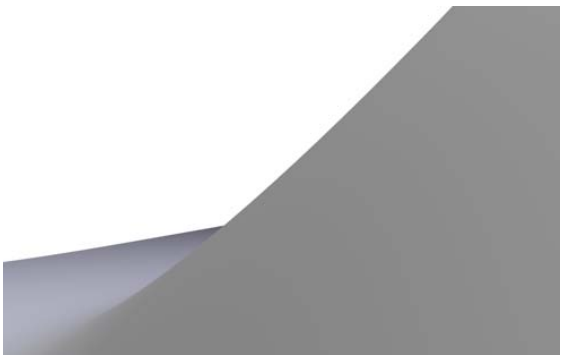


(c) Closer view of the 'fold' on the left-hand side of (b).

Figure 4.6: Failure of the direct midpoint knot insertion strategy. This degree 5 surface was generated using the control mesh shown in (a) by inserting knots into the midpoint of every interval. This results in an undesirable fold in the surface. See Figure 4.7 for an improved result, created using the knot insertion strategy described in §4.3.



(a) Overview.



(b) A similar view to Figure 4.6(c).

Figure 4.7: Removing the problem highlighted by Figure 4.6. This degree 5 surface was defined using the control mesh shown in Figure 4.6(a), but using the knot insertion strategy described in §4.3. The resulting surface is convex, like the control mesh.

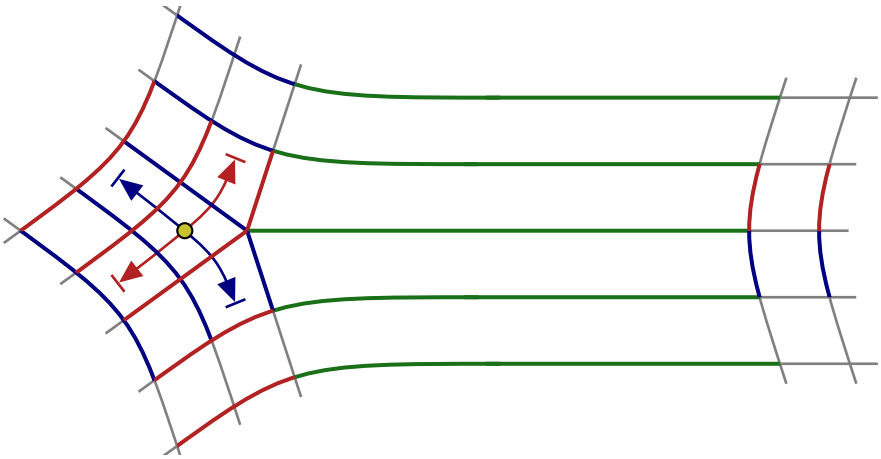


Figure 4.8: Knot vectors can be independent of knot intervals in vertex-connected faces. Two knot vectors are used to position the new point ●; the red knot vector is calculated from the knot intervals assigned to the edges drawn in red, and the blue knot vector from the knot intervals assigned to the edges drawn in blue. The point ● is therefore positioned independently of the large knot interval to the right of the extraordinary vertex, where the strip of edges is drawn in green. This creates the problem shown in Figure 4.6 if knots are inserted in the centre of every knot interval.

Around extraordinary vertices, however, the limit surface is *created* as a result of generalised knot insertion: it is not known in advance. The sequence in which knots are inserted is therefore important. Previous non-uniform subdivision schemes [51, 89] have inserted a new knot at the midpoint of every existing knot interval. This produces an unnecessary number of knots at any multiple knot, as Miura and Masuda [49] observed, but the knot insertion algorithm described in §3.3 is selective, and so we could remove this disadvantage by choosing not to subdivide *zero* knot intervals, and subdivide only every *non-zero* knot interval at its midpoint instead.

Unfortunately, this natural insertion strategy has shortcomings, as shown in Figure 4.6, where a convex control mesh results in a limit surface with a concave fold. The *variation-diminishing* property of B-splines guarantees that this can never occur using NURBS surfaces on a regular mesh. It is therefore highly undesirable for the inclusion of extraordinary vertices in a control mesh to result in this effect. The problem arises because a large knot interval, adjacent to a high-valency vertex, has no influence on the local knot vectors which are constructed on the other side of the extraordinary vertex (see Figure 4.8 for an example). The knot insertion strategy described in this section is able to alleviate this problem by taking account of the relative sizes of knot intervals when selecting which knots to insert; as a result, we obtain the surface shown in Figure 4.7 instead of the one shown in Figure 4.6. The knot insertion strategy works in two phases:

- subdividing large intervals first, and then
- creating uniform extraordinary regions.

Subdividing large intervals first 4.3.1

The problem shown in Figure 4.6 results from inserting a knot into a small knot interval while there are relatively large knot intervals nearby. If every vertex has valency three or four, then each

(a) Subdivide every interval. (b) Subdivide where $k_i > \frac{1}{2}\max(\boldsymbol{k})$. (c) Subdivide where $k_i \geq 2\kappa$.

Figure 4.9: The effect of three different knot insertion strategies. The surfaces in this figure are shown with reflection lines and are all the result of subdivision at degree 7 from the non-uniform control mesh shown in Figure 6.1(a). The naïve strategy (a) results in folds: the same problem as shown in Figure 4.6. Strategies (b) and (c), described in §4.3.1, both remove this problem, but (c) gives fairer reflection lines where three knot intervals of different sizes interact in the lower-left corner.

face receives information on the parametric size of adjacent faces when constructing local knot vectors. For meshes containing vertices of higher valency, however, Figure 4.8 shows that this is not true, and so we need to account for knot intervals that are close in the mesh but distant in size. One way to do this is to avoid inserting a knot into smaller intervals altogether. If a subdivision step never inserts a knot into a small knot interval while there are large knot intervals in the mesh, then there is no possibility that any individual face will suffer from being isolated from a large knot interval in an adjacent face. This first phase of the knot insertion strategy therefore uses early steps to subdivide only large knot intervals, leaving smaller intervals unmodified. A subdivision step replaces any subdivided knot interval with two smaller intervals, and so it is possible, by this process, to make all knot intervals approximately the same size. See Figure 4.10(b) for an example.

To make this precise, let the set of knot intervals be $\boldsymbol{k} = \{k_i\}$. A natural strategy then inserts a new knot in the centre of only those knot intervals k_i where $k_i > \frac{1}{2}\max(\boldsymbol{k})$. This brings a unified treatment to multiple knots, as at any given step, knot intervals $k_i \leq \frac{1}{2}\max(\boldsymbol{k})$ are considered 'too small' to be subdivided. A multiple knot, given by $k_i = 0$ for some i, is then the limiting case that is considered too small at *every* subdivision step, no matter how small the value of $\max(\boldsymbol{k})$. Figure 4.9(b) shows a surface created from a non-uniform control mesh by following this strategy.

If, instead, we are willing to treat multiple knots as a special case, then I find that a subtly different strategy gives slightly fairer surfaces. We can evaluate the fairness of a surface using curvature variation, as visualised by the relative spacing of reflection lines drawn on the surface. Using the definition of an *artifact* given by Sabin and Barthe [77], we would like to remove features of the surface which occur at a higher frequency than the edges in the control mesh, as these features cannot be controlled by altering the position of the control points. The alternative strategy, which I describe below, results in the surface shown in Figure 4.9(c), removing the sharp change of curvature visible in the lower-left corner of Figure 4.9(b).

Separating zero knot intervals from others may be a reasonable concession, as multiple knots

reduce continuity and are therefore deliberately introduced to achieve particular results: they do not arise accidentally. This alternative strategy starts by establishing the minimum non-zero interval $\kappa = \min(\{k_i : k_i > 0\})$. We want to ensure that

$$\max(\boldsymbol{k}) < 2\kappa \tag{4.1}$$

If this is not the case, we can use a limited subdivision step that inserts knots only into intervals which are 2κ or greater. Subdividing each interval at its midpoint gives a total of $\lfloor \log_2(\max(\boldsymbol{k})/\kappa) \rfloor$ subdivision steps to achieve the condition (4.1).

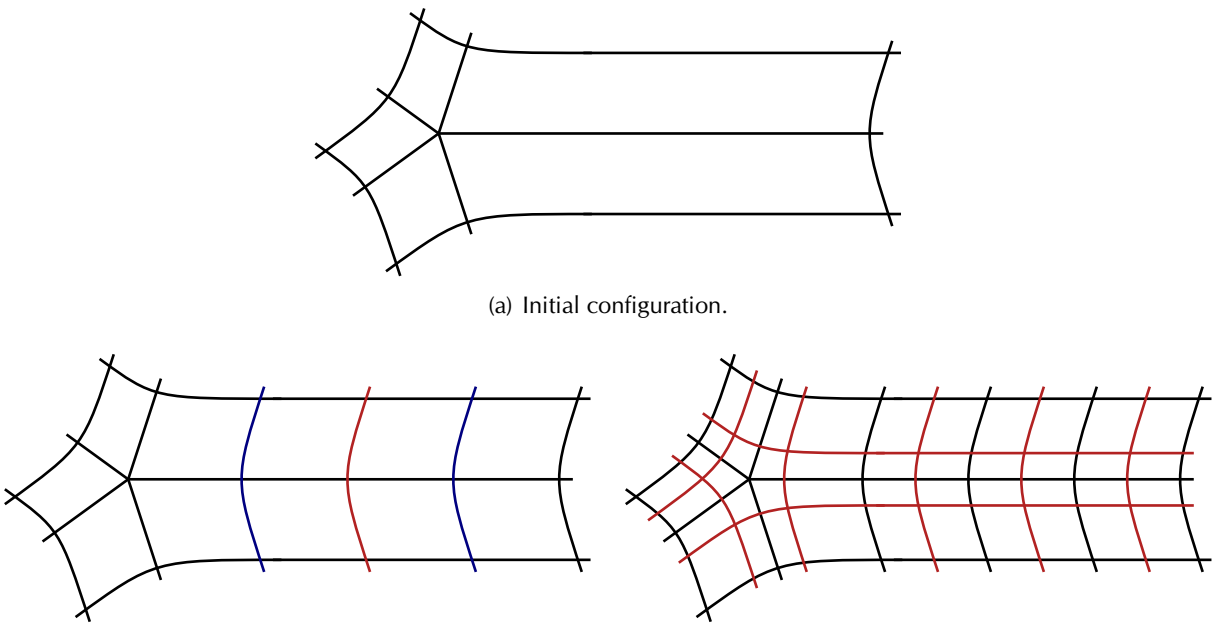
Comparing Figure 4.9(c) with 4.9(b), note that the condition (4.1) can give slightly fairer results than subdividing knot intervals where $k_i > \frac{1}{2}\max(\boldsymbol{k})$. However, both of the strategies in this section subdivide large knot intervals first. Therefore both methods remove the concave folds from Figure 4.9(a), which shows the poor surface which results from inserting a knot into the centre of every interval.

Creating uniform extraordinary regions 4.3.2

The problem shown in Figure 4.6 is macroscopic: it appears at the scale of the first subdivision step. The first phase of our knot insertion strategy, subdividing large intervals first, is able to avoid problems of this type. Once knot intervals are regularised to approximately the same size, however, we can turn our attention to the effect of extraordinary vertices at smaller scales. To make guarantees on the behaviour of these extraordinary regions, subdivision theory uses eigenanalysis on a local *subdivision matrix* (see §5.1). This analysis requires a *stationary* configuration: a subdivision matrix that is identical at every subdivision step, and the most natural way to achieve a stationary matrix is to create a region of the control mesh with uniform knot intervals. I therefore use this second phase of the knot insertion strategy to insert knots uniformly around extraordinary vertices, while subdividing other intervals at their midpoint. Creating uniform extraordinary regions in this way is crucial for the bounded-curvature properties I consider in Chapter 5, including the proof of C^1 continuity I discuss in §5.6.

To create uniform extraordinary regions, we need to surround every extraordinary vertex with at least one layer of evenly spaced knots. Once the first layer of uniform knots has been created, a further s subdivision steps will create a total of at least 2^s uniform layers. No matter how high the degree (and hence, how large the subdivision matrix), the limit surface around each extraordinary vertex will therefore always be determined by uniform subdivision rules, as long as we can create this first uniform layer. In the simplest case, we can achieve this in just one subdivision step: consider a single extraordinary vertex of valency n surrounded by regular mesh, and let the knot intervals surrounding the extraordinary vertex be k_1 to k_n. Then we can insert knots into each interval k_i to create a layer of uniform knots at a distance $\min(\{k_1, \ldots, k_n\})/2$ away from the extraordinary vertex. Figure 4.10(c), for example, shows a single subdivision step which introduces a layer of uniform knots in this way.

In the general case, where there are multiple extraordinary vertices in an arbitrary configuration, we may need two subdivision steps before every extraordinary vertex is surrounded by the first uniform layer. Figure 4.11 shows that the uniform knot insertion for one extraordinary vertex may impact on another. In fact, wherever extraordinary vertices appear on the same side of a strip of quadrilateral faces, they must insert knots at the same distance away in order to achieve



(a) Initial configuration.

(b) After subdividing large knot intervals first (§4.3.1). The knot shown in red is inserted in the first subdivision step; those shown in blue are inserted in the second.

(c) The knots inserted in the subdivision step which creates uniform extraordinary regions (§4.3.2).

Figure 4.10: The knot insertion strategy in §4.3 for an example non-uniform configuration. The size of knot intervals is represented geometrically by using the length of associated edges.

uniform knot spacing. This is a result of the decision (in §4.1) to maintain the NURBS constraint that opposite edges of a face share the same knot interval. Each knot interval can have at most one knot inserted per subdivision step and, like NURBS, knots are inserted along the entire length of the interval. Creating uniform knot spacing for one extraordinary vertex may, therefore, have an impact on the knot spacing surrounding another extraordinary vertex which is distant in the control mesh.

To address these mutual dependencies, we must find a way of passing information between extraordinary vertices which appear on the same side of a strip of quadrilateral faces. For another way of formulating this relationship, consider n rays of edges emanating from an extraordinary vertex of valency n. Let each ray pass over four-valent vertices by always taking the edge directly opposite the vertex. These rays terminate if they reach a boundary or another extraordinary vertex, but otherwise they continue indefinitely (see Figure 4.12 for an example). Any extraordinary vertices which are connected by such an emanating ray are mutually constrained to achieve uniform knot spacing at the same distance away. We can define a set of equivalence classes for extraordinary vertices connected by emanating rays, and represent the classes using a disjoint set data structure [28] with associated *find* and *union* operations. We already require data structures which represent each of the knot intervals in the mesh, and these objects are ideally positioned to merge any equivalence classes which are connected by an emanating ray. This results in the algorithm shown in Figure 4.13, which calculates the membership of the classes in a single pass through points in the mesh. The equivalence classes are unaffected by subdivision, so they need only be calculated once, and we only have to consider vertices that appear in the input control mesh.

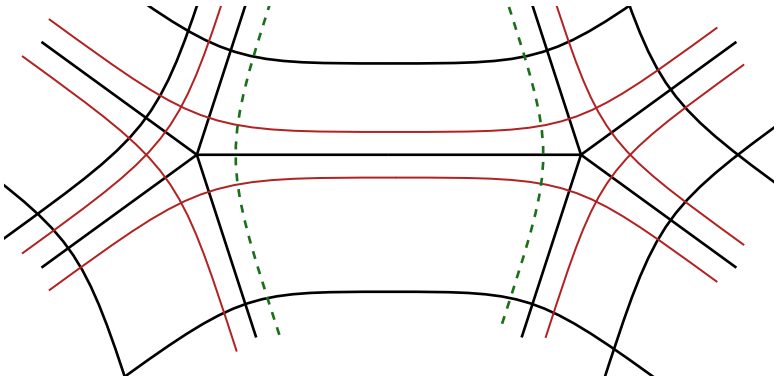


Figure 4.11: The interaction of multiple extraordinary vertices while creating uniform regions. In this figure, knot interval size is symbolised geometrically; black edges indicate original knots, and we insert the red knots in the next subdivision step. Note that the uniform spacing for the extraordinary vertex on the right is influenced by the intervals surrounding the vertex on the left. There is also a conflict in the central knot interval, as the knots shown using dashed green lines cannot both be inserted in the first subdivision step.

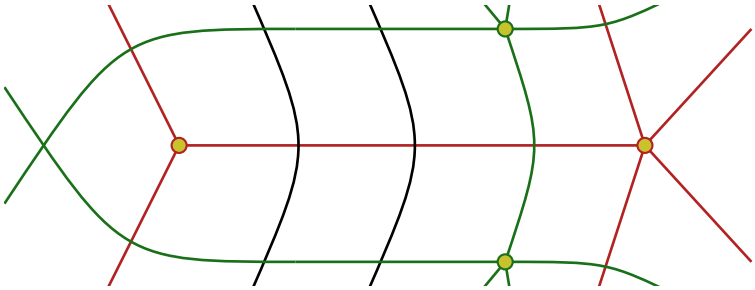


Figure 4.12: An equivalence class contains extraordinary vertices connected by emanating rays. For this example, the two extraordinary vertices marked ● (with emanating rays drawn in red) are in the same equivalence class, as are the two vertices marked ● (with emanating rays drawn in green).

Once calculated, the equivalence classes can be used to insert knots at a distance which achieves uniform spacing for all members of the class. However, Figure 4.11 shows that two or more extraordinary vertices, on *opposite* sides of a knot interval, might make conflicting requests for where a knot should be inserted. In this case, I use a first subdivision step to insert a knot between the two positions. As the knot interval is then split, there can be no conflict in the subsequent subdivision steps.

Uniform knot spacing around extraordinary vertices is thus guaranteed in at most two steps. In summary, the second phase of our knot insertion strategy is:

- for each equivalence class, establish the minimum of the knot intervals that surround extraordinary vertices within it,
- for each extraordinary vertex, request that knots be inserted into surrounding intervals at half of the minimum interval for the class,
- for each knot interval, subdivide at the midpoint, or at the mean of the requested positions, if there are insertion requests.

This algorithm is given as pseudocode in lines 7 to 16 of Figure 4.14.

```
foreach extraordinary vertex v do
    equivalence class e_v ← { v }

foreach extraordinary vertex v do
    foreach adjacent knot interval k do
        if the side of k where v appears is associated with an equivalence class f then
            Union(e_v, f)
        else
            Associate the side of k where v appears with e_v
```

Figure 4.13: Algorithm to assign each extraordinary vertex to an equivalence class.

```
initialize equivalence classes using the algorithm shown in Figure 4.13
foreach subdivision step do
    // Knot insertion strategy
    if max(k) ≥ 2κ then
        foreach knot interval k where k ≥ 2κ do
            subdivide k at its midpoint
    else
        foreach equivalence class e_i do
            κ_i ← min(knot intervals surrounding points in e_i)
        foreach extraordinary vertex v in equivalence class e_i do
            foreach knot interval surrounding v do
                make insertion request at a distance (1/2)κ_i from v
        foreach knot interval k do
            if k has insertion requests then
                subdivide k at mean of requested positions
            else
                subdivide k at its midpoint

    // Subdivide mesh based on inserted knots
    foreach quadrilateral face f with knot intervals k_i and k_j do
        if k_i and k_j both subdivided then
            compute refine stage for new point within f
        if k_i or k_j is subdivided then
            make contributions for new points on edge of f
    for σ ← 1 to degree step 2 do
        foreach quadrilateral face f do
            make contributions for smoothing stage σ within f
    foreach 3-valent vertex v do
        compute final position of v
```

Figure 4.14: Pseudocode for non-uniform general-degree subdivision. The final phase of subdivision (modifying the position of 3-valent vertices) is described in §5.4.

We cannot use the method described here if there is a multiple knot (i.e. a zero knot interval) adjacent to an extraordinary vertex. Such a configuration makes it impossible to create uniform knot spacing using knot insertion, so the guarantees on surface continuity provided in Chapter 5 cannot apply in this case. Where uniform spacing is achievable, however, recall that additional subdivision steps insert knots into the centres of the surrounding intervals. This creates a growing region of uniformity around each extraordinary vertex, so the behaviour of the limit surface at these points is determined by uniform rules. Therefore Chapter 5 (which modifies the uniform case for bounded curvature), and in particular §5.6 (which discusses a proof that the uniform rules create C^1 surfaces), applies to any non-uniform configuration with non-zero knot intervals adjacent to extraordinary vertices.

Summary and discussion 4.4

Figure 4.14 summarises the generalisation described in this chapter, and the knot insertion strategy described in §4.3. By moving away from inserting a knot into every existing interval, the knot insertion strategy allows non-uniform behaviour away from extraordinary vertices, yet takes advantage of the well-understood and higher-quality surfaces that result from uniform extraordinary regions. This is the first subdivision scheme that attempts to build such a transition, and it is clear that the strategy presented here improves significantly on the naïve approach of subdividing every non-zero knot interval (see Figures 4.6, 4.7 and 4.9).

However, inserting knots in a way that depends on the size of knot intervals has a notable disadvantage: the surface is no longer a continuous function of its knot specification. In particular, for the strategy in §4.3, a single knot interval that steadily increases in size may give a discontinuous deformation of the surface whenever that size crosses a power of 2 as a multiple of the other knot intervals. NURBS surfaces, by contrast, are a continuous function of their knot specification. Therefore if the growing knot interval has no influence over extraordinary regions, then the surface will still deform continuously. Around extraordinary vertices, however, the surface is dependent on the sequence of inserted knots (see §4.3 and further discussion in §6.3). It therefore no longer holds that the surface is a continuous function of its knot specification, despite the fact that this property holds for both NURBS surfaces and earlier non-uniform subdivision schemes [51, 89].

This chapter recommends a knot insertion strategy that gives the best results in the experiments I have conducted. However, Figure 4.9 shows that three different generalisations of NURBS knot insertion can give three different limit surfaces from the same control mesh. This raises the question of how we should choose which surface represents the best result. The definition of an artifact given by Sabin and Barthe [77] is one way to precisely formulate the set of surface features we want to avoid: those which appear at a higher frequency than the control mesh, as features of this frequency cannot be removed by modifying the position of the control points. There is also strong evidence that minimising curvature variation creates appealing surfaces (see Moreton [50], for example). Ultimately, however, choosing which of the surfaces in Figure 4.9 is most preferable is an aesthetic judgement. Farin and Sapidis [27] recognise that there is no perfect mathematical formulation of this quality. They quote Pierre Bézier, who said:

> "From an industrial viewpoint, a curve or a surface is nice if and only if it looks nice to the chairman of the board, the sales manager, the head stylist, and the prospective customer."

An accurate and objective measure of surface fairness is difficult, if not impossible, to achieve. Even for researchers who hope to provide a qualitative measure of the aesthetic quality of a surface, such as Joshi [36], this is a current area of research in its own right. I therefore do not claim to have found the optimum knot insertion strategy, and it is possible that ideas such as the strategy shown in Figure 4.9(b), which I believe gives inferior results, may still be worth investigating in future work.

There are also areas where the knot insertion strategy that I have recommended could be improved. A natural development would be to make the condition (4.1) local rather than global. A knot interval would then only be considered too large if there was an interval of half the size sufficiently nearby (where this distance depends on degree). This modification would allow normal midpoint subdivision for knot intervals which increase gradually across a surface, without compromising continuity around extraordinary vertices. Other disadvantages include the observation, in §4.3.2, that we can make no guarantees on surface continuity if zero knot intervals lie adjacent to extraordinary vertices. Extraordinary regions are affected by the sequence of inserted knots, so we are also unable to insert an arbitrary knot while maintaining an invariant limit surface. None of these limitations affect compatibility with NURBS, but they allow room for further analysis and improvement.

Despite these shortcomings, the outcome of this chapter is that almost all non-uniform configurations are able to transition to a uniform configuration around extraordinary vertices, and do so without adversely affecting surface smoothness. Any improvements to the uniform case therefore benefit most non-uniform surfaces as well. Chapter 5 examines this uniform case in greater detail, building on top of the generalisation in this chapter to guarantee C^1 continuity, and the best possible second-order behaviour for a subdivision scheme of this type.

Bounded curvature 5

This chapter presents research that has also been published in the following papers:

T. J. Cashman, U. H. Augsdörfer, N. A. Dodgson and M. A. Sabin. NURBS with Extraordinary Points: High-degree, Non-uniform, Rational Subdivision Schemes. *ACM Transactions on Graphics*, 28(3):#46, 1–9, 2009.

U. H. Augsdörfer, T. J. Cashman, N. A. Dodgson and M. A. Sabin. Numerical Checking of C^1 for Arbitrary Degree Subdivision Schemes based on Quadrilateral Meshes. In *13th IMA Conference on the Mathematics of Surfaces*, volume 5654 of *Lecture Notes in Computer Science*, pages 45–54. Springer, 2009.

Subdivision schemes based on tensor-product B-spline surfaces benefit from many of the favourable properties of B-splines. Those which generalise degree d B-splines, for example, have $d-1$ continuous derivatives almost everywhere. The only points which do not automatically inherit the continuity of the regular case are the singularities in the surface. For the odd degree schemes described in Chapter 4, each singularity is the limit of an extraordinary vertex through the infinite sequence of subdivision steps. Here surface continuity is harder to both achieve and analyse, and the popular subdivision schemes (including Catmull-Clark [10] and Loop [46]) are only C^1 at these isolated points.

Research to understand the singularities of subdivision surfaces began in one of the earliest papers in the field: Doo and Sabin [21] gave necessary conditions for both C^1 and C^2 continuity. However, the theory required for a thorough analysis is far from trivial, and it took another twenty years before the continuity of subdivision schemes was fully understood. This chapter starts by giving an overview of the mathematical tools that have been developed in this area, as well as some of the ways that they have been employed to improve the smoothness of subdivision surfaces in extraordinary regions. I then apply these methods to the NURBS generalisation discussed in Chapter 4, to create arbitrary-degree subdivision surfaces with bounded curvature. This class of surfaces has not appeared before, even in the uniform case, but the knot insertion strategy described in §4.3 means that almost all *non-uniform* configurations can also benefit from the modifications I describe here.

Continuity of subdivision surfaces 5.1

As in §3.1, we can write the linear map computed by a subdivision step as a matrix. If we consider the action of a subdivision step on the whole control mesh, then this matrix is taller than it is wide, as each step increases the density of the mesh. However, the subdivision rules compute only local affine combinations, so at each step we can consider a fixed number of vertices around an extraordinary vertex to obtain a square subdivision matrix $\boldsymbol{S}$. If a subdivision scheme

is *stationary*, then $\boldsymbol{S}$ is constant for each subdivision step. For an initial control mesh $\boldsymbol{Q}$, the singularity in the limit surface is then given by $\boldsymbol{S}^\infty \boldsymbol{Q}$, and Doo and Sabin [21] observed that we can infer properties of $\boldsymbol{S}^\infty$ using a diagonalisation of $\boldsymbol{S}$. If a subdivision scheme is also *uniform* then the same subdivision rules apply in every part of the mesh, and around an extraordinary vertex of valency n, the action of a subdivision step therefore has a rotational symmetry of order n. Doo and Sabin showed that in this case, a Discrete Fourier Transform (DFT) can simplify the analysis further, as $\hat{\boldsymbol{S}}$, the DFT of $\boldsymbol{S}$, is a block diagonal matrix. We can therefore decompose a subdivision step into blocks $\hat{\boldsymbol{S}}_\omega$, each of which acts on the ωth Fourier component of the input data, $\hat{\boldsymbol{Q}}_\omega$. Note that after a finite number of subdivision steps, the NURBS-compatible subdivision scheme described in Chapter 4 meets the above criteria: it is both stationary and uniform. This is a result of the knot insertion strategy which creates uniform extraordinary regions (§4.3.2).

With this view of a subdivision scheme, one property arises immediately from the fact that subdivision rules are affine combinations (i.e. take weighted means). Each row of $\boldsymbol{S}$ therefore sums to one, and so the vector of ones, $\mathbf{1}$, is an eigenvector with eigenvalue 1 (i.e. $\boldsymbol{S}\mathbf{1} = \mathbf{1}$). This eigencomponent appears in $\hat{\boldsymbol{S}}_0$, as $\mathbf{1}$ is a constant (zero frequency) vector, and so we say that it has *Fourier index* 0. An *eigenbasis* function is the limit of applying a subdivision scheme to an eigenvector $\boldsymbol{V}$, and clearly the eigenbasis function corresponding to this unit eigenvalue is the constant unit function.

To analyse a scheme any further, we must first place some constraints on $\boldsymbol{S}$. In §3.1, I established a set of properties that apply to *uniform* refine-and-smooth knot insertion, and that we could also retain in a *non-uniform* analogue. When examining the singularities of a subdivision scheme, we have a similar task: we want to find properties of the *regular* ($n = 4$) case, that we can retain for *extraordinary* vertices (arbitrary n). It is therefore useful to ask how diagonalisation and the DFT can help us to understand the regular case. If we consider $\boldsymbol{S}$ for $n = 4$ and degree cubic or higher, some important properties that arise from this analysis are:

- The unit eigenvalue is dominant, and the next largest eigenvalue is double. This corresponds to the fact that the space of bivariate linear functions has dimension two.

- The subdominant eigenvalue λ has Fourier index ± 1; the rotational symmetry which allowed us to use the DFT means that λ appears with equal value in both $\hat{\boldsymbol{S}}_1$ and $\hat{\boldsymbol{S}}_{-1}$.

- The space of bivariate quadratic functions has dimension three, and there are therefore three eigencomponents that form a corresponding basis. These eigencomponents capture quadratic properties of the limit surface at the singularity $\boldsymbol{S}^\infty \boldsymbol{Q}$.

- One of these eigencomponents has Fourier index 0. The associated eigenvalue, which we shall write as μ_0, is subdominant in $\hat{\boldsymbol{S}}_0$ (recall that the *dominant* eigenvalue in $\hat{\boldsymbol{S}}_0$ is the unit eigenvalue).

- The other two quadratic eigencomponents correspond to the dominant eigenvalues in $\hat{\boldsymbol{S}}_{\pm 2}$; we shall write their value as μ_2. As for λ, this eigenvalue is double as a result of rotational symmetry.

- All other eigenvalues are strictly less than the eigenvalues μ_0 and μ_2.

In this chapter, I consider only schemes with subdivision matrices which retain these six properties for arbitrary n. It is possible, with more care, to analyse subdivision schemes which do not retain these properties [54], but we do not need that level of generality here. It therefore simplifies the discussion to assume that these six properties hold.

For the specific case when $n = 4$, we can also find several properties which some or all of the schemes I consider in this chapter are *unable* to retain. For example, in the regular case the double subdominant eigenvalue λ is always equal to $\frac{1}{2}$. Furthermore, the subsubdominant eigenvalue is triple, as μ_0 and μ_2 are equal to a common value $\mu = \lambda^2 = \frac{1}{4}$. When $n = 4$, the eigenbasis function associated with μ_0 is a quadratic cup (an elliptic paraboloid), and the eigenbasis functions associated with μ_2 are orthogonal quadratic saddles (hyperbolic paraboloids). In summary, when $n = 4$ and degree is at least cubic, the eigenvalues of $\boldsymbol{S}$ in decreasing order are:

$$1,\ \overbrace{\tfrac{1}{2},\ \tfrac{1}{2}}^{\lambda},\ \overbrace{\tfrac{1}{4},\ \tfrac{1}{4},\ \tfrac{1}{4}}^{\mu},\ \ldots \text{ other values} < \tfrac{1}{4} \tag{5.1}$$

When degree is *quadratic*, the first six eigenvalues of $\boldsymbol{S}$ in the regular case are the same as in (5.1). However, quadratic B-splines are only C^1: they do not have a continuous second derivative, and as a result the μ eigenvalues are not dominant with respect to the remaining eigencomponents. The Doo-Sabin subdivision scheme [21] generalises biquadratic B-splines to arbitrary topology while maintaining these first six eigenvalues, and thereby achieves an optimal spectrum for arbitrary n. Doo and Sabin hypothesised that it might be possible to create a bicubic subdivision scheme that maintained exactly the spectrum (5.1). They also showed that the Catmull-Clark scheme [10] fails to do so, and instead has three different values for λ^2, μ_0 and μ_2 wherever $n \neq 4$. The result is that although the Catmull-Clark scheme has a continuous first derivative, in general the surface has a *divergent* second derivative at every singularity where $n > 4$: curvature grows without bound through the infinite sequence of subdivision steps.

Proving C^1 5.1.1

Ball and Storry [6] attempted to bring the spectrum of the Catmull-Clark subdivision matrices closer to the regular case, and identified [5] the importance of the subdominant eigenvectors for tangent-plane continuity. They coined the name *natural configuration* for the mesh created by taking co-ordinates from the two eigenvectors corresponding to λ, and showed that, in the limit, the configuration of points around a singularity will always be an affine transform of this natural configuration.

However, Ball and Storry, like Doo, Sabin [21] and Loop [46] before them, relied on properties of the *eigenvalues* to verify C^1 continuity, and neglected to prove conditions on the *eigenbasis* functions. Reif [69] showed that this meant the previous analyses were incomplete, and proposed a new set of criteria for proving not only tangent-plane continuity, but also *regularity*, a condition which guarantees that the surface is free of local self-intersections. He introduced the name *characteristic map* for the function from a mesh-based parameter space to $\mathbb{R}^2$ that evaluates the two eigenbasis functions corresponding to λ. This characteristic map is therefore the 'surface' that results from applying a subdivision scheme to Ball and Storry's natural configuration[14]. Reif then

[14]The natural configuration exists in a range of possible sizes, depending on how large a neighbourhood of points is

showed that the characteristic map can be used as a parametrisation of a subdivision surface near singularities, and concluded that a subdivision scheme is regular if

- the subdominant eigenvalue $\lambda < 1$,
- the subsubdominant eigenvalue is strictly less than λ, and also
- the characteristic map is regular and injective.

These conditions have become the standard measure of C^1 continuity for subdivision surfaces, as any scheme that fails to be regular would not be usable in practice.

Bounded curvature 5.1.2

Sabin [74] and Reif [70] independently showed that no modification of the Catmull-Clark subdivision rules, thus producing a different $\mathbf{S}$, would allow the Catmull-Clark scheme to create C^2 surfaces[15]. This was an important result, as the regular regions of Catmull-Clark surfaces are curvature continuous, and so Ball and Storry, among others, had hoped that modifying the rules might allow the singularities to hold the same level of continuity.

Although a modified subdivision matrix cannot allow the Catmull-Clark scheme to create non-degenerate C^2 surfaces, the second derivative can still exhibit a range of possible behaviours at singularities. Sabin et al. [81] summarise the most important options, including the situation where a subdivision matrix has the eigenvalues:

$$1,\ \lambda,\ \lambda,\ \overbrace{\lambda^2,\ \lambda^2,\ \lambda^2}^{\mu},\ \ldots \text{ other values} < \lambda^2 \tag{5.2}$$

That is, $\mu_0 = \mu_2 = \lambda^2$. This gives the subdivision scheme a property known as *bounded curvature*. It is a necessary condition for nontrivial curvature continuity [64] but it is not sufficient. Guaranteeing C^2 continuity, just as for C^1 continuity, requires analysis of the eigenbasis functions: in this case those corresponding to the subsubdominant eigenvalues. However, subdivision schemes with bounded curvature do preserve curvatures in both of the quadratic eigencomponents through subdivision. This avoids several undesirable outcomes:

- If $\mu < \lambda^2$, then the surface has a flat spot, as the quadratic components shrink faster than the square of the linear components.
- If $\mu > \lambda^2$, then the surface has divergent curvature, as the quadratic components shrink slower than the square of the linear components.
- If $\mu_0 > \mu_2$, then the surface has prescribed positive Gaussian curvature for almost all initial control meshes, as the hyperbolic quadratic components shrink faster than the elliptic component.

considered when constructing the subdivision matrix. The limit surface for the smallest possible natural configuration is just a single point: the singularity. For the natural configuration to give a non-degenerate characteristic map, we must therefore take one more ring of points around an extraordinary vertex than this minimum number.

[15] These results apply specifically to subdivision schemes that can hold the same range of quadratic shapes as the regular points of a surface, a property that Zorin [104] calls '2-flexibility'. As we shall see in §5.1.3, it *is* possible to create modifications of Catmull-Clark that are technically C^2, by creating a 'flat spot', of zero curvature, at each singularity. However, these surfaces have artifacts that are too severe for most practical purposes.

- If $\mu_2 > \mu_0$, then the surface has prescribed negative Gaussian curvature for almost all initial control meshes, as the elliptic quadratic component shrinks faster than the hyperbolic components.

Bounded-curvature schemes, by contrast, allow extraordinary regions to hold an arbitrary non-zero curvature, just as in regular regions. However, if the eigenbasis functions for the quadratic components are not, in fact, quadratic surfaces (as is the case for any modifications to the Catmull-Clark subdivision rules where $n \neq 4$), then curvature at the singularity $\boldsymbol{S}^\infty \boldsymbol{Q}$ is undefined within bounds that depend on both $\boldsymbol{S}$ and $\boldsymbol{Q}$ [56].

Karčiauskas et al. [40] show that the unmodified Catmull-Clark scheme has $\lambda^2 < \mu_0 < \mu_2$ at valencies greater than four, leading to a limit surface which is always hyperbolic, irrespective of the control mesh $\boldsymbol{Q}$. Several researchers have addressed this shortcoming with bounded-curvature variants of Catmull-Clark; Sabin [74] was the first to do so, and several more bounded-curvature schemes are discussed in §5.1.4. The modifications described in this chapter also lead to a bounded-curvature variant of Catmull-Clark in the special case where degree is 3 and knot vectors are uniform. Although this dissertation is only concerned with subdivision schemes generalising tensor-product B-splines, the above analysis also applies to triangular control meshes. Holt [34] and Loop [45] both presented bounded-curvature schemes for this type of refinement.

C^2 schemes 5.1.3

Reif's proof [70], showing that low-degree stationary subdivision schemes cannot produce surfaces with nontrivial C^2 continuity, motivated schemes which achieve curvature continuity by dint of either degenerate surfaces, or more complicated constructions. Prautzsch and Umlauf [65] presented an example of the former. They modified Catmull-Clark rules to produce a subdivision matrix where $\mu_0 < \lambda^2$ and $\mu_2 < \lambda^2$. The quadratic components therefore decay fast enough that they are absent from the singularity, and the resulting flat surface is trivially curvature continuous. However, flat spots at each singularity make this solution unacceptable for many applications. This includes applications where reflections are important, such as car bodies, since flat spots will typically make it impossible to create a smooth, even flow of reflection lines.

Another way of circumventing the barrier to C^2 continuity is to use higher-degree B-splines. Both Prautzsch [61] and Reif [71] presented constructions with C^k singularities, for arbitrary k, by using B-spline patches of degree $2k+2$. This is the lowest degree at which it is possible for a stationary, uniform subdivision scheme to have a C^k limit surface, but for C^2 continuity this already requires bisextic patches, and the minimum degree increases rapidly with k.

Zulti et al. [107] were able to achieve C^2 continuity in a theoretical case where there is only one extraordinary vertex in the control mesh. This is an interesting theoretical result, but not useful in practice. Levin [44] and Zorin [104] modified the Catmull-Clark and Loop schemes, respectively, to create C^2 surfaces by smoothly blending the subdivision surface with another, best-fit C^2 surface. Karčiauskas and Peters [37] used a similar idea, except instead of blending the fitted surface directly, they used it as a guide to direct a subdivision surface into a fairer shape. They called the resulting algorithm *guided subdivision*. Myles [52] was also able to achieve C^2 low-degree subdivision surfaces for a particular 'polar' patch layout, by using subdivision rules which increase the valency of a polar vertex at every subdivision step.

One of the main objections to these C^2 constructions is the additional complexity which is required to implement and use them. Several of the ideas above, such as Levin's blended surfaces [44], could also be applied to NURBS-compatible subdivision schemes once uniform knot spacing has been created around extraordinary vertices. In this dissertation, however, I have chosen to remain in the simpler setting of stationary subdivision matrices.

Tuning 5.1.4

Previous work has also investigated to what extent subdivision surfaces can be improved whilst remaining within the constraints of stationary subdivision. Barthe and Kobbelt [7] treated subdivision weights as degrees of freedom in a nonlinear optimisation. They manipulated the subdivision matrix by minimising an energy which favoured desirable properties for the matrix eigenstructure, including bounded curvature. They coined the name *tuning* for this kind of subdivision rule optimisation.

Karčiauskas et al. [40] analysed subdivision schemes in terms of *shape charts*, which test a set of representative meshes for *hybrid shapes*. These are subdivision surfaces which contain both positive and negative Gaussian curvature in every infinitesimal region of a singularity. Augsdörfer et al. [2] and Ginkel and Umlauf [29] used a variant of these charts to tune schemes for bounded curvature: Augsdörfer et al. sought to minimise the range of the curvature bounds, while Ginkel and Umlauf tried to eliminate hybrid shapes by minimising cases where the bounded range includes both positive and negative curvatures.

There are different approaches to tuning, which are separated largely by how many subdivision weights a tuning procedure is permitted to alter. Barthe and Kobbelt modified every weight in affine combinations where a new vertex is influenced by an extraordinary vertex, but Augsdörfer et al. used the restricted setup known as *mask tuning*. Here only the influence of an extraordinary vertex is modified: the weights on contributions from all regular vertices stay the same. This requires affine combinations to be normalised, so that the weights in each combination still sum to one. However, mask tuning has the advantage that early subdivision steps, where extraordinary vertices may be close in the control mesh, can be handled using a single implementation without special cases. A vertex which receives contributions from two extraordinary vertices simply uses a different denominator to normalise the affine combination.

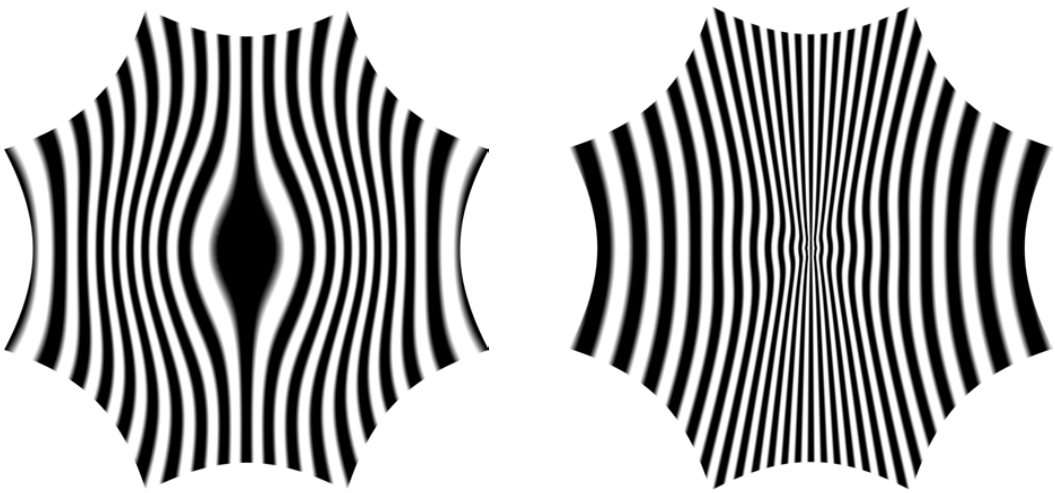
Existing high-degree schemes 5.2

This chapter provides the first bounded-curvature subdivision surfaces that generalise arbitrary-degree B-splines. Existing arbitrary-degree subdivision surfaces [62, 63, 91, 93, 103, 105], which I discussed in §2.2.4, all have either zero or unbounded curvature at singularities. These schemes generalise the Lane-Riesenfeld midpoint smoothing operator, and mostly do so by using smoothing stages that replace every face with its centroid. We can use the tools described in §5.1 to analyse the second-order behaviour of these centroid-averaging schemes: Table 5.1 gives the ratios μ_0/λ^2 and μ_2/λ^2 for the first five odd degrees at a selection of valencies. For bounded curvature, both ratios must be equal to 1, but this only occurs in the regular case, where $n = 4$. In the irregular case, the table illustrates that

- the curvatures are further from bounded as valency increases,

	μ_0/λ^2					μ_2/λ^2				
n	$d=3$	$d=5$	$d=7$	$d=9$	$d=11$	$d=3$	$d=5$	$d=7$	$d=9$	$d=11$
3	1.487	1.550	1.567	1.574	1.578	0.906	0.974	0.987	0.991	0.994
4	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
5	0.827	0.797	0.785	0.779	0.775	1.124	1.060	1.040	1.030	1.024
6	0.744	0.699	0.681	0.671	0.665	1.220	1.123	1.087	1.070	1.058
7	0.698	0.644	0.622	0.610	0.602	1.289	1.177	1.133	1.108	1.093
8	0.669	0.611	0.586	0.572	0.563	1.339	1.221	1.172	1.144	1.126

Table 5.1: The ratios μ_0/λ^2 and μ_2/λ^2 for centroid-averaging schemes.



(a) Cup configuration. (b) Saddle configuration.

Figure 5.1: Eigenbasis functions for the centroid-averaging schemes at degree 9 and valency 8. The functions are rendered as surfaces and viewed using *reflection lines*. Reflection lines are a function of the surface normal and therefore 'amplify' second-order surface properties into first-order effects. Here $\lambda = 0.661$, which gives $\lambda^2 = 0.437$. $\mu_0 = 0.25 < \lambda^2$, so the eigenbasis function generalising a quadratic cup has a flat spot at the singularity. $\mu_2 = 0.5 > \lambda^2$, so the eigenbasis functions generalising quadratic saddles have unbounded curvature.

- curvatures in the cup (μ_0) component diverge for valency three and tend to zero for valencies greater than four,
- curvatures in the saddle (μ_2) components tend to zero for valency three and diverge for valencies greater than four,
- as degree increases, curvatures in the cup component grow or shrink at a faster rate (second-order behaviour worsens), but curvatures in the saddle components do so at a slower rate (second-order behaviour improves).

Figure 5.1 gives the eigenbasis functions corresponding to μ_0 and μ_2 in the particular case of degree nine and valency eight. This figure shows the undesirable flat spot for the cup component and unbounded curvature for the saddles.

Bounded curvature for saddle components 5.3

We are now ready to return to the main theme of this chapter: modifying the NURBS-compatible subdivision schemes described in Chapter 4 to hold the bounded-curvature property. Like Augsdörfer et al. [2], we would like to limit the modifications we make to mask tuning, the technique described in §5.1.4, as this allows adjacent extraordinary vertices to be handled in a unified framework. Our goal is therefore to modify the influence of an extraordinary vertex on surrounding vertices in order to achieve the eigenspectrum (5.2).

The refine-and-smooth factorisation described in Chapter 3 uses local affine combinations, so an extraordinary vertex contributes to vertices in only three different positions: to itself, to vertices which are edge-connected, and to those which are face-connected. Mask tuning therefore allows for at least three modified weights for these three different types of contribution. In addition, where degree is greater than three, we could modify the weights of each smoothing stage separately, but we have only two bounded-curvature equations to satisfy, so this is an unnecessary number of degrees of freedom. To keep the problem manageable, here I have chosen to further limit the mask tuning to just three parameters, no matter how high the degree.

Previous work on mask tuning [2, 79] uses constant parameters that set the weight of contributions from an extraordinary vertex. However the knot insertion algorithm described in §3.3 uses weights that change between smoothing stages and with degree, even for uniform knot intervals. For degree seven and valency four, for example, the weight of a vertex on its successor is $\left(\frac{1}{6}\right)^2$ in the first smoothing stage and $\left(\frac{3}{4}\right)^2$ in the last. If the tuning for arbitrary n is to generalise the four-valent case, we therefore cannot use a constant parameter to set the weight of contributions from an extraordinary vertex directly: for the degree seven example, such a parameter would have to be equal to both $\left(\frac{1}{6}\right)^2$ and $\left(\frac{3}{4}\right)^2$ at the same time. Instead, therefore, I use constant *multipliers* to modify the regular weight of the extraordinary vertex in a given affine combination. This formulation generalises the four-valent case, since the regular weights are restored by setting every multiplier equal to one. For degree d and valency n, the multipliers are:

- α_n^d, for contributions to the extraordinary vertex itself,
- β_n^d, for contributions to edge-connected vertices,
- γ_n^d, for contributions to face-connected vertices.

I use the same multipliers for both the refine and smoothing stages. In the refine stage, this means that the weight of an extraordinary vertex is multiplied by β_n^d for contributions to new vertices on adjacent edges (e.g. Figure 4.5), and by γ_n^d for contributions to new vertices in adjacent faces. Note that $\alpha_4^d = \beta_4^d = \gamma_4^d = 1$ for valency four at any degree d (i.e. the mask tuning does not modify the four-valent weights), as the regular uniform case is already curvature-continuous.

We are interested in the Fourier blocks $\hat{S}_0$, for μ_0, $\hat{S}_1$, for λ, and $\hat{S}_2$, for μ_2. However, the extraordinary vertex appears explicitly only in $\hat{S}_0$: where $\omega > 0$, symmetry dictates that the extraordinary vertex is always implicitly located at the origin. Therefore α has no effect on the blocks $\hat{S}_1$ and $\hat{S}_2$, and the condition $\lambda^2 = \mu_2$ must be achieved using only β and γ. This condition does not uniquely determine β and γ, however: Figure 5.2 shows a range of solutions which all satisfy $\lambda^2 = \mu_2$. We therefore need a way of resolving this additional degree of freedom.

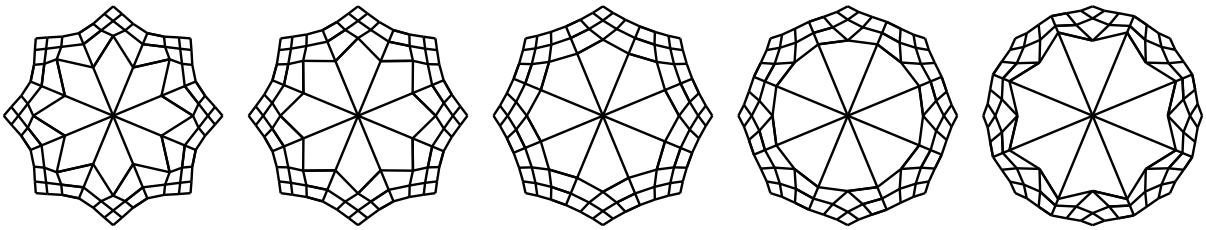


Figure 5.2: Five different bounded-curvature settings for β_8^5 and γ_8^5. This figure shows natural configurations for each possibility; each one satisfies the saddle bounded-curvature condition $\lambda^2 = \mu_2$. As two multipliers are available to satisfy one equation, there is a remaining degree of freedom which controls the relative sizes of β and γ, changing the shape of the natural configuration.

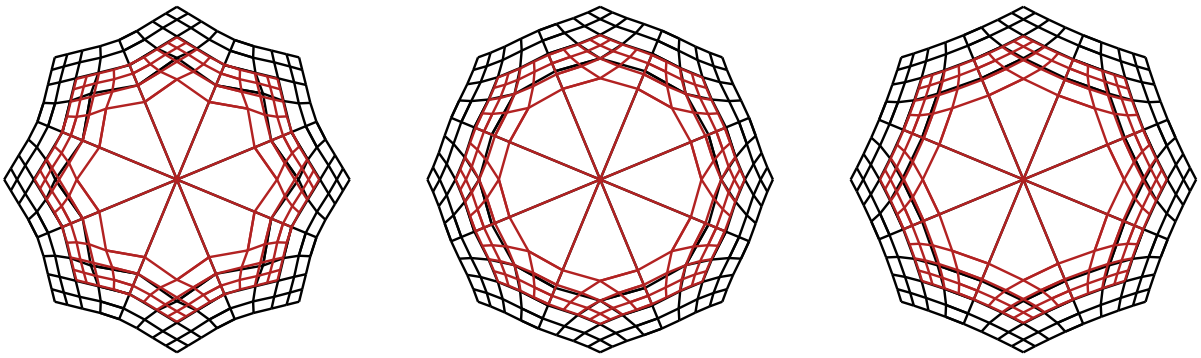
Augsdörfer et al. found that for bounded-curvature schemes, there is a correlation between the value of λ and the shape of a natural configuration, in the sense shown in Figure 5.2. They also found that the amount of variation in Gaussian curvature for the extraordinary regions of a surface is heavily dependent on this degree of freedom. I therefore expect NURBS-compatible subdivision schemes which produce good-quality, fair surfaces to have natural configurations with a similar shape to these earlier results. However, Augsdörfer et al. found solutions with minimum variation in Gaussian curvature through a complicated optimisation; I found that I could produce similar results with a much simpler heuristic.

The heuristic is based on the observation that, when $n = 4$, the natural configuration is exactly the result of sampling the characteristic map at evenly spaced values. Subdividing the natural configuration results in a denser uniform sampling, so the positions of existing vertices are unchanged. However, when $n \neq 4$, this property no longer holds: it is possible for a subdivision step on the natural configuration to modify the positions of existing vertices. As the natural configuration is an eigenvector, a subdivision step simply scales the mesh by the corresponding eigenvalue λ. Therefore the subdivided position of each point $\boldsymbol{p}_{(u,v)}$, with co-ordinates (u, v) measured from the extraordinary vertex, is given by $\lambda \boldsymbol{p}_{(2u,2v)}$. Let the *stability* of a natural configuration be a measure of how close these positions are to each other for each point $\boldsymbol{p}$. We could use a sum of squared distances, for example. Then I propose the following hypothesis, which was developed jointly with Malcolm Sabin:

> The stability of a subdivision scheme's natural configuration is inversely correlated with the amount of curvature variation in extraordinary regions. (5.3)

This hypothesis has not yet been thoroughly tested (see the discussion of future work in §6.5). However, resolving the degree of freedom shown in Figure 5.2 by selecting the *most stable* natural configuration does lead to solutions which are close to those found by Augsdörfer et al., in both the value of λ and the shape of the natural configuration. This is evidence in support of the hypothesis (5.3). Using stability as a heuristic also produces surfaces which are good enough for our purposes as a demonstration of NURBS-compatible subdivision; the resulting bounded-curvature schemes certainly have much less curvature variation than the Catmull-Clark scheme, for example (see Figure 5.10).

We have only one degree of freedom with which to choose the most stable natural configuration, so there is a conflict between stability for the vertices lying along edges, and those lying diagonally across faces. Choosing to make edge-connected vertices stable is equivalent to setting β from the



(a) Stable along edges, but not diagonally across faces.

(b) Stable diagonally across faces, but not along edges.

(c) The equal-shape condition, as an approximation to stability.

Figure 5.3: Finding an approximation to a stable natural configuration. We have only one degree of freedom to make the natural configuration as stable as possible, so we cannot make all the 1-ring vertices stable. This figure shows natural configurations for degree 3 and valency 8, accompanied by subdivided (i.e. scaled by λ) configurations in red.

n	β_n^3	β_n^5	β_n^7	β_n^9
3	1.2560	1.1370	1.1204	1.1172
5	0.6499	0.7919	0.8152	0.8339
6	0.4364	0.6048	0.6417	0.6770
8	0.2321	0.3604	0.3979	0.4421
20	0.0343	0.0578	0.0664	0.0790

n	γ_n^3	γ_n^5	γ_n^7	γ_n^9
3	1.4012	1.1003	1.0758	1.0608
5	0.6224	0.7727	0.8228	0.8466
6	0.4115	0.5638	0.6420	0.6813
8	0.2164	0.3098	0.3870	0.4289
20	0.0317	0.0430	0.0605	0.0695

Table 5.2: β_n^d and γ_n^d at a selection of valencies and with d ranging from 3 to 9.

stability condition, and then using γ to achieve $\lambda^2 = \mu_2$. Conversely, we could set γ to achieve stability for face-connected vertices, and then use β to satisfy the saddle bounded-curvature condition. However, these choices, as shown in Figure 5.3, are both detrimental to the stability of the natural configuration in the direction we ignore. Instead, therefore, I have chosen to set β and γ simultaneously to satisfy two conditions:

- that $\lambda^2 = \mu_2$, and
- that the quadrilateral in the first ring of the natural configuration is similar to the quadrilateral formed by the corresponding vertices in the second ring.

The result of these conditions is that the edge-connected and face-connected vertices in the natural configuration are not completely stable when $n \neq 4$, as a subdivision step alters their positions. However, any modification acts equally, scaling by the same value, across the whole of the first ring. I believe this is a good approximation to a stable natural configuration, which should therefore give surfaces with a small amount of curvature variation in extraordinary regions, assuming the hypothesis (5.3).

We are now ready to find β_n^d and γ_n^d for each odd d and $n \geq 3$. To do so, we must compute eigenvalues of a subdivision matrix whose size increases with degree. For high degrees, the matrices are so large that it is impractical to find an exact solution in symbolic form, even using

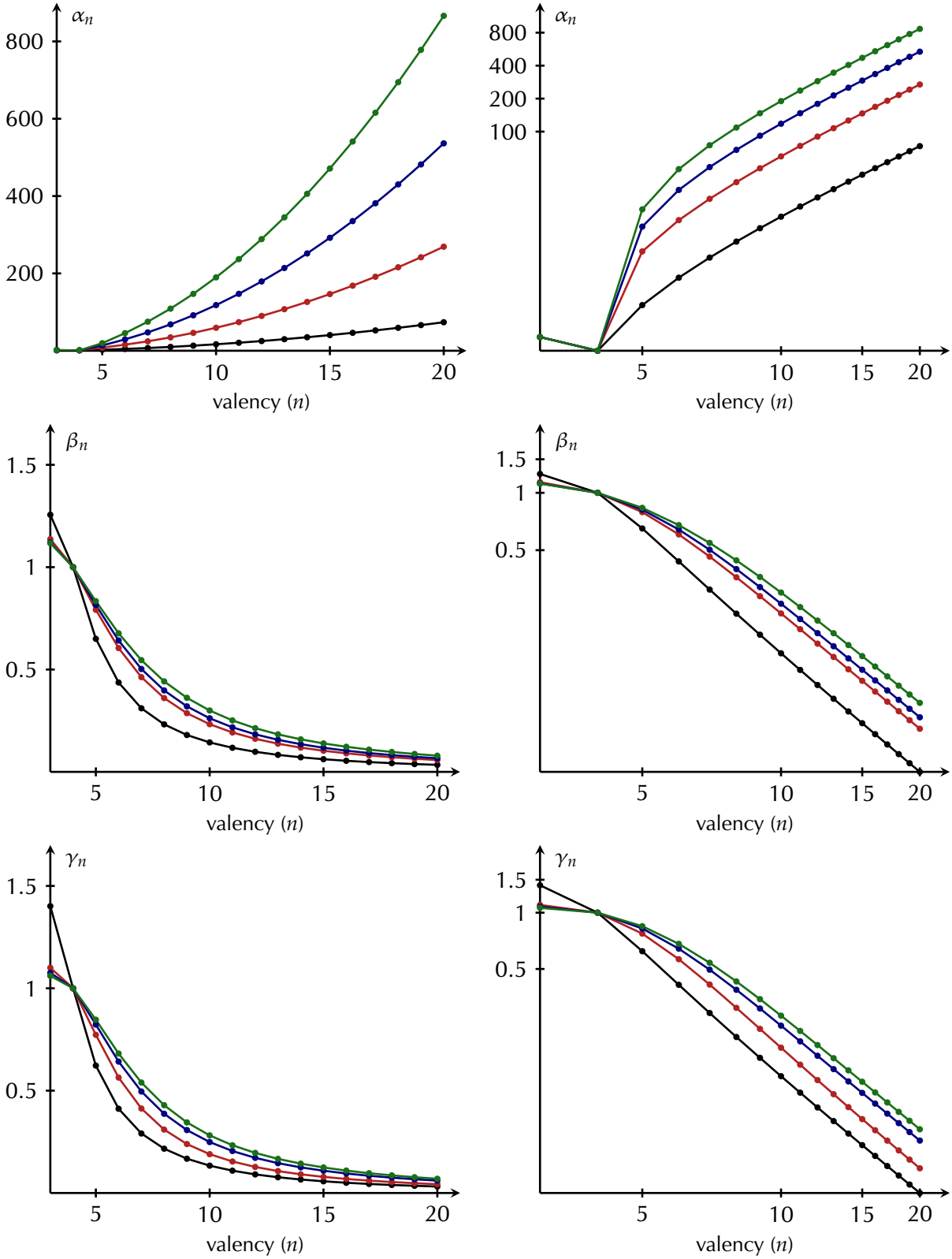


Figure 5.4: Bounded-curvature solutions for α_n^d, β_n^d and γ_n^d. Valency ranges from 3 to 20. The multipliers are plotted for degree, d, equal to 3 —•—, 5 —•—, 7 —•— and 9 —•— with a linear scale (left), and a logarithmic scale (right). Note that $\alpha_4^d = \beta_4^d = \gamma_4^d = 1$ by definition, as the regular case is curvature-continuous and hence already satisfies the bounded-curvature equations $\lambda^2 = \mu_0 = \mu_2$.

n	α_n^3	α_n^5	α_n^7	α_n^9
3	1.3333	1.3333	1.3333	1.3333
5	2.6079	8.0641	13.574	19.510
6	4.6412	15.560	29.347	45.446
8	9.9085	34.571	68.153	108.89
20	73.691	269.09	536.37	866.17

Table 5.3: α_n^d at a selection of valencies and with d ranging from 3 to 9. The values for α_3^d are explained in §5.4.2.

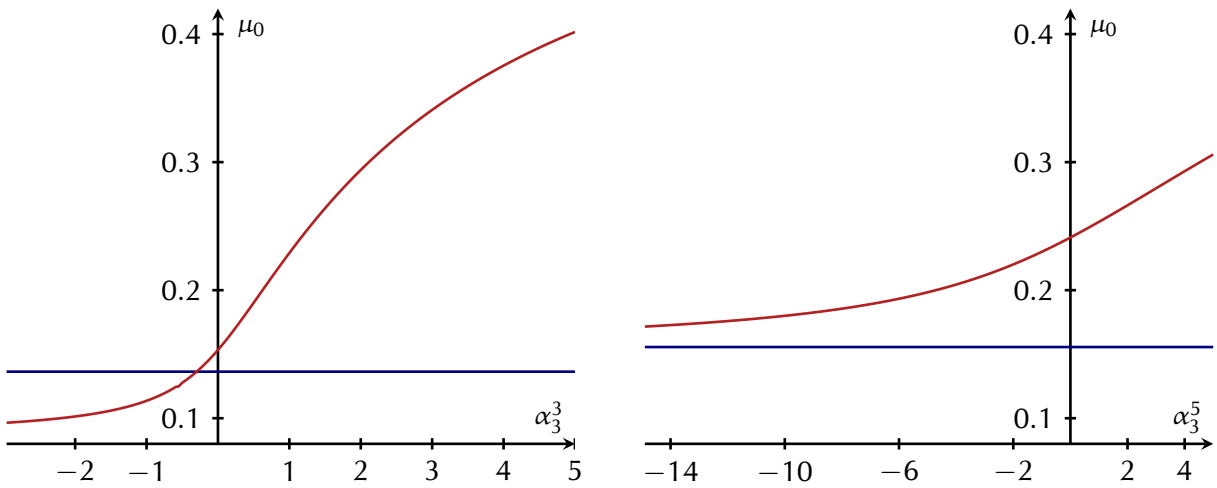
a DFT and considering each block separately. I therefore use a nonlinear equation solver on floating-point values. The equation solver does not form part of an implementation, as the values α_n^d, β_n^d and γ_n^d can be stored in a lookup table. This process therefore has no impact on the speed at which NURBS-compatible subdivision surfaces can be computed. Figure 5.4 and Table 5.2 show the resulting solutions for degrees 3 to 9 and valencies up to 20.

Bounded curvature for cup component 5.4

Once β and γ have been fixed in the way described in §5.3, we can use their values to compute $\hat{S}_0$ as a function of α. We already know λ, as $\hat{S}_1$ depends on only β and γ. We can therefore use an equation solver for floating-point values, just as in §5.3, to find the value for α that satisfies the bounded-curvature condition $\mu_0 = \lambda^2$. The use of floating-point arithmetic means that these solutions are only approximate, but I ensure that λ^2 differs from both μ_0 and μ_2 by no more than 10^{-12}, with subdivision matrices calculated to double precision and with floating-point eigenvalue routines. These values are close enough to be considered equal for any practical purpose. To see this, note that λ^{2s}, μ_0^s and μ_2^s are the scaling factors after s subdivision steps which, for bounded curvature, must be equal. If these values differ, then the quadratic eigencomponents decay at a different rate, leading to zero or unbounded curvature. However, $1 - \lambda^2 \gg 10^{-12}$, even for very large n, and so any disparity between the scaling factors grows many orders of magnitude slower, with s, than the rates at which the eigencomponents themselves shrink. The fact that the values are not exactly equal therefore has no measurable effect on the limit surface. The equation-solving procedure to find α_n^d works for any $n > 4$, and Figure 5.4 and Table 5.3 show some of the resulting values.

The logarithmic plots in Figure 5.4 (right) also show that an implementation would not necessarily need to store the values of α_n^d, β_n^d and γ_n^d for large n in order to be able to support high valencies. Although we do not have algebraic expressions for the multipliers, we know from simpler cases [79] that the values are rational functions of powers of n and $\cos(2\pi/n)$. As $n \to \infty$, $\cos(2\pi/n) \to 1$, and so for high valency we expect the values of the multipliers to be approximately polynomial in n. Figure 5.4 shows that this is the case: for high valency, β and γ are close to a constant multiple of n^{-2} and similarly, α is close to a constant multiple of n^2. The combination of a lookup table, for low n, and a polynomial model, for high n, would therefore allow an implementation to handle vertices of any valency.

When $n = 3$, however, using α to satisfy the condition $\lambda^2 = \mu_0$ gives solutions for α_3^d that are negative, as shown in Figure 5.5(a). Most approximating subdivision schemes hold the property



(a) Degree 3: the required value for α_3^3 is negative.

(b) Degree 5: there is no solution such that $\mu_0 = \lambda^2$.

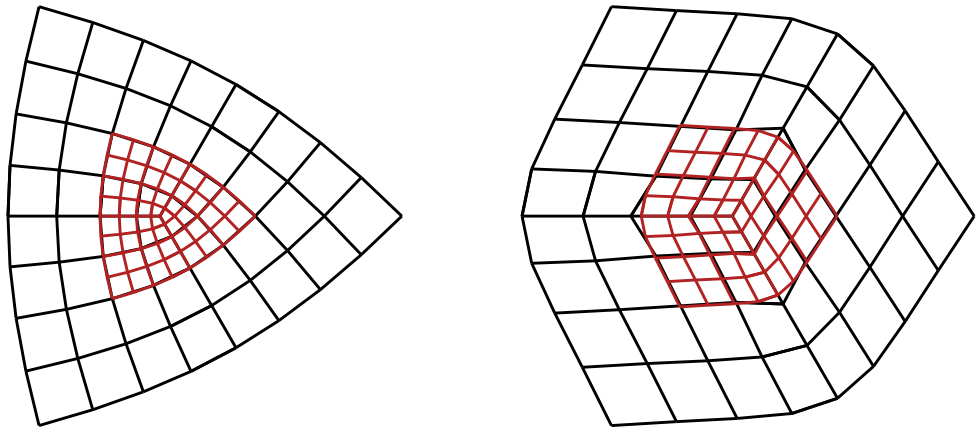
Figure 5.5: Problems that arise when trying to use α to satisfy $\lambda^2 = \mu_0$ for $n = 3$. These graphs show how μ_0 —— varies with α, as well as the target value λ^2 ——. For both graphs, α is plotted for a range whose minimum is the value which causes the denominator in an affine combination to become zero. Values for α which are less than this are not feasible, as normalising by a negative denominator creates an eigencomponent with an eigenvalue greater than 1.

that every affine combination uses non-negative weights, and the surface therefore lies inside the convex hull of the control points. This is useful for finding the solution to intersection queries, for example. A negative value for α_3^d does not necessarily invalidate this convex hull property: in the uniform case, the subdivision matrix $\boldsymbol{S}$ may still consist entirely of non-negative entries, in which case the convex hull property still holds for the subdivision step as a whole. However, we are interested in using these multipliers for both uniform and *non-uniform* configurations, and if a multiplier is negative then there may be a configuration of non-uniform knots that does give negative entries for $\boldsymbol{S}$. To guarantee the convex hull property for the general case, including non-uniform knot intervals, it is therefore important to use only non-negative multipliers.

In addition, Figure 5.5(b) shows that when $n = 3$, there are some degrees with no bounded-curvature solution at all. We therefore need to treat the valency three case separately. The remainder of this section summarises some approaches to solving this problem, starting with a collection of possible solutions that all have notable disadvantages. These negative results help to inform a list of requirements that any modification for valency three must fulfil. I conclude this section with a simple modification for valency three that allows for $\lambda^2 = \mu_0 = \mu_2$, granting bounded curvature, at all degrees and valencies. This modification also retains the convex hull property for degrees 3 to 13.

Negative results 5.4.1

In §5.3, I restricted mask tuning to use just three parameters, no matter how high the degree. However, as discussed above, this setup proves to be too constrained for a bounded-curvature solution when $n = 3$, so instead we could consider separate multipliers for each refine and smoothing stage. The additional freedom obtained in this way does allow us to find a bounded-curvature solution for degree five and valency three, which is the case shown in Figure 5.5(b)



(a) The natural configuration obtained using the same multipliers for the refine and smoothing stages.

(b) The natural configuration that results from modifying the refine stage to satisfy the condition $\lambda^2 = \mu_0$.

Figure 5.6: Modifying the refine stage to achieve bounded curvature for degree 5 and valency 3. Both of these natural configurations satisfy the equal shape condition shown in Figure 5.3(c), but the modifications to the refine stage, shown in (b), adversely affect the stability of vertices in the *second* ring of vertices around the extraordinary vertex.

to have no bounded-curvature solution using only three multipliers. However, there are several disadvantages to this modification:

- The bounded-curvature condition $\lambda^2 = \mu_0$ still requires negative values for α. This is true at degree three, for example, as the refine-and-smooth factorisation alternates, at odd degrees, between the actions shown in Figures 3.6(a) and 3.6(d) (see Figure 3.9 for an example of this structure). This means that at degree three, contributions weighted by β and γ appear in the refine stage but not in the first and only smoothing stage. Using separate multipliers for each stage therefore grants no additional freedom for this case: there are still just three parameters α_3^3, β_3^3 and γ_3^3 to use in tuning the scheme for bounded curvature. We therefore find the same negative solution for α_3^3 shown in Figure 5.5(a).

- Figure 5.6 shows that the natural configuration at degree five and valency three is potentially less stable if different multipliers are used in the refine and smoothing stages.

- Using separate multipliers for each stage introduces an unmanageable number of parameters at high degrees.

- Using separate multipliers for the refine stage but the same values for all smoothing stages would still allow the solution shown in Figure 5.6(b), and brings the number of parameters down to just five. However, the influence of the refine stage becomes weaker after each smoothing stage. There are therefore cases (e.g. degree nineteen and valency three) which have no bounded-curvature solution with three multipliers, and also have no bounded-curvature solution if we allow a modified refine stage.

This approach is therefore not a good way to achieve bounded curvature for valency three. Furthermore, this investigation raises the possibility that, in attempting to find bounded-curvature solutions when $n = 3$, we might modify the blocks $\hat{\mathbf{S}}_\omega$ for $\omega \neq 0$, thus altering the natural configuration from the result in §5.3. The results from §5.3 are simple to implement and hold good theoretical properties, so we would like any future modifications to modify only the block $\hat{\mathbf{S}}_0$. In the degree five case, for example, this means that the subdivision scheme must retain the natural configuration shown in Figure 5.6(a).

At the start of §5.3, I also limited the bounded-curvature modifications to mask tuning. Relaxing this constraint therefore offers another possibility for resolving the problems we encounter when $n = 3$. The more general framework, which was used by Barthe and Kobbelt [7], is called *stencil tuning*, and allows the weight of any contribution to be modified, not merely the extraordinary vertex. Consider the problematic case shown in Figure 5.5(b): degree five and valency three. If we modify only the block $\hat{\mathbf{S}}_0$, then instead of one parameter α, we have a bivariate barycentric space (e, f) which determines the new position of the extraordinary vertex. The triangle $e > 0, f > 0, e + f < 1$ is the set of values which uses positive weights in this affine combination, and therefore retains the convex hull property. Unfortunately, this set does not contain a value where $\lambda^2 = \mu_0$, so the additional freedom does not help us to find a bounded-curvature solution in this case.

As a result of these investigations, I believe that for a bounded-curvature version of NURBS-compatible subdivision which retains the convex hull property, the structure of the affine combinations given by the regular case is too tight a constraint. The solution I present in §5.4.2 therefore modifies this structure. In my first attempt to do so, I achieved bounded-curvature solutions in the uniform case, but omitted to consider the generalisation to non-uniform knot vectors. The result was that the position of a three-valent vertex was always modified during a subdivision step, even if no knots were inserted near that vertex. This is clearly undesirable, as the knot insertion strategy I presented in §4.3 relies on the fact that the surface is purely a function of inserted knots. It therefore requires regions of the mesh where no knots are inserted to be unmodified during a subdivision step, and so this experience adds an additional requirement for the bounded-curvature modifications at valency three.

Requirements

We can now compile a full list of those requirements; we are looking for a modification of the subdivision rules in the regular case, which, when $n = 3$:

- results in a subdivision matrix that satisfies the bounded-curvature condition $\lambda^2 = \mu_0$,
- maintains, from the mask tuning described in §5.3, the Fourier blocks $\hat{\mathbf{S}}_\omega$ where $\omega \neq 0$ (thus keeping the same values for λ, μ_2, and the same natural configuration as the solutions found there),
- only modifies the position of a point if a knot is inserted in the domain of its basis function,
- maintains the convex hull property by using only non-negative weights.

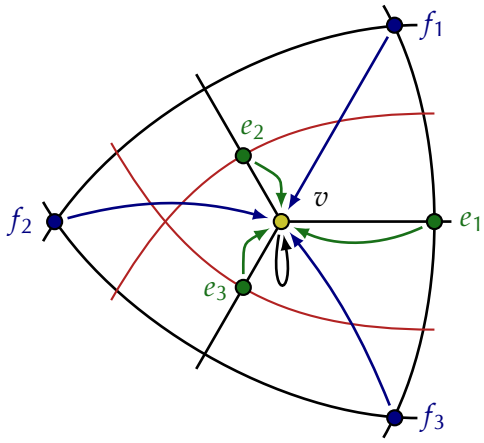


Figure 5.7: Final affine combination used to position 3-valent vertices for bounded curvature. Red lines indicate knots which have been inserted in this subdivision step (as an illustration, I show a case where no knot has been inserted into the interval on the right). The extraordinary vertex, in the centre, takes a new position calculated from the existing position v ●, the points e_i ● which are edge-connected at the *end* of the subdivision step, and the points f_i ● which are in the positions which were face-connected at the *start* of the subdivision step.

Additional smoothing stage 5.4.2

Here I describe a modification which meets all of the above requirements for a useful range of degrees from 3 to 13. For degree greater than 13, we can satisfy the first three requirements but require affine combinations with negative weights in order to do so.

The problem at valency three, as shown in Figure 5.5, is that the quadratic cup eigencomponent fails to shrink fast enough. Instead of using a negative weight for the contribution from an extraordinary vertex to its successor, however, we can shrink μ_0 by letting the extraordinary vertex receive contributions from vertices that are farther away than in the regular case. This is because vertices which are farther away in the mesh are also farther away from the extraordinary vertex in the eigenvector corresponding to μ_0 (see Figure 5.8 for an example). Allowing the extraordinary vertex to be influenced by more distant vertices therefore makes it possible to shrink μ_0 without using negative weights.

By taking contributions from vertices which are farther away than in the regular case, the basis functions of some vertices gain a larger support. This may be undesirable[16], but for most applications, the convex hull property is likely to be more important. In fact the modification described here allows for a trade-off between the two properties, so different applications could make different decisions about the relevant priorities. It would be possible, for example, to meet all of the above requirements at degrees even greater than 13, if we were also willing to allow a greater increase in support.

The modification introduces an additional, final smoothing stage, affecting only three-valent vertices (see Figure 5.7). We need to take a contribution from vertices that are sufficiently far away from the extraordinary vertex to shrink μ_0 at a rate of λ^2, but we also want to avoid complications arising from arbitrary connectivity. My solution is therefore to take f_i, the vertices which were face-connected to the extraordinary vertex *at the start of the subdivision step*. In the uniform

[16] §6.3 discusses a disadvantage that arises from this increase in support near surface boundaries.

δ_3	δ_5	δ_7	δ_9	δ_{11}	δ_{13}
0.0013	0.0813	0.1915	0.3537	0.5732	0.8550

Table 5.4: Values for δ_d to achieve bounded curvature for valency three.

case, the f_i therefore lie at the corners of the two rings of faces surrounding an extraordinary vertex. If a face has not been subdivided in one direction or both, however, they will be more closely connected. In Figure 5.7, for example, f_1 and f_3 are only a knight's move away from the extraordinary vertex. I also use e_i, the vertices connected to the extraordinary vertex by edges, and the position, v, of the three-valent extraordinary vertex before this final smoothing stage. The new position of the extraordinary vertex, $\tilde{v}$, is then given by

$$\tilde{v} = \rho\, v + (1-\rho)\left[(1-\delta_d)\sum_{i=1}^{3}\tfrac{1}{3}e_i \;+\; \delta_d\sum_{i=1}^{3}\tfrac{1}{3}f_i\right]$$

where ρ is the product, over every smoothing stage, of the normalised weight in the contribution from the extraordinary vertex to its successor. Including ρ allows us to satisfy the third requirement in the list above: if a three-valent vertex has no knots inserted into the domain of its basis function then $\rho = 1$, and therefore $\tilde{v} = v$ as required.

This final smoothing stage makes it possible to solve $\lambda^2 = \mu_0$ using the same numerical solver as when finding α_n^d for $n > 4$, but varying δ_d instead of α_3^d (see Table 5.4). Note that every affine combination in the univariate knot insertion algorithm from Chapter 3 uses non-negative weights. The same is therefore true of the tensor product, and of the modifications for bounded curvature, as the multipliers α_n^d, β_n^d and γ_n^d are all non-negative. Therefore $0 \le \rho \le 1$, as ρ is the product of non-negative weights, and if $0 \le \delta_d \le 1$, then the NURBS-compatible schemes hold the convex hull property.

In solving $\lambda^2 = \mu_0$, we find $0 \le \delta_d \le 1$ for the six degrees where d ranges from 3 to 13: the resulting values are given in Table 5.4. With the inclusion of an additional smoothing stage, α_3^d has relatively little effect, as the position of the extraordinary vertex is largely determined by the value of δ_d. However, we do want to make sure that $\delta_3 \ge 0$, and therefore I set $\alpha_3^d = \frac{4}{3}$ for all d as a useful simplification that achieves this. When knot vectors are uniform, this value scales contributions from three-valent points to their successors so that the total unnormalised weight is the same as in the regular, four-valent case.

Polar artifacts 5.5

An observation from the bounded-curvature multipliers graphed in Figure 5.4 is that α_n^d takes on large values for high n. These values arise because the eigenvector corresponding to the μ_0 eigenvalue becomes increasingly 'pointed' as n grows. An example is shown in Figure 5.8, which plots the eigenvector corresponding to μ_0 as the z co-ordinates of a control mesh which uses the natural configuration in the xy-plane. Sabin [75] called this control mesh the *quadratic natural configuration*, by analogy to Ball and Storry's (linear) natural configuration. A high value for α is necessary to maintain this pointed shape during the course of a subdivision step, and if there are multiple smoothing stages (when d is large), α must be even higher, as shown in Figure 5.4.

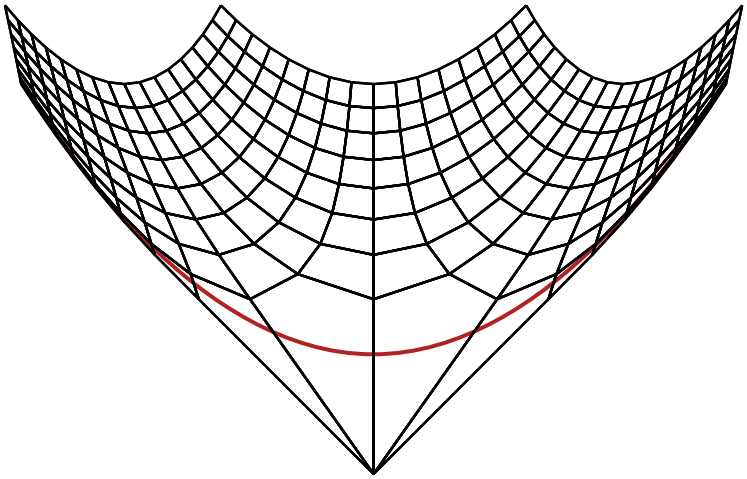


Figure 5.8: The quadratic natural configuration for bounded-curvature subdivision schemes. This figure shows a side view of the quadratic natural configuration corresponding to μ_0 for degree five and valency eight. An elliptic paraboloid through the origin is drawn in red, for comparison, to show that this eigenvector is far from satisfying the 'local quadratic precision' condition of Barthe and Kobbelt [7].

However, it is not immediately obvious to what extent the shape of this eigenvector, and hence a high value for α, is desirable or necessary.

Indeed, Barthe and Kobbelt [7] explicitly tried to optimise the quadratic natural configuration to be a mesh which is sampled from an elliptic paraboloid: the surface shown in red in Figure 5.8. They called this goal 'local quadratic precision'. Prautzsch and Reif [64] showed that if a scheme is C^2, then the eigenbasis functions associated with the quadratic eigencomponents must be in the span of quadratic functions on the characteristic map. However, this condition applies to the *eigenbasis functions*, not the *eigenvectors*, and my results show that for stationary bounded-curvature subdivision schemes, the eigenvector may need to hold a shape which is significantly different from a sampled quadratic surface.

The requirement for a pointed shape in this eigenvector comes from a high value for λ. Where $\lambda > \frac{1}{2}$, Sabin and Barthe [77] described the effect in extraordinary regions as a *polar artifact*: the faces surrounding such an extraordinary vertex shrink slower than faces in the regular regions of the surface, leading to a poor approximation if the subdivided control mesh is used as a way to render the limit surface [3]. Sabin and Barthe [77] and Augsdörfer et al. [3] stress that the polar artifact is not a property of the limit surface. However, the observations in this section allow us to clarify that a high value for λ does affect the limit surface, in that it alters the function which gives the limit surface from a control mesh. Where $\lambda > \frac{1}{2}$, any bounded-curvature scheme will produce a limit surface which, for convex shapes, is *farther* from the extraordinary vertex than in the regular case. This is because the only way to maintain the appropriate quadratic scaling $\lambda^2 = \mu_0$ is to have an extraordinary vertex which is offset farther from the limit surface, as shown in Figure 5.8. An artifact is defined [77] as a feature of the limit surface which cannot be controlled by moving the control points. For bounded-curvature subdivision, the polar artifact can, therefore, appear as a 'true' limit-surface artifact. For example, there is no way to control how closely a subdivision surface defined by a cube mesh approximates a sphere, without increasing the number of control points (possibly using a subdivision step). Using a bounded-curvature subdivision

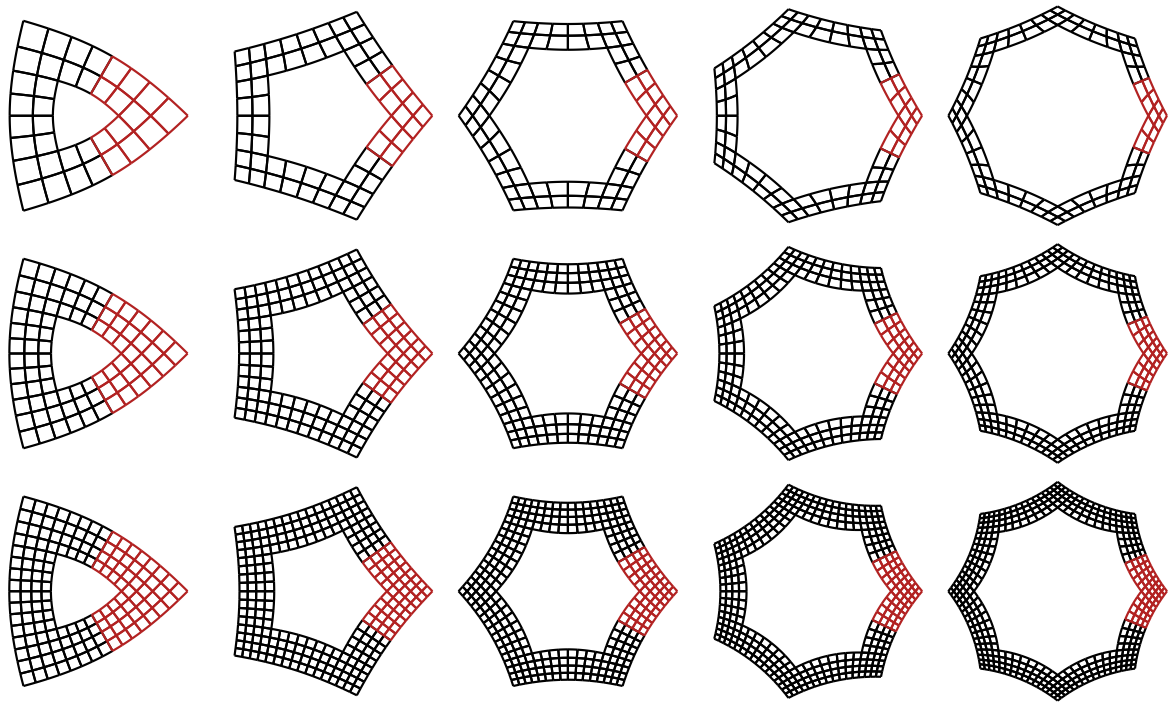


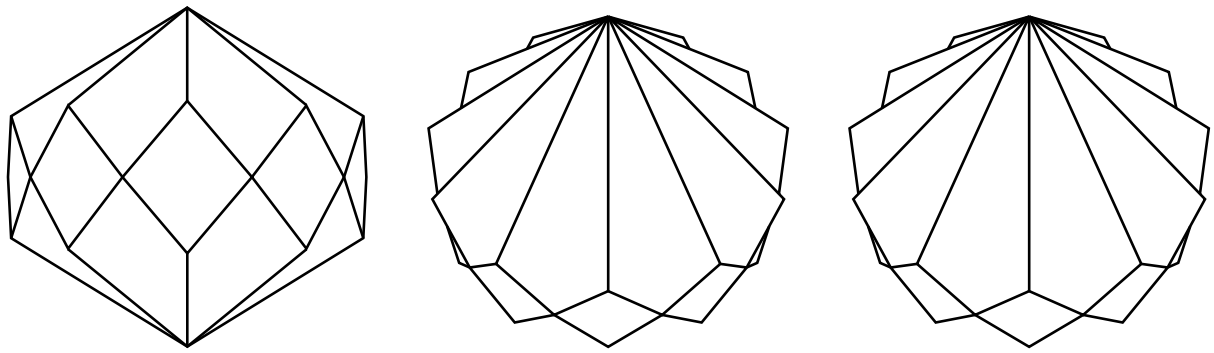
Figure 5.9: Characteristic rings resulting from bounded-curvature modifications. Characteristic rings [54] are shown at valencies three and five to eight (left to right) and degrees three, five and seven (top to bottom). In each case, the rings are shown by drawing the boundaries of B-spline patches, where the first sector is drawn in red.

scheme, the limit-surface approximation is likely to be *worse* than the Catmull-Clark algorithm, as the three-valent corners of the cube have a greater influence over the limit surface than the regular points that are introduced during subdivision.

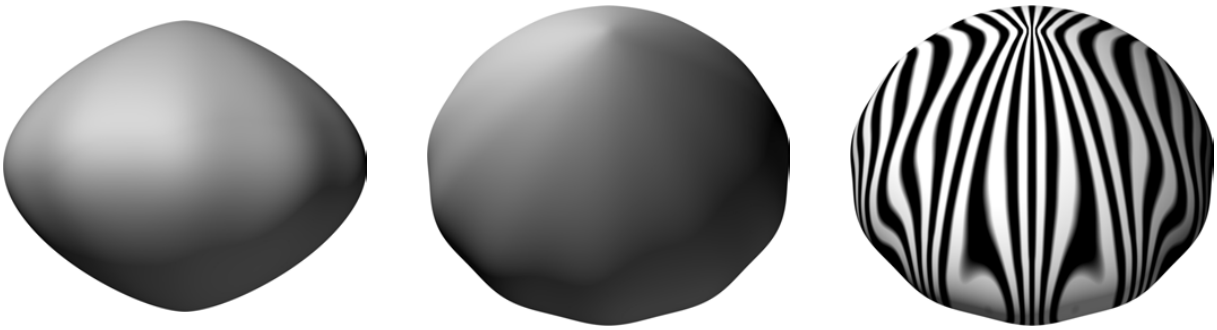
Understanding this effect also explains the results of Karčiauskas and Peters [38], who show that on near-elliptic data, the bounded-curvature schemes of Augsdörfer et al. [2] are notably flatter than we might expect around high-valency singularities. Karčiauskas et al. [40] made the same observation for the limit surface of Sabin's bounded-curvature scheme [74] on a very similar control mesh. However, from this section, we know that in order to produce an approximately quadratic shape from bounded-curvature stationary subdivision schemes which have $\lambda > \frac{1}{2}$, the required control mesh is not, in fact, a sampled quadratic, but is instead a pointed shape, such as the example shown in Figure 5.8.

Proving C^1 continuity 5.6

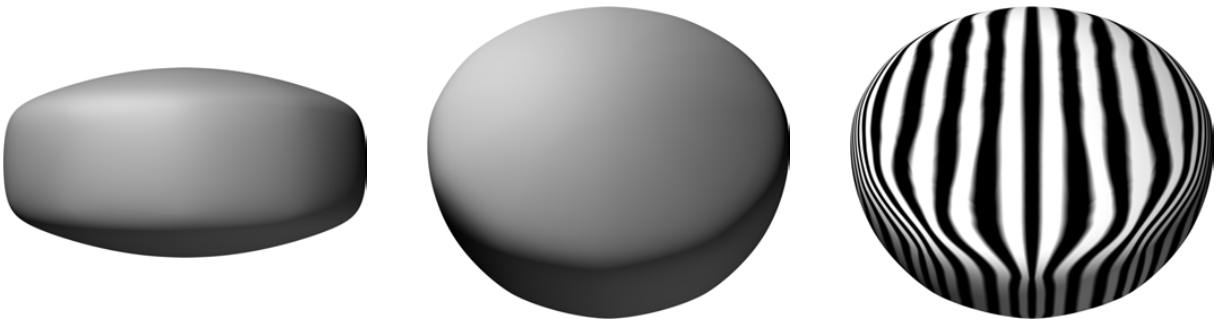
This chapter has focused on the second-order properties of NURBS-compatible subdivision, but we have not yet verified that the schemes, as modified for bounded curvature, have good *first-order* behaviour. We would like to verify that the schemes are tangent-plane continuous and regular for all d and n. The knot insertion strategy in Chapter 4 creates a region of uniformity around every extraordinary vertex surrounded by non-zero knot intervals. As I explained in §5.1, the continuity of the limit surface is therefore determined by a stationary subdivision matrix, and we can use eigenanalysis to prove that bounded-curvature NURBS-compatible schemes produce C^1 regular surfaces. Doing so requires a proof that Reif's conditions [69] hold (see §5.1.1). This



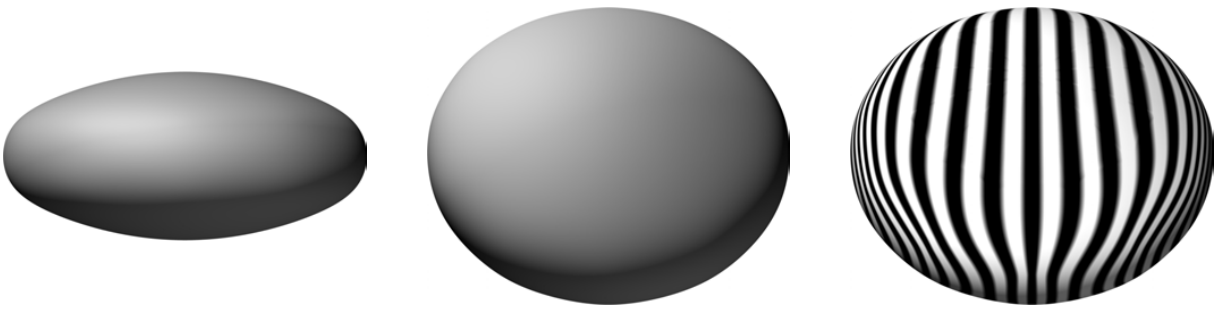
(a) Control mesh shown in the two different views which are used to render the surfaces below.



(b) Catmull-Clark [10]. Note the unbounded curvature at singularities corresponding to 12-valent vertices.



(c) NURBS-compatible subdivision, degree 3.



(d) NURBS-compatible subdivision, degree 9.

Figure 5.10: Surfaces defined by a control mesh with 12-valent vertices. Each surface is rendered with vertices projected to the limit surface; (c) and (d) use exact evaluation [90] to sample the limit surface in the region of the 12-valent extraordinary vertices. The rightmost images in (b), (c) and (d) are rendered using reflection lines to highlight the differences between the three surfaces.

section summarises work which provides this proof for $3 \leq d \leq 19$ and $3 \leq n \leq 50$. The work was completed in collaboration with others, in particular Ursula Augsdörfer, who implemented the proof using a strategy that I helped to formulate.

Reif's conditions [69] require that the characteristic map is regular and injective. However, researchers have found this hard to prove rigorously (e.g. [41]), and so for specific cases, Peters and Reif [54] provide equivalent conditions which are easier to verify. We used Theorem 5.25 from their monograph [54]. The subdivision matrices derived from the modifications in this chapter satisfy the conditions of this theorem, as they are *flip-symmetric*, *shift-invariant*, and the subdominant eigenvalues have Fourier index ± 1. The final precondition of the theorem is that the derivatives of the characteristic ring do not overlap. A *characteristic ring* is the building block of Reif's characteristic map (see Figure 5.9 for examples). The characteristic map is composed of an infinite sequence of nested rings, each of which is a scaled copy of the characteristic ring. We want to verify that, in each part of this ring, the derivatives with respect to the two B-spline parameter values lie in non-overlapping cones. Due to symmetry, however, we need only check that the derivatives in one direction of the first sector (drawn red in Figure 5.9) lie in the same quadrant [54]. This is enough to prove that the characteristic ring is regular and injective, and thus that the schemes are C^1 and regular.

The characteristic ring is made up of B-spline patches of the degree of the scheme. Using the convex hull property of B-splines, we can therefore prove this condition on the derivatives using the first differences of vertices in the natural configuration. For high valencies or high degrees, however, these differences may form only a loose bound on the B-spline derivatives, and hence fail to provide the required proof. In these cases, we can obtain a tighter bound by using subdivision steps to bring the first differences closer to the derivative; the required subdivision steps use only uniform B-spline knot insertion, so the Lane-Riesenfeld algorithm [42] provides a straightforward way to compute the subdivided vertices for arbitrary degree. Dr Augsdörfer has verified this condition [1], and hence proved C^1 continuity, for degree from 3 to 19 and valency from 3 to 50. The trends suggest that no problem is likely to occur at higher valencies and degrees.

Summary 5.7

This chapter has shown that we can modify NURBS-compatible subdivision schemes to have bounded curvature at all degrees and valencies, and simultaneously retain the convex hull property for degrees 3 to 13. At valency three these modifications require a small increase in the support of surrounding basis functions. This is the first time that arbitrary-degree subdivision schemes have been tuned for bounded curvature. The resulting schemes offer additional smoothness at higher degrees, which is the property of B-splines shown in Figure 2.2, without the second-order problems exhibited by previous arbitrary-degree schemes, which are shown in Figure 5.1 and described in §5.2.

Figure 5.10 shows the effect of the bounded-curvature modifications on a control mesh containing a high-valency vertex. Subdivision surfaces, such as the example shown in Figure 5.10(b), typically suffer from poor shape around points with high valency. Figure 5.10(c) shows that the bounded-curvature modifications produce a much smoother shape, even at low degree, compared to schemes with divergent curvature. We can also increase the degree to obtain a yet smoother surface: the degree nine surface in Figure 5.10(d) shows even less curvature variation.

Recall that although these bounded-curvature solutions apply only to uniform knot intervals, almost all *non-uniform* configurations are also able to benefit from bounded-curvature extraordinary regions. This is a result of the knot insertion strategy in Chapter 4, which creates uniform knot intervals around each extraordinary vertex. This chapter has therefore shown how to create NURBS-compatible subdivision schemes which provide the best possible curvature behaviour for stationary subdivision schemes, demonstrating the thesis that I introduced in §1.3.

Conclusion 6

Chapters 3, 4 and 5 describe a class of NURBS-compatible subdivision surfaces. These surfaces are able to reproduce any odd-degree NURBS surface exactly, and can also represent surfaces of arbitrary topology by allowing a control mesh to include extraordinary vertices. They are therefore a candidate to fulfil Ma's vision [48], introduced in §1.2, of a subdivision representation that is able to offer the complete set of NURBS features.

Contributions 6.1

The main contribution of this dissertation is a surface representation that unifies NURBS and subdivision surfaces. In pursuit of that goal I have also made several other contributions:

- In Chapter 3, I described a symmetric refine-and-smooth factorisation for non-uniform B-spline knot insertion. The only alternative factorisation for general non-uniform knot vectors is Schaefer's algorithm [83]. Schaefer's algorithm is asymmetric, however, which makes it unsuitable for use on subdivision surfaces (see §3.2.2).

- This is the first refine-and-smooth factorisation that is able to accommodate *selective* knot insertion, where some knot intervals are left unmodified. This property is important in order to elegantly handle multiple knots.

- In §4.3, I observed that a large disparity in knot intervals can be detrimental to the shape of a non-uniform subdivision surface, and developed a knot insertion strategy that militates against the worst behaviour.

- In §5.3, I hypothesised a connection between the stability of a subdivision scheme's natural configuration and the quality of surfaces produced by that scheme. This may lead to a new measure by which subdivision schemes can be evaluated.

- In §5.5, I explained that for bounded-curvature subdivision schemes, contrary to previous expectations [7], a high subdominant eigenvalue λ necessarily implies that an extraordinary vertex exerts a weaker influence on the limit surface.

- As a consequence of Chapters 4 and 5, I have developed the first non-uniform subdivision surfaces with bounded curvature at singularities. Previous non-uniform subdivision schemes [51, 89] are based on the Catmull-Clark [10] behaviour where each singularity has either zero or unbounded curvature [40].

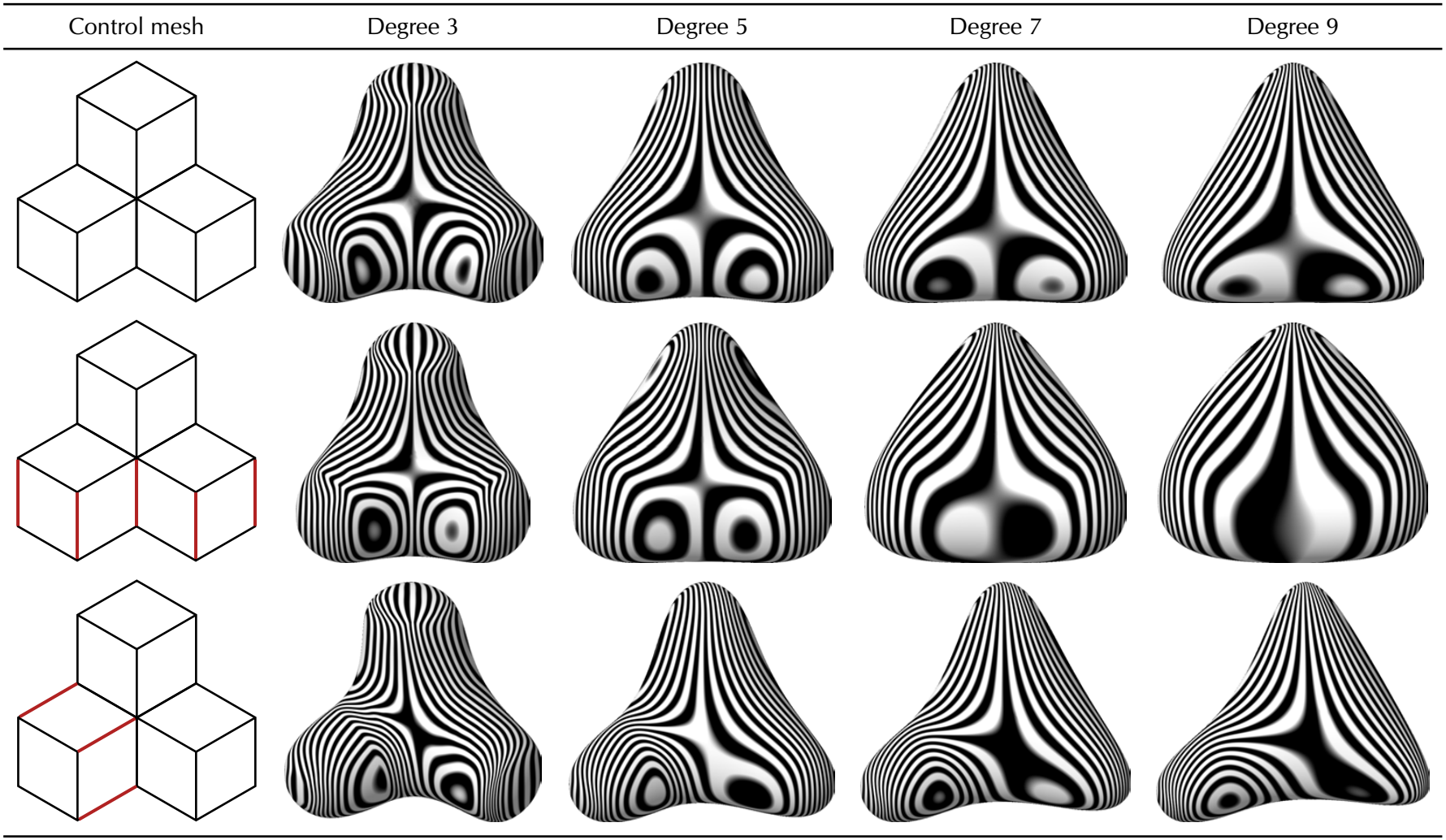


Table 6.1: Surfaces with three different knot configurations (top to bottom) at degrees 3 to 9 (left to right). Each surface is drawn with reflection lines and is defined by the control mesh shown on the left. In each row, the knot intervals shown in red are five times larger than the others; the first row shows the uniform case. Note the widening influence of a knot interval as degree increases. The six surfaces which are both high-degree and non-uniform (bottom-right) cannot be generated by any previous subdivision scheme.

- Chapter 5 also gives the first arbitrary-degree uniform subdivision surfaces with bounded curvature at singularities. Previous arbitrary-degree subdivision schemes [62, 63, 91, 93, 103, 105] all have either zero or unbounded curvature at singularities (see §5.2).

Applications 6.2

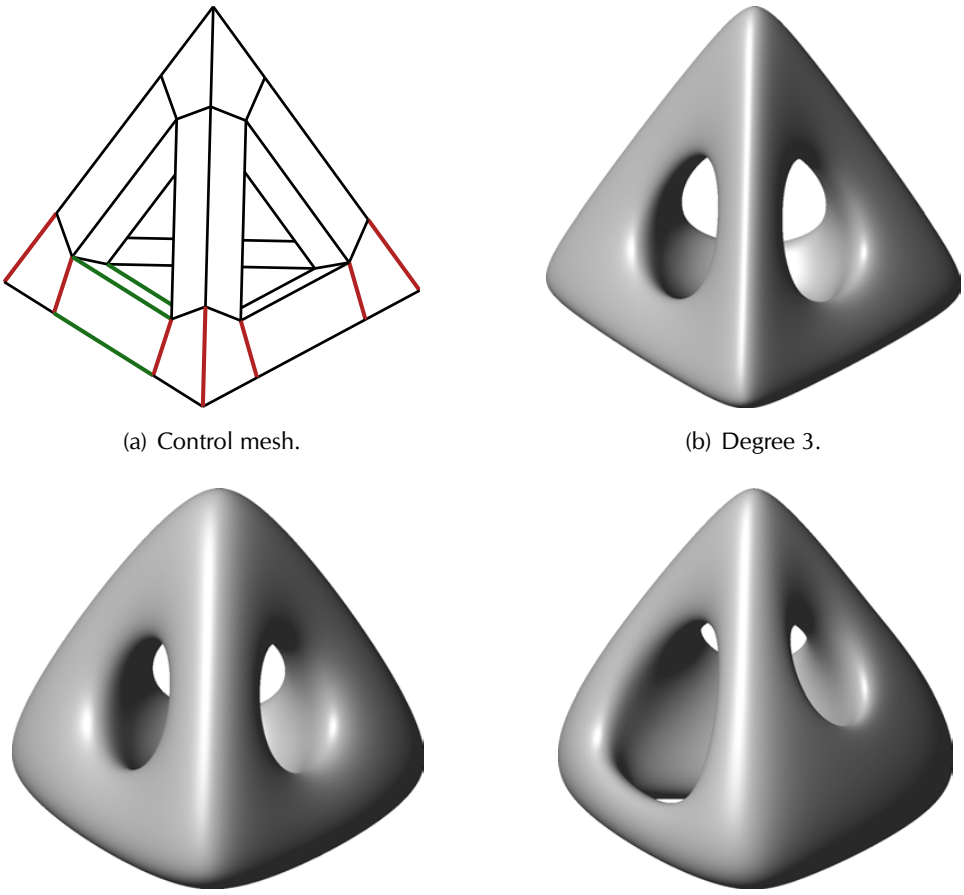
Table 6.1 and Figure 6.1 show example NURBS-compatible subdivision surfaces at a range of degrees and with various knot configurations. An immediate application of these surfaces is that users need no longer be presented with mutually exclusive surface primitives for NURBS and Catmull-Clark subdivision surfaces. Instead, software applications that allow freeform surface design using quadrilateral control meshes can use NURBS-compatible subdivision surfaces to present a unified interface to users, granting control which is familiar from NURBS (non-uniform parametrisations at arbitrary degree) alongside the flexibility of arbitrary topology, which is familiar from subdivision.

The availability of non-uniform parametrisations may be useful wherever subdivision surfaces are textured (for example, procedurally) using the piecewise parametrisation associated with the control mesh. Like NURBS, the class of surfaces presented here allows each face to be parametrised by a rectangle, rather than the unit square. A chordal parametrisation, for example, assigns knot intervals to the geometric length of the associated control mesh edges. Figure 6.2 shows that this parametrisation can reduce the distortion of textures by increasing the regularity of the parametrisation's projection onto the surface. Gonsor and Neamtu [32] also highlight that, for a fair surface, it is important to use non-uniform parametrisations for control meshes with vertices which are non-uniformly spaced.

Yet another use for non-uniform parametrisations is to allow good control of the boundaries of subdivision surfaces. §3.3.7 described how to handle boundaries in the curve case and this is easily transferred to surfaces. As a result, we can use multiple knots to create surfaces which meet a univariate B-spline curve at the boundary, whilst providing good control of both tangent and curvature. Existing boundary conditions, such as duplicating control points to gain a clamped surface, suffer from zero Gaussian curvature at the boundary [78].

Figure 5.10 and Table 6.1 show that NURBS-compatible subdivision surfaces may also be useful where high-quality surfaces are required, as they offer the possibility of using higher degrees to increase fairness. The disadvantages of doing so apply to B-splines in general: there is a greater computational cost, increased shrinkage when comparing the limit surface to the control mesh, and control points have a less local influence over the surface. However, I believe that even in the completely uniform case, the degree 5 surfaces are an interesting alternative to Catmull-Clark. The computational cost is only slightly higher (being degree 5 rather than degree 3), and the resulting surfaces strike a good balance between respecting the shape of the control mesh and reducing curvature variation. In addition, the surfaces at degree 5 provide sound theoretical properties, such as C^4 continuity in regular regions, and bounded curvature at singularities.

Returning to the motivation I gave in §1.1, I anticipate that the main use for these surfaces will be within CAD. Here NURBS-compatibility is particularly crucial, because of the importance of the NURBS representation to the current set of CAD tools, and because of the large volume of existing freeform data that already uses NURBS. Within CAD, NURBS-compatible subdivision surfaces may prove useful as a direct replacement for NURBS, as I suggested at the start of



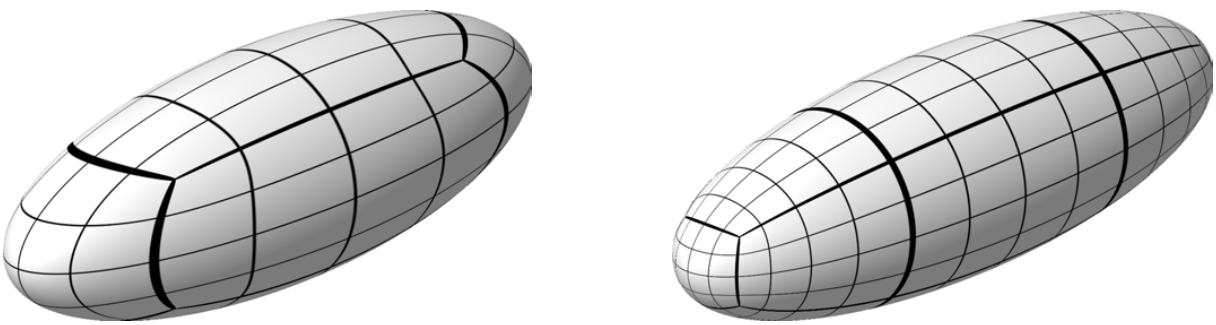
(a) Control mesh.

(b) Degree 3.

(c) Degree 7.

(d) Degree 7, non-uniform knots.

Figure 6.1: The same control mesh subdivided in three different configurations. All surfaces have topological genus 3. In regular regions, the degree 3 surface (b) is C^2, and the degree 7 surfaces (c) and (d) are C^6. No previous subdivision scheme or NURBS surface patch can represent the surface (d). The knot intervals modified to give (d) are shown in (a); the red interval is ten times greater than the unmarked intervals, and the green interval is four times greater. Comparing (d) with (c), note that in this case the non-uniform intervals change the whole surface, because the influence of a knot interval grows with degree for both NURBS and NURBS-compatible subdivision surfaces.



(a) Uniform parametrisation.

(b) Chordal parametrisation.

Figure 6.2: Non-uniform knot intervals affect the projection of a surface's parametrisation. Both (a) and (b) show a cuboid subdivided at degree three, with subdivision surfaces drawn with isoparameter lines. Setting knot intervals to chord length increases the regularity of the parametrisation on the surface (b).

this section. However, they could also play a more subservient role, by addressing the gaps and inaccuracies in NURBS models that result from stitching multiple NURBS patches together. Sederberg et al. [87] were able to repair the intersection of two NURBS surfaces by converting NURBS into T-splines, merging to form a watertight surface, and then exporting the resulting surface back into NURBS. It is possible that NURBS-compatible subdivision surfaces could be used in a similar way, particularly where multiple NURBS surfaces are intended to meet at a common point. For any number of surfaces other than four, it is not possible to represent this union as a single NURBS patch that guarantees good continuity properties. However, we could follow a similar strategy to Sederberg et al. [87] by converting the NURBS surfaces into NURBS-compatible subdivision surfaces. As this class is a superset of NURBS (see Figure 2.9), the conversion does not introduce any error, irrespective of the degree of the NURBS patches or the value of the knot intervals. The subdivision surfaces could then be merged to improve the surface smoothness across boundaries and at the singularity, using knot insertion to create compatible knot vectors if required. This process necessarily modifies the input to increase continuity, but the modification could be constrained to lie within a user-specified tolerance. The resulting NURBS-compatible subdivision surface would be watertight and, assuming there are non-zero knot intervals adjacent to the extraordinary vertex, would have bounded curvature at the singularity.

Furthermore, NURBS-compatible subdivision surfaces can be exported as NURBS surfaces to within any given deviation tolerance: the regular regions of the surface are already NURBS patches, and the extraordinary regions can be represented as a nested series of *spline rings*. This approximation can be brought arbitrarily close to the target subdivision surface, by increasing the number of spline rings that are exported. Finally, a finite cap is used to fill the remaining region near the singularity. There are many possible finite fillings: one possibility is Prautzsch's freeform splines [61], which would allow any required level of continuity. As a whole, this pipeline therefore offers a way of taking any number of degree-d NURBS patches, which may not even meet with C^0 continuity, and returning a replacement set of patches which meet C^{d-1} across edges and have, at least, bounded curvature at the singularity. This could prove to be a useful tool for the repair of CAD models.

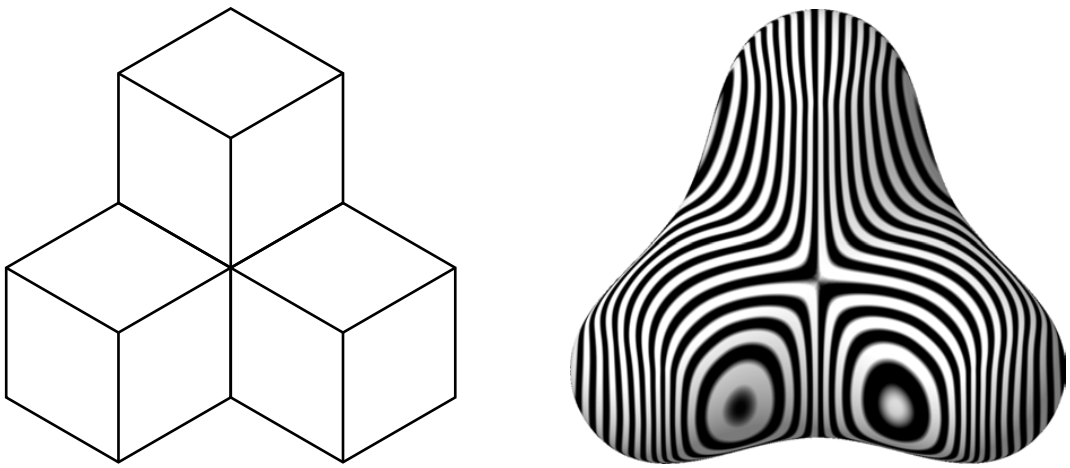
Limitations 6.3

Karčiauskas et al. [40] note that bounded-curvature schemes can suffer from a lack of fairness. This effect seems to be visible in the degree 3 surfaces of Table 6.1, since the reflection lines are not as smooth as a Catmull-Clark surface on the same data (Figure 6.3). I believe this is another consequence of observations in §5.5, as the 3-valent vertices have $\lambda < \frac{1}{2}$ and the limit surface is therefore pulled closer to the 3-valent corners than for Catmull-Clark[17]. At degree 5 and higher, however, the reflection lines shown in Table 6.1 compare favourably with those in Figure 6.3.

§4.4 also gives several limitations of the knot insertion strategy I described in §4.3, which creates uniform extraordinary regions from almost all non-uniform configurations. In summary, these limitations are:

- The surface is a discontinuous function of its knot intervals, as a result of the first part of the strategy which subdivides large intervals first (§4.3.1).

[17] At 3-valent singularities a Catmull-Clark surface holds zero, rather than bounded, curvature



(a) Control mesh. (b) Catmull-Clark surface with reflection lines.

Figure 6.3: The Catmull-Clark surface defined on the control mesh used in Table 6.1.

- The comparison between knot interval sizes, as described in §4.3.1, is global, although large knot intervals have no effect on distant small intervals. The comparison could therefore be localised without adversely affecting surface smoothness.
- It is not possible to create uniform extraordinary regions where there is a zero knot interval adjacent to an extraordinary vertex. Surface singularities that lie on multiple knots therefore have no continuity guarantees: neither the proof of C^1 regularity (§5.6) nor the curvature bounds from §5.3 and §5.4 can apply.

A further limitation, although not specifically of the knot insertion strategy, relates to the ability to selectively insert knots. The possibility of leaving knot intervals unmodified, as a result of the algorithm in §3.3, is crucial to the knot insertion strategy in §4.3 and to handling multiple knots effectively. For NURBS surfaces, knots can be inserted selectively without side effects, as knot insertion does not alter the limit surface. Subdivision surfaces do not have the same property, however, so the similarity to NURBS knot insertion partially breaks down in extraordinary regions. It is hard to see how this limitation could be overcome without providing a finite collection of patches to fill the extraordinary region, with parametrisations that match the boundary curves. Existing methods to provide such a finite filling for non-uniform surfaces (as presented by Piegl and Tiller [59], for example) are only able to provide weak or approximate guarantees on continuity.

Another limitation is introduced by my decision (in §4.1) to require that opposing edges of every face have equal knot intervals. This means that, like NURBS, the chordal parametrisation shown in Figure 6.2(b) can only be assigned exactly when the edges in an entire strip of faces have equal geometric lengths. This is unlikely to be the case for many control meshes designed in practice, and in these cases a chordal parametrisation can therefore only be approximated (by taking an average of the lengths of the edges in each strip, for example).

Finally, there is an unintended consequence of the additional smoothing stage for valency three (§5.4.2) when a 3-valent point appears close to a surface boundary. Using the method described in §3.3.7, a 3-valent vertex may attempt to receive contributions from a point marked as outside the boundary, and which would have had no influence on the extraordinary vertex

in the regular case. The result is that the surface is curtailed around the 3-valent vertex in a counter-intuitive way, since all other bounded-curvature modifications maintain the *structure* of the regular affine combinations, and only modify the associated *weights*. It might be possible to cater for this scenario by simply ignoring contributions from points marked as outside the boundary which would have no influence over a 3-valent vertex in the regular case. The weights on other contributions would then have to be scaled to normalise the affine combination. This may have undesirable consequences, however, for the shape of the surface in the region of such 3-valent vertices; further work is required to assess whether this straightforward modification offers the best possible solution.

Even degrees 6.4

The odd-degree NURBS-compatible surfaces described in this dissertation have a natural counterpart for even degrees, by using the refinement pattern shown in Figure 4.1(a) rather than 4.1(b). For this type of refinement, every vertex is 4-valent, but it is possible to introduce *extraordinary faces*, which have a number of edges other than four.

The knot insertion algorithm described in §3.3 caters for even degrees using the same framework as for odd degrees. Furthermore, the set of knot intervals, when degree is even, has the same primal structure as the odd-degree case: knot intervals are associated with a strip of edges, rather than a strip of faces, and so they diverge at the singularities of the surface in exactly the same way. Therefore I anticipate that the knot insertion strategy described in §4.3 would apply equally to even-degree schemes. The two missing components for a complete even-degree solution are:

- A way to generalise the tensor product of univariate knot insertion when there are extraordinary faces (i.e. the even-degree counterpart of calculating refine-and-smooth stages one face at a time, as described in §4.2).

- Bounded-curvature modifications for even-degree schemes (i.e. the even-degree counterpart of §5.3 and §5.4).

As I explained in §5.1, the Doo-Sabin subdivision scheme [21] has an optimal eigenspectrum, in that the first six eigenvalues, ordered by magnitude, are the same for all valencies. The continuity at singularities is therefore exactly the same as for regular faces: the surface is C^1 wherever B-spline patches meet. The Doo-Sabin scheme generalises biquadratic B-splines, and so in the uniform case when degree is 2, we would like to use exactly the Doo-Sabin weights.

The Doo-Sabin scheme also hints at a way to tackle the first of the missing components above, as the weights within an extraordinary face are generalised from the regular case using the projection of an appropriately-sized circle. We can use the same idea to formulate a generalisation for arbitrary weights, where the weights are derived instead from the knot insertion algorithm in §3.3.

However the main hurdle to overcome, in creating an analogue for even degrees, is tuning the schemes for bounded curvature. Where degree is even, it is *faces* that have arbitrary valency, and yet the knot insertion algorithm acts on *vertices*. This creates a mismatch which I believe would prove hard to reconcile in a solution as elegant as the odd degree case. Furthermore, the tuning in Chapter 5 works well because we have an extraordinary vertex, which for dominant

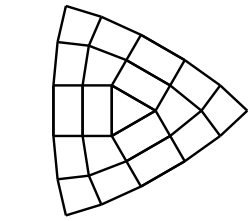
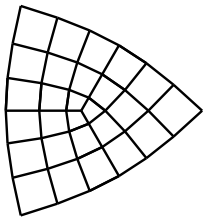
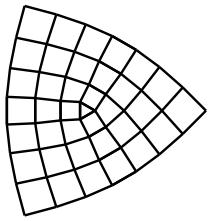
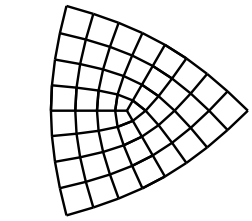
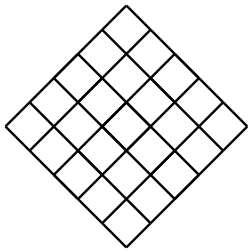
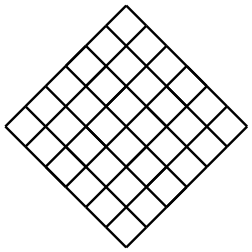
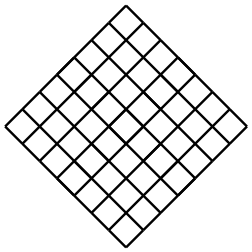
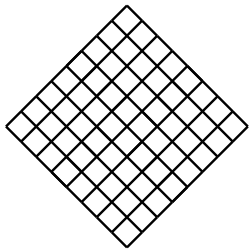
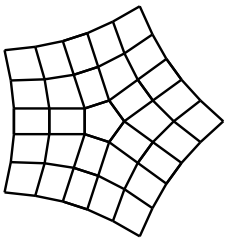
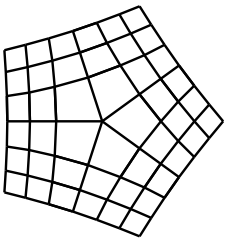
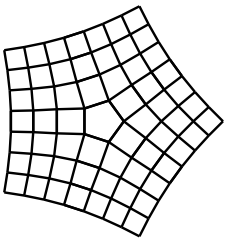
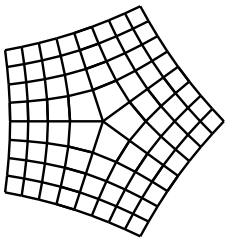
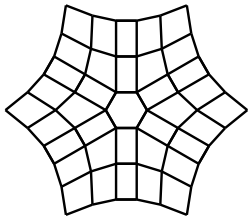
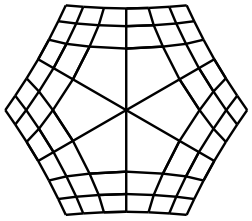
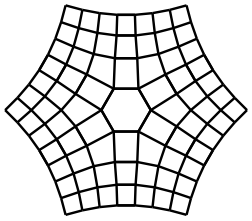
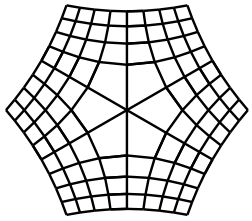
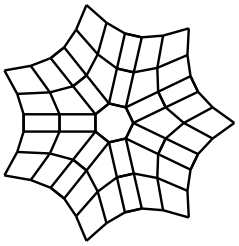
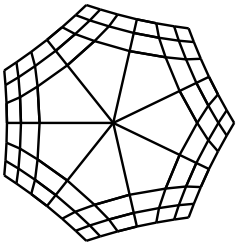
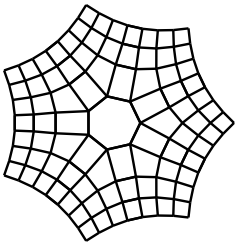
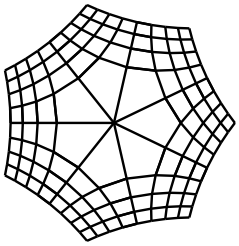
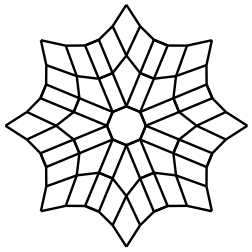
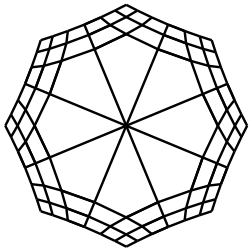
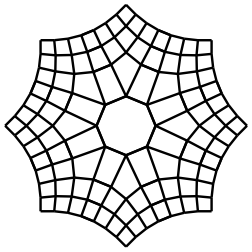
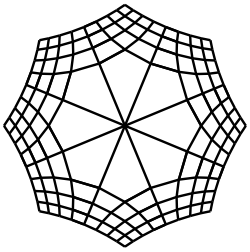
Degree 2	Degree 3	Degree 4	Degree 5
			
			
			
			
			
			

Table 6.2: Natural configurations for both odd- and even-degree bounded-curvature schemes. The column for degree 2 shows the natural configurations for the Doo-Sabin scheme [21].

eigencomponents is located at the singularity, with which to control the position of vertices along edges and across faces. For even degrees, there is no centrally-located position to use as an anchor for this kind of modification, and weight tuning is therefore much less successful.

Despite these challenges, a preliminary investigation has discovered a way to obtain bounded curvature at degrees 2, 4, 6 and 8, and at valencies 3 to 20. Figure 6.2 shows natural configurations for some of these modifications, with the odd-degree bounded-curvature schemes for comparison. I used *five* tunable parameters to achieve this for even degrees, instead of the *three* required for odd degrees. These correspond to mask-tuning parameters which play a similar role to α, β and γ introduced in Chapter 5, as well as two new values which only alter the weights of contributions within an extraordinary face. These parameters accommodate the Doo-Sabin weights as a particular case, which I used as the bounded-curvature modifications for degree 2. However, there are several questions surrounding the derivation of these parameters:

- Are five parameters truly necessary for good bounded-curvature solutions?
- Are there any guides, similar to the stability heuristic used in §5.3, that can help us to select bounded-curvature modifications with a small amount of curvature variation? Ideally, we would like to draw on the experience of a thorough analysis of curvature variation, just as the stability heuristic draws on the work of Augsdörfer et al. [2].
- Can we constrain the set of modifications so that there is only one possible value for each parameter at each degree and valency? The modifications used to generate Figure 6.2 used parameters that were chosen from a narrow set rather than uniquely determined.

Furthermore, the modifications that I have developed so far lead to an implementation which is relatively complicated compared to the odd-degree case. For a worthy even-degree analogue, all these issues require further analysis and improvement.

Future work 6.5

As well as the even-degree case, there are several other areas where future work could focus attention. The shortcomings of the current knot insertion strategy (summarised in §6.3) suggest that there may be an alternative strategy that achieves the same goals without as many disadvantages. In particular, it might be possible to provide better solutions where zero knot intervals appear adjacent to extraordinary vertices. In certain cases, multiple knots can reasonably reduce the level of continuity at singularities, as they always do so in regular regions. Particularly at high degrees, however, it is counter-intuitive for a solitary double knot to remove even C^1 continuity; in regular regions, increasing the multiplicity to k only reduces the continuity to C^{d-k}. By handling such configurations as a special case, we might be able to provide continuity guarantees even in the presence of multiple knots, as long as the multiplicity is not too high.

There might also be the potential for more extensive changes. Instead of constraining faces to occupy a rectangle in parameter space, Sederberg et al. [89] and Müller et al. [51] used a more general framework where each edge can be assigned a knot interval independently. Müller et al. [51] used the name *augmented faces* for faces associated with a non-rectangular parameter space. Future work could consider a generalisation of the local knot vectors (which are collected as shown in Figure 4.2) in order to incorporate augmented faces as part of NURBS-compatible

subdivision surfaces. With this generalisation in place, it might also be possible to incorporate extraordinary faces into odd-degree surfaces, or extraordinary vertices into the even-degree case.

In addition to extensions of the current representation, by improving the handling of multiple knots or including even degrees and augmented faces, there is also scope for greater analysis of the surfaces in their current form. It would be useful, for example, to investigate further the hypothesis (5.3) that I introduced in §5.3. If it holds, then this would become a useful heuristic for evaluating current subdivision schemes and designing new ones. If it proves to be unfounded, however, then we may learn how the current bounded-curvature solutions for NURBS-compatible subdivision schemes could be improved.

Finally there are applications for these schemes, such as the NURBS repair pipeline I described in §6.2, which are excellent candidates for future work. Testing NURBS-compatible subdivision on real-world data may highlight changes which would be necessary for the schemes to be useful in production, and this dissertation is certainly not the final word on NURBS-compatibility. I have shown, however, that NURBS and subdivision surfaces need not be seen as worlds apart, and that NURBS surfaces can be freed, in all their generality, from the topological restrictions that they have always held.

Bibliography

[1] U. H. Augsdörfer, T. J. Cashman, N. A. Dodgson and M. A. Sabin. Numerical Checking of C^1 for Arbitrary Degree Subdivision Schemes based on Quadrilateral Meshes. In *13th IMA Conference on the Mathematics of Surfaces*, E. R. Hancock, R. R. Martin and M. A. Sabin, editors, volume 5654 of *Lecture Notes in Computer Science*, pp. 45–54. Springer, 2009.

[2] U. H. Augsdörfer, N. A. Dodgson and M. A. Sabin. Tuning Subdivision by Minimising Gaussian Curvature Variation Near Extraordinary Vertices. *Computer Graphics Forum*, 25(3):263–272, 2006.

[3] U. H. Augsdörfer, N. A. Dodgson and M. A. Sabin. Removing Polar Rendering Artifacts in Subdivision Surfaces. *Journal of Graphics, GPU, and Game Tools*, 14(2):61–76, 2009.

[4] C. Bajaj, S. Schaefer, J. Warren and G. Xu. A subdivision scheme for hexahedral meshes. *The Visual Computer*, 18(5):343–356, 2002.

[5] A. A. Ball and D. J. T. Storry. Conditions for Tangent Plane Continuity over Recursively Generated B-Spline Surfaces. *ACM Transactions on Graphics*, 7(2):83–102, 1988.

[6] A. A. Ball and D. J. T. Storry. An Investigation of Curvature Variations Over Recursively Generated B-Spline Surfaces. *ACM Transactions on Graphics*, 9(4):424–437, 1990.

[7] L. Barthe and L. Kobbelt. Subdivision scheme tuning around extraordinary vertices. *Computer Aided Geometric Design*, 21(6):561–583, 2004.

[8] W. Boehm. Inserting new knots into B-spline curves. *Computer-Aided Design*, 12(4):199–201, 1980.

[9] W. Boehm. On the efficiency of knot insertion algorithms. *Computer Aided Geometric Design*, 2(3):141–143, 1985.

[10] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10(6):350–355, 1978.

[11] G. Chaikin. An algorithm for high speed curve generation. *Computer Graphics and Image Processing*, 3(4):346–349, 1974.

[12] E. Cohen, T. Lyche and R. Riesenfeld. Discrete *B*-Splines and Subdivision Techniques in Computer-Aided Geometric Design and Computer Graphics. *Computer Graphics and Image Processing*, 14(2):87–111, 1980.

[13] M. G. Cox. The Numerical Evaluation of B-Splines. *IMA Journal of Applied Mathematics*, 10(2):134–149, 1972.

[14] C. de Boor. On calculating with B-splines. *Journal of Approximation Theory*, 6(1):50–62, 1972.

[15] C. de Boor. Cutting corners always works. *Computer Aided Geometric Design*, 4(1–2):125–131, 1987.

[16] C. de Boor. *A Practical Guide to Splines*, volume 27 of *Applied Mathematical Sciences*. Springer-Verlag, revised edition, 2001.

[17] C. de Boor and K. Höllig. B-splines without divided differences. In *Geometric Modeling: Algorithms and New Trends*, G. Farin, editor, pp. 21–27. SIAM, 1987.

[18] P. de Faget de Casteljau. *Formes à pôles*, volume 2 of *Mathématiques et CAO*. Hermes, 1985.

[19] G. de Rham. Un peu de mathématique à propos d'une courbe plane. *Elemente der Mathematik*, 2:73–76, 89–97, 1946.

[20] T. DeRose, M. Kass and T. Truong. Subdivision surfaces in character animation. In *Proceedings of SIGGRAPH 98*, M. Cohen, editor, Computer Graphics Proceedings, Annual Conference Series, pp. 85–94. ACM, 1998.

[21] D. Doo and M. A. Sabin. Behaviour of recursive division surfaces near extraordinary points. *Computer-Aided Design*, 10(6):356–360, 1978.

[22] D. W. H. Doo. A subdivision algorithm for smoothing down irregular shaped polyhedrons. In *Proceedings: Interactive Techniques in Computer Aided Design, Palazzo Dei Congressi, Bologna, Italy, September 21–23*, pp. 157–165. IEEE, 1978.

[23] N. Dyn, M. S. Floater and K. Hormann. Four-point curve subdivision based on iterated chordal and centripetal parameterizations. *Computer Aided Geometric Design*, 26(3):279–286, 2009.

[24] N. Dyn, D. Levin and J. A. Gregory. A 4-point interpolatory subdivision scheme for curve design. *Computer Aided Geometric Design*, 4(4):257–268, 1987.

[25] N. Dyn, D. Levin and J. A. Gregory. A Butterfly Subdivision Scheme for Surface Interpolation with Tension Control. *ACM Transactions on Graphics*, 9(2):160–169, 1990.

[26] G. Farin. A History of Curves and Surfaces in CAGD. In *Handbook of Computer Aided Geometric Design*, G. Farin, J. Hoschek and M.-S. Kim, editors, pp. 1–21. Elsevier, 2002.

[27] G. Farin and N. Sapidis. Curvature and the Fairness of Curves and Surfaces. *IEEE Computer Graphics and Applications*, 9(2):52–57, 1989.

[28] Z. Galil and G. F. Italiano. Data structures and algorithms for disjoint set union problems. *ACM Computing Surveys*, 23(3):319–344, 1991.

[29] I. Ginkel and G. Umlauf. Loop subdivision with curvature control. In *Eurographics Symposium on Geometry Processing*, K. Polthier and A. Sheffer, editors, pp. 163–171. Eurographics, 2006.

[30] R. Goldman and J. Warren. An extension of Chaiken's algorithm to B-spline curves with knots in geometric progression. *CVGIP: Graphical Models and Image Processing*, 55(1):58–62, 1993.

[31] R. N. Goldman. Blossoming and knot insertion algorithms for B-spline curves. *Computer Aided Geometric Design*, 7(1–4):69–81, 1990.

[32] D. Gonsor and M. Neamtu. Subdivision Surfaces – Can they be Useful for Geometric Modeling Applications? Technical Report 01-011, The Boeing Company, 2001.

[33] J. A. Gregory and R. Qu. Nonuniform corner cutting. *Computer Aided Geometric Design*, 13(8):763–772, 1996.

[34] F. Holt. Toward a curvature-continuous stationary subdivision algorithm. *Zeitschrift für angewandte Mathematik und Mechanik*, 76:423–424, 1996.

[35] I. P. Ivrissimtzis, M. A. Sabin and N. A. Dodgson. On the Support of Recursive Subdivision. *ACM Transactions on Graphics*, 23(4):1043–1060, 2004.

[36] P. P. Joshi. *Minimizing Curvature Variation for Aesthetic Surface Design*. PhD thesis, University of California at Berkeley, 2008.

[37] K. Karčiauskas and J. Peters. Concentric tessellation maps and curvature continuous guided surfaces. *Computer Aided Geometric Design*, 24(2):99–111, 2007.

[38] K. Karčiauskas and J. Peters. On the curvature of guided surfaces. *Computer Aided Geometric Design*, 25(2):69–79, 2008.

[39] K. Karčiauskas and J. Peters. Adjustable speed surface subdivision. *Computer Aided Geometric Design*, 26(9):962–969, 2009.

[40] K. Karčiauskas, J. Peters and U. Reif. Shape characterization of subdivision surfaces–case studies. *Computer Aided Geometric Design*, 21(6):601–614, 2004.

[41] L. Kobbelt. $\sqrt{3}$-Subdivision. In *Proceedings of SIGGRAPH 2000*, K. Akeley, editor, Computer Graphics Proceedings, Annual Conference Series, pp. 103–112. ACM, 2000.

[42] J. M. Lane and R. F. Riesenfeld. A Theoretical Development for the Computer Generation and Display of Piecewise Polynomial Surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(1):35–46, 1980.

[43] E. T. Y. Lee. A note on blossoming. *Computer Aided Geometric Design*, 6(4):359–362, 1989.

[44] A. Levin. Modified subdivision surfaces with continuous curvature. *ACM Transactions on Graphics*, 25(3):1035–1040, 2006.

[45] C. Loop. Bounded curvature triangle mesh subdivision with the convex hull property. *The Visual Computer*, 18:316–325, 2002.

[46] C. T. Loop. Smooth Subdivision Surfaces Based on Triangles. *Master's thesis, University of Utah, Department of Mathematics*, Aug 1987.

[47] T. Lyche and K. Morken. Making the OSLO Algorithm More Efficient. *SIAM Journal on Numerical Analysis*, 23(3):663–675, 1986.

[48] W. Ma. Subdivision surfaces for CAD—an overview. *Computer-Aided Design*, 37(7):693–709, 2005.

[49] K. T. Miura and H. Masuda. Selective Non-Uniform Subdivision. In *Proceedings of 10th Pacific Conference on Computer Graphics and Applications*, pp. 457–459. IEEE Computer Society, 2002.

[50] H. P. Moreton. *Minimum Curvature Variation Curves, Networks, and Surfaces for Fair Free-Form Shape Design*. PhD thesis, University of California at Berkeley, 1992.

[51] K. Müller, L. Reusche and D. Fellner. Extended subdivision surfaces: Building a bridge between NURBS and Catmull-Clark surfaces. *ACM Transactions on Graphics*, 25(2):268–292, 2006.

[52] A. Myles and J. Peters. Bi-3 C^2 polar subdivision. *ACM Transactions on Graphics*, 28(3):#48, 1–12, 2009.

[53] J. Peters and U. Reif. The Simplest Subdivision Scheme for Smoothing Polyhedra. *ACM Transactions on Graphics*, 16(4):420–431, 1997.

[54] J. Peters and U. Reif. *Subdivision Surfaces*. Springer, 2008.

[55] J. Peters and L. J. Shiue. Combining 4- and 3-Direction Subdivision. *ACM Transactions on Graphics*, 23(4):980–1003, 2004.

[56] J. Peters and G. Umlauf. Computing curvature bounds for bounded curvature subdivision. *Computer Aided Geometric Design*, 18(5):455–461, 2001.

[57] L. Piegl and W. Tiller. Curve and surface constructions using rational B-splines. *Computer-Aided Design*, 19(9):485–498, 1987.

[58] L. A. Piegl and W. Tiller. *The Nurbs Book*. Springer, 1997.

[59] L. A. Piegl and W. Tiller. Filling n-sided regions with NURBS patches. *The Visual Computer*, 15(2):77–89, 1999.

[60] G. Plonka. Two-scale symbol and autocorrelation symbol for B-splines with multiple knots. *Advances in Computational Mathematics*, 3(1):1–22, 1995.

[61] H. Prautzsch. Freeform splines. *Computer Aided Geometric Design*, 14(3):201–206, 1997.

[62] H. Prautzsch. Smoothness of subdivision surfaces at extraordinary points. *Advances in Computational Mathematics*, 9(3):377–389, 1998.

[63] H. Prautzsch and Q. Chen. Analyzing Midpoint Subdivision. Preprint, 2009.

[64] H. Prautzsch and U. Reif. Degree estimates for C^k-piecewise polynomial subdivision surfaces. *Advances in Computational Mathematics*, 10(2):209–217, 1999.

[65] H. Prautzsch and G. Umlauf. A G^2-Subdivision Algorithm. In *Geometric Modelling, Dagstuhl, 1996*, G. Farin, H. Bieri, G. Brunnett and T. DeRose, editors, volume 13 of *Computing Supplement*, pp. 217–224. Springer, 1998.

[66] K. Qin and H. Wang. Eigenanalysis and Continuity of Non-Uniform Doo-Sabin Surfaces. In *Proceedings of Seventh Pacific Conference on Computer Graphics and Applications*, pp. 179–186. IEEE Computer Society, 1999.

[67] L. Ramshaw. Blossoming: A Connect-the-Dots Approach to Splines. Technical Report 19, Digital Systems Research Center, 1987.

[68] L. Ramshaw. Blossoms are polar forms. *Computer Aided Geometric Design*, 6(4):323–358, 1989.

[69] U. Reif. A unified approach to subdivision algorithms near extraordinary vertices. *Computer Aided Geometric Design*, 12(2):153–174, 1995.

[70] U. Reif. A Degree Estimate for Subdivision Surfaces of Higher Regularity. *Proceedings of the American Mathematical Society*, 124(7):2167–2174, 1996.

[71] U. Reif. TURBS—Topologically Unrestricted Rational B-Splines. *Constructive Approximation*, 14(1):57–77, 1998.

[72] R. F. Riesenfeld. *Application of B-spline Approximation to Geometric Problems of Computer Aided Design*. PhD thesis, Syracuse University, 1972.

[73] R. F. Riesenfeld. On Chaikin's algorithm. *Computer Graphics and Image Processing*, 4(3):304–310, 1975.

[74] M. A. Sabin. Cubic recursive division with bounded curvature. In *Curves and surfaces*, pp. 411–414. Academic Press Professional, Inc., 1991.

[75] M. A. Sabin. Eigenanalysis and Artifacts of Subdivision Curves and Surfaces. In *Tutorials on Multiresolution in Geometric Modelling*, A. Iske, E. Quak and M. S. Floater, editors, pp. 69–92. Springer, 2002.

[76] M. A. Sabin. Recent Progress in Subdivision: a Survey. In *Advances in Multiresolution for Geometric Modelling*, N. A. Dodgson, M. S. Floater and M. A. Sabin, editors, pp. 203–230. Springer, 2005.

[77] M. A. Sabin and L. Barthe. Artifacts in Recursive Subdivision Surfaces. *Curve and Surface Fitting: Saint-Malo*, pp. 353–362, 2002.

[78] M. A. Sabin and A. Bejancu. Boundary Conditions for the 3-Direction Box-Spline. In *10th IMA Conference on the Mathematics of Surfaces*, M. J. Wilson and R. R. Martin, editors, volume 2768 of *Lecture Notes in Computer Science*, pp. 244–261. Springer, 2003.

[79] M. A. Sabin, T. J. Cashman, U. H. Augsdörfer and N. A. Dodgson. Bounded Curvature Subdivision Without Eigenanalysis. In *12th IMA Conference on the Mathematics of Surfaces*, R. R. Martin, M. A. Sabin and J. R. Winkler, editors, volume 4647 of *Lecture Notes in Computer Science*, pp. 391–411. Springer, 2007.

[80] M. A. Sabin and N. A. Dodgson. A circle-preserving variant of the four-point subdivision scheme. In *Mathematical Methods for Curves and Surfaces: Tromsø 2004*, M. Dæhlen, K. Mørken and L. L. Schumaker, editors, Modern Methods in Mathematics, pp. 275–286. Nashboro, 2005.

[81] M. A. Sabin, N. A. Dodgson, M. F. Hassan and I. P. Ivrissimtzis. Curvature behaviours at extraordinary points of subdivision surfaces. *Computer-Aided Design*, 35(11):1047–1051, 2003.

[82] P. Sablonniére. Spline and Bézier polygons associated with a polynomial spline curve. *Computer-Aided Design*, 10(4):257–261, 1978.

[83] S. Schaefer and R. Goldman. Non-uniform Subdivision for B-splines of Arbitrary Degree. *Computer Aided Geometric Design*, 26(1):75–81, 2009.

[84] S. Schaefer, E. Vouga and R. Goldman. Nonlinear subdivision through nonlinear averaging. *Computer Aided Geometric Design*, 25(3):162–180, 2008.

[85] I. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions. *Quarterly of Applied Mathematics*, 4:45–99 (Part A), 112–141 (Part B), 1946.

[86] L. L. Schumaker. *Spline Functions: Basic Theory*. Cambridge University Press, 2007.

[87] T. W. Sederberg, G. T. Finnigan, X. Li, H. Lin and H. Ipson. Watertight Trimmed NURBS. *ACM Transactions on Graphics*, 27(3):#79, 1–8, 2008.

[88] T. W. Sederberg, J. Zheng, A. Bakenov and A. Nasri. T-splines and T-NURCCs. *ACM Transactions on Graphics*, 22(3):477–484, 2003.

[89] T. W. Sederberg, J. Zheng, D. Sewell and M. Sabin. Non-Uniform Recursive Subdivision Surfaces. In *Proceedings of SIGGRAPH 98*, M. Cohen, editor, Computer Graphics Proceedings, Annual Conference Series, pp. 387–394. ACM, 1998.

[90] J. Stam. Exact evaluation of Catmull-Clark subdivision surfaces at arbitrary parameter values. In *Proceedings of SIGGRAPH 98*, M. Cohen, editor, Computer Graphics Proceedings, Annual Conference Series, pp. 395–404. ACM, 1998.

[91] J. Stam. On subdivision schemes generalizing uniform B-spline surfaces of arbitrary degree. *Computer Aided Geometric Design*, 18(5):383–396, 2001.

[92] J. Stam and C. Loop. Quad/Triangle Subdivision. *Computer Graphics Forum*, 22(1):79–85, 2003.

[93] I. F. Stewart and A. R. Foisy. Arbitrary-degree subdivision with creases and attributes. *Journal of Graphics Tools*, 9(4):3–17, 2004.

[94] G. Taubin. A Signal Processing Approach To Fair Surface Design. In *Proceedings of SIGGRAPH 95*, R. Cook, editor, Computer Graphics Proceedings, Annual Conference Series, pp. 351–358. ACM, 1995.

[95] W. Tiller. Rational B-splines for curve and surface representation. *IEEE Computer Graphics and Applications*, 3(6):61–69, 1983.

[96] L. Velho and D. Zorin. 4–8 Subdivision. *Computer Aided Geometric Design*, 18(5):397–427, 2001.

[97] K. J. Versprille. *Computer-Aided Design Applications of the Rational B-spline Approximation Form*. PhD thesis, Syracuse University, 1975.

[98] E. Vouga and R. Goldman. Two blossoming proofs of the Lane-Riesenfeld algorithm. *Computing*, 79(2):153–162, 2007.

[99] J. Wallner and N. Dyn. Convergence and C^1 analysis of subdivision schemes on manifolds by proximity. *Computer Aided Geometric Design*, 22(7):593–622, 2005.

[100] H. Wang, K. Qin and R. Kikinis. Exact Evaluation of NURSS at Arbitrary Parameter Values. In *Proceedings of the IASTED International Conference on Computer Graphics and Imaging*, pp. 169–174, 2000.

[101] H. Wang, K. Qin and H. Sun. Evaluation of non-uniform Doo-Sabin surfaces. *International Journal of Computational Geometry and Applications*, 15(3):299–324, 2005.

[102] J. Warren. Binary subdivision schemes for functions over irregular knot sequences. In *Mathematical Methods for Curves and Surfaces*, M. Dæhlen, T. Lyche and L. L. Schumaker, editors, pp. 543–562. Vanderbilt U.P., 1995.

[103] J. Warren and H. Weimer. *Subdivision Methods for Geometric Design*. Morgan Kaufmann, 2001.

[104] D. Zorin. Constructing Curvature-continuous Surfaces by Blending. In *Symposium on Geometry Processing*, A. Sheffer and K. Polthier, editors, pp. 31–40. Eurographics Association, 2006.

[105] D. Zorin and P. Schröder. A unified framework for primal/dual quadrilateral subdivision schemes. *Computer Aided Geometric Design*, 18(5):429–454, 2001.

[106] D. Zorin, P. Schröder, T. DeRose, L. Kobbelt, A. Levin and W. Sweldens. Subdivision for Modeling and Animation. ACM SIGGRAPH Course Notes, July 2000.

[107] A. Zulti, A. Levin, D. Levin and M. Teicher. C^2 subdivision over triangulations with one extraordinary point. *Computer Aided Geometric Design*, 23(2):157–178, 2006.