Universität Rostock Traditio et Innovatio

Generalized B-spline surfaces for ship hull form representation and modeling

Dissertation

to attain the degree of

Doktor-Ingenieur (Dr.-Ing.)

at the

Faculty of Mechanical Engineering and Marine Technology

of the

University of Rostock

submitted by

Sebastian H. Greshake

born on November 24, 1987

in Essen, Germany

Rostock, 2021

https://doi.org/10.18453/rosdok_id00002984

Reviewer:

Prof. Dr.-Ing. Robert Bronsart
Chair of Ship Design
Faculty of Mechanical Engineering and Marine Technology
University of Rostock

Prof. Dr.-Ing. habil. Nikolai Kornev
Chair of Modeling and Simulation
Faculty of Mechanical Engineering and Marine Technology
University of Rostock

Submission: July 23, 2020
Colloquium: January 8, 2021

This page intentionally left blank.

Abstract

Ship hull form modeling is currently based on the interpolation of curve networks. This method is based on a network of curves which is smoothly interpolated by tensor-product B-spline patches to obtain a surface representation of the hull form. It is, however, well-known that interpolation tends to unwanted oscillations and this makes hull form modeling and especially fairing of hull surfaces a time-consuming task.

Tensor-product B-splines are designed as a smooth approximation of the control mesh they are defined on. While the clear link between the control mesh and the surface provides a simple, but yet powerful tool for modeling, it is only rarely used for hull form modeling in practice. The reason is that tensor-product B-splines are limited to four-sided surfaces and complex surfaces such as hull forms have to be decomposed into sufficiently simple patches to represent an overall smooth surface. This introduces complex constraints to the control points which are difficult to handle and may be circumvented during modeling by the above-mentioned interpolation of curve networks. In contrast to a common belief in the community, however, it is analyzed that the dominance of curve-based hull design is a consequence of the need to avoid these constraints rather than just reflecting the inherent nature of hull design as being assumed by most other authors in the domain.

Generalized B-splines are introduced to overcome the limitation of tensor-product B-splines to four-sided surfaces. They behave just like conventional B-spline surfaces, but can represent surfaces of any complexity on a single unconstrained control mesh. The underlying theory of generalized B-splines is comprehensively described and the implementation of this surface representation is reviewed. Generalized B-splines are applied in the context of ship hull form modeling and examples are presented for a range of different vessel types. It is shown that the structure of the control mesh and the principles to work with control points are basically the same as for curve networks and the control mesh therefore integrates equally well into hull form modeling. While the curve-based design is usually limited to tangent continuous surfaces, a generalized B-spline surface is almost curvature continuous. Finally, a method is presented which converts a generalized B-spline into tensor-product B-spline patches once the modeling is finished to integrate this surface representation into the ship design process. The good continuity characteristics are maintained upon conversion and therefore the result clearly outperforms the curve-based design in terms of the modeling performance and especially quality of the generated surfaces.

This page intentionally left blank.

Contents

1. Introduction **1**

1.1. Motivation 1
1.2. Scope 3
1.3. Overview 3

2. Literature review **5**

2.1. Hull form modeling 5
2.2. B-spline surfaces 13

3. Tensor-product B-spline surfaces **23**

3.1. Definition 23
3.2. Important characteristics 26
3.3. Complex surfaces 29

4. Generalized B-spline surfaces **33**

4.1. Introduction 33
4.2. Assumptions 34
4.3. Subdivision 37
4.4. Domain 44
4.5. Patches 50
4.6. Continuity 58
4.7. Implementation 60
4.8. Example 62
4.9. Bibliographical notes 64

5. Hull surface representation **65**

5.1. Requirements 65
5.2. Generalized B-spline surface 67
5.3. Discussion 72
5.4. Bibliographical notes 73

6. Ship hull form modeling **75**

6.1. Structure of the control mesh 75
6.2. Feature curves 77
6.3. Modeling with control points 77

6.4. Irregular control points 81
6.5. Fairing 85
6.6. Design process 85
6.7. Example vessels 87
6.8. Bibliographical notes 89

7. Integration into early ship design **91**
7.1. Integration strategy 91
7.2. Fitting algorithm 92
7.3. Results 94
7.4. Bibliographical notes 98

8. Conclusion **99**

Bibliography **101**

A. Container vessel **111**

B. Bulk carrier **117**

C. RoRo vessel **123**

D. Barge **129**

E. Research vessel **135**

F. Naval vessel **141**

G. Remarks on smoothness **147**

H. Knot refinement **153**

I. Diagonalization **159**

List of Figures

1.1. Hull form design 2

2.1. Curve-based hull form design 6
2.2. Form parameter hull form design 8
2.3. B-spline surfaces on irregular control meshes 14
2.4. Subdivision as a method to refine polygon meshes 16

3.1. Tensor-product B-spline function 24
3.2. Local modification of control points 27
3.3. Regularity of tensor-product B-spline control meshes 28
3.4. Decomposition of complex surfaces into B-spline patches 29
3.5. Curve-based B-spline modeling 31

4.1. Generalized B-spline function 34
4.2. Isotropic degree of generalized B-splines 35
4.3. Non-uniform generalized B-splines 36
4.4. Knot refinement for tensor-product B-splines 39
4.5. Example and notation of subdivision rules 41
4.6. Topological subdivision for odd and even degrees 43
4.7. Isolation of irregularities based on subdivision 44
4.8. Cells of a generalized B-spline domain 45
4.9. Terminology for the domain 46
4.10. Examples of generalized B-spline domains 47
4.11. Correspondence of control mesh and domain 49
4.12. Definition of quadrants 52
4.13. Subdivision to decompose an irregular quadrant into regular segments 53
4.14. Subdivision matrices for irregular quadrants 54
4.15. Identification of regular segments 55
4.16. Theory of generalized B-splines 59
4.17. Example for a cubic generalized B-spline surface 63

5.1. Geometrical features of ship hull forms 66
5.2. Subdivision algorithm for a cubic generalized B-spline surface 69
5.3. Basis functions of the subdivision algorithm 71

6.1. Typical structure of the control mesh 76
6.2. Handling feature curves based on the control mesh 78
6.3. Hull form modeling with control points 80
6.4. Definition of the main section 81
6.5. Impact of irregular control points on surface quality 82
6.6. Application of irregular control points 83
6.7. Representation of a bulbous bow 84
6.8. Representation of a stern bulb 85
6.9. Fairing of a hull form 86

7.1. Notation of the least-squares fitting algorithm 94
7.2. Conversion of a hull form to tensor-product B-splines 95
7.3. Conversion of a bulbous bow to tensor-product B-splines 97

List of Symbols

General

k	Number of subdivisions
i, j	Indices
n_e	Edge valence
n_f	Face valence
n_v	Vertex valence
C^m	Parametric continuity
G^m	Geoemtric continuity
$\mathbf{x}$	B-spline surface
$\mathbf{q}_{ij}$	Control point
$\mathbf{Q}$	Control points of $\mathbf{x}$

Tensor-product B-splines

u, v	Parameter values
u_i, v_i	Knots
U, V	Knot vectors
p, q	Degree in u-direction, degree in v-direction
Σ	Domain of tensor-product B-spline
$\mathbf{S}$	Domain of a generalized B-spline surface
$N_{i,p}$	Univariate B-spline basis function in u-direction
$N_{j,q}$	Univariate B-spline basis function in v-direction
$N_{i,j}$	Bivariate B-spline basis function
B	Vector of bivariate B-spline basis functions of $\mathbf{x}$

Generalized B-splines

$\mathbf{x}_i$	Patch of a generalized B-spline surface
$\mathbf{x}_c$	Central point
$\mathbf{S}$	Domain of a generalized B-spline surface
Σ_i	Domain of $\mathbf{x}_i$
$\mathbf{Q}_i$	Control points of $\mathbf{x}_i$
$\mathbf{Q}_i^k$	k-times subdivided control points of $\mathbf{x}_i$
$\phi_n(u, v)$	Precomputed basis function
$\hat{\mathbf{Q}}_{i,m}$	Precomputed vector of control points
$\hat{\mathbf{q}}_j$	Precomputed control points

Quadrants

$\mathbf{x}_{i,m}$	Quadrant of a patch $\mathbf{x}_i$ ($m = 1, 2, 3, 4$)
$\Sigma_{i,m}$	Domain of mth quadrant of $\mathbf{x}_i$
$\mathbf{Q}_{i,m}$	Control points of mth quadrant of $\mathbf{x}_i$
$\mathbf{Q}_{i,m}^k$	k-times subdivided control points of mth quadrant of $\mathbf{x}_i$

Segments

$\mathbf{x}_{i,m,n}^k$	Segment ($n = 1, 2, 3$)
Ω_n^k	Domain of nth segment
B_n	Vector of basis functions of nth segment
M_n	Picking matrix for nth segment

Subdivision

A	Subdivision matrix
A_k	Extended subdivision matrix
λ_i	Eigenvalues

$\mathbf{x}_i$	Eigenvectors
Λ	Diagnonal matrix of eigenvalues
X	Matrix of eigenvectors
α, β, ω	Weigths to compute control points
$\mathcal{M}^k$	k-times subdivided control mesh
$\mathcal{V}_i$	Vertices of a control mesh
$\mathcal{E}_i$	Edges of a control mesh
$\mathcal{F}_i$	Faces of a control mesh
$\mathbf{v}_i$	Vertex points, i.e. control point computed for a vertex $\mathcal{V}_i$
$\mathbf{e}_i$	Edge points, i.e. control point computed for an edge $\mathcal{E}_i$
$\mathbf{f}_i$	Face points, i.e. control point computed for a face $\mathcal{F}_i$

1 Introduction

1.1. Motivation

The hull form affects many design decisions made throughout the ship design process and influences many of the vessel's characteristics as illustrated in Figure 1.1. The hull form is initially an important constraint for the capacity of a vessel. Taking the example of a container vessel, it limits the size of the holds and therefore the number of containers which can be stored in the holds, but it also limits the deck size and is therefore an important constraint for the number of containers which can be stored on deck as well. A container vessel is naturally a rather simple example compared to more sophisticated vessel types such as cruise vessels or naval ships, but for any ship type the availability of spaces is a major design issue and largely dependent on the hull form. In addition to the capacity, the hull form has a major impact on the operating costs of a vessel. The resistance and the propulsion characteristics depend on the hull form. Thus the power required to sail with a given speed and finally the fuel costs are influenced by the hull form, with the latter being the predominant share of the operating costs for many vessel types. Next to the operating costs, the hull form also has a bearing on the building costs. The production of curved plates is more expensive compared to flat plates. Therefore the building costs can be optimized when curved regions of the hull forms are minimized in favor of flat regions. Naturally, this compromises the operating costs and the designer has to make a reasonable compromise between low operating costs and low building costs. Finally, the hull form is important for the safety of a vessel. The intact stability essentially depends on two items: the hull form and the weight distribution. Loosely speaking, the hull form causes the buoyancy which is counteracted by the weight. Stability ensures that the buoyancy provided by the hull form always enables a safe ship operation which takes static as well as dynamic aspects such as ship motions, waves, and parametric roll into account. The same holds for the damage stability, but additionally the internal subdivision of the hull is taken into account. Therefore one of the most important safety measures for a vessel, the stability, depends on the hull form.

The previously discussed characteristics of a vessel concern the main ship design questions: the capacity, respectively the question whether the vessel does fit to the intended operation profile, the costs, and the safety. All of those questions are at the core of the early ship design. This design phase begins with the customer's inquiry and is followed by the basic design and finally the detail design. The basic design focuses on the approval of the design by the class

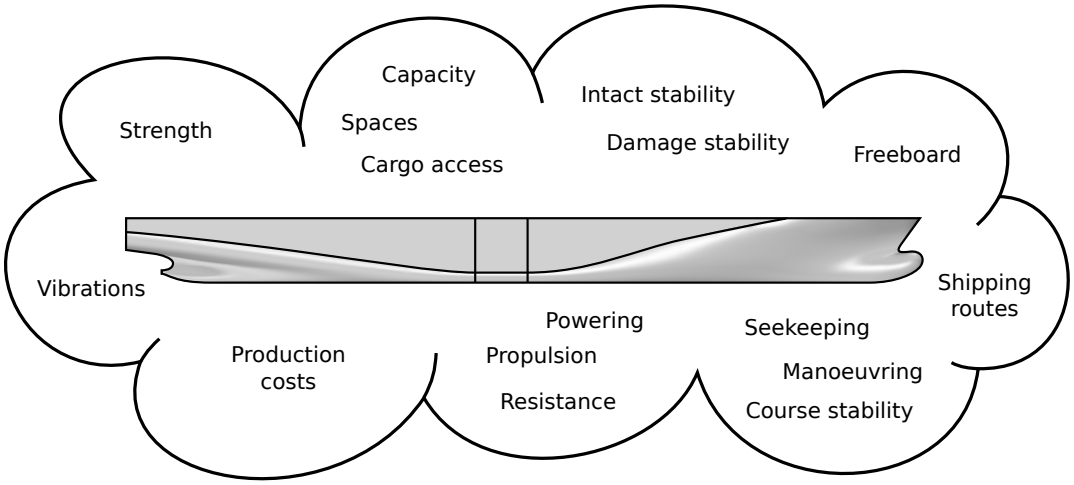


Figure 1.1.: A non-exhaustive list of characteristics of a vessel which are influenced by the hull form. All three major areas in ship design, namely safety, costs, and capacity, are affected by the hull form. (Figure based on the lecture notes on Ship Design held by Robert Bronsart).

society and the detail design is focusing on the production. The early design, however, takes care about the vessel's concept. It is usually the shortest design phase, but with the majority of the vessel's characteristics as well as operation and building costs determined at this stage it can be considered as the most important design stage. While the concept of the ship is largely fixed in later design phases and changes only to a minor extent at best, the concept in the early design phase develops rapidly. With this nature of the early ship design phase at hand, the importance of hull design is evident.

A ship hull form is usually represented by B-spline surfaces. This surface representation is the current standard for the representation of curved surfaces in computer-aided design and they are successfully employed to represent all types of objects ranging from simple basic geometries to complex surfaces such as automobile bodies. From a theoretical point of view, they provide the required surface quality for ship design, but to arrive at a sufficient quality in practice, it needs much time and designers who are experienced in handling this surface type. Both are a rare assets in the early ship design phase. Therefore in this stage the quality is often sacrificed in favor of the time spend on hull form modeling.

B-spline surfaces are naturally modified based on control points which are assembled into a control mesh. As long as only a single B-spline surface is considered, the control mesh is an easy and fast way to define a free-form surface and results in a good quality. Complex surfaces such as hull forms, however, are composed of several B-spline surfaces. Maintaining smooth transitions between adjacent B-spline patches based on the control mesh is a difficult task and requires significant experience with B-spline surfaces and a profound background on

B-spline mathematics. Therefore state-of-the-art hull form modeling software usually does not rely on the control mesh and instead prompts the designer for a network of curves to describe the hull form. The software then takes care to generate B-spline surfaces which interpolate the given curves and join smoothly. This method is promoted for several decades and implemented in a variety of software tools. However, the practical experience gained over many years shows the weaknesses of this approach. While a preliminary outline of a hull form is indeed efficiently created, it still takes too much time to achieve a satisfactorily quality, especially in the early ship design phases. The importance of hull design in the early ship design and the last-mentioned limitations of the current hull design software are the motivation for this work.

1.2. Scope

In the field of naval architecture, hull design refers to two related challenges. On the one hand, this term covers the process to generate a geometrical representation of the hull, referred to as *hull form modeling* in this thesis. The hull geometry is used at other design stages to predict for example various performance characteristics of the vessel or to prepare the required production data. In contrast, hull design refers to the definition of a hull form which meets the various requirements such as a sufficient capacity, minimal operating and production costs, or good safety characteristics. Naturally, just the problem to minimize the operating costs of a hull form constitutes a full-fledged field of research on its own. However, this second reading of hull design is beyond the scope of this thesis.

This thesis is solely dedicated to the first reading of hull design. The goal is to simplify the geometrical modeling of the hull form during the early design phase and therefore to overcome the limitations faced with state of the art hull form modeling software. The curve-based approach is questioned and the latest research is reviewed in order to consider suitable alternatives to this method. The requirements of the early ship design phase are essential and this implies that the method going to be developed has to be fast enough to keep pace with the rapidly evolving concept of the vessel at this stage. Furthermore, the resulting geometrical representation of the hull form has to be compatible to downstream or parallel design processes.

1.3. Overview

This thesis is organized as follows:

- In Chapter 2 the research in the field of hull form modeling is reviewed. It is pointed out that this thesis is the first to seriously consider other methods than curve-based modeling in hull design based on a comprehensive scientific study. In addition the research on

subdivision surfaces is introduced to the reader. It is shown that subdivision surfaces originate from the idea to overcome the limitation of B-splines to regular control meshes and thus four-sided surfaces in order to simplify B-spline surface modeling.

- In Chapter 3 the basic theory and the relevant properties of B-spline surfaces are introduced. In particular, it is demonstrated how the limitation of B-splines to four-sided surfaces calls for the curve-based modeling prevalent in hull design. It is argued why this is first and foremost a necessity caused by the mathematical constraints of the underlying surface representation rather than reflecting the inherent nature of hull design as being assumed by most other authors.

- In Chapter 4 the advances of the subdivision theory of the last two decades are utilized in order to develop the notion of generalized B-splines. This is an important part of this thesis because the concept of generalized B-splines clarifies how the power of subdivision surfaces is applicable in the context of engineering, where subdivision surfaces are usually considered as being limited to the domain of computer artists and entertainment.

- In Chapter 5 the requirements of a surface representation for hull form design are worked out and a generalized B-spline surface is constructed which meets at least a significant subset of those requirements.

- In Chapter 6 shows how hull form modeling based on the potentially irregular control mesh of generalized B-spline fits to the hull design process. In particular it is described how the control mesh fits to the typical hull design process and how it outperforms the curve-based modeling in terms of the quality of the results, speed and robustness of the method. Consequently this chapter contains the primary result of this thesis as it refutes the long-lived misconception in hull design that the interpolation of curve networks best reflects the needs of hull design.

- In Chapter 7 finally shows how a generalized B-spline surface is converted to classical B-spline surfaces which are the standard surface representation used for data exchange in ship design. Thus, the final chapter of the thesis illustrates how generalized B-splines are integrated into the ship design process.

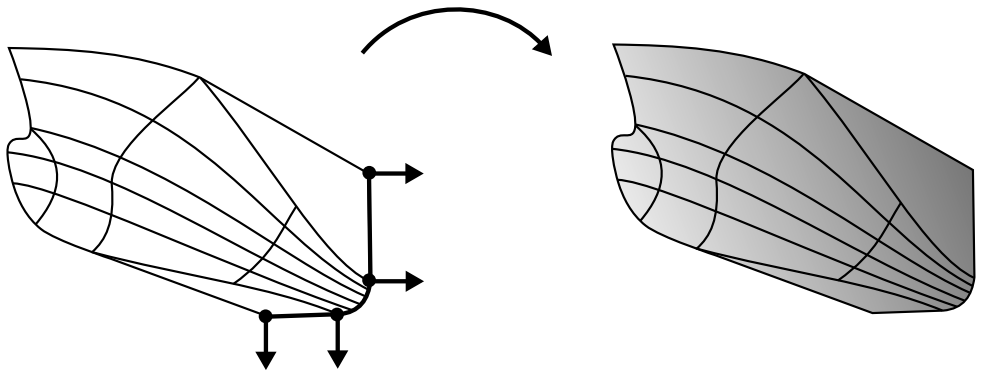
2 Literature review

2.1. Hull form modeling

State of the art

In order to classify this thesis with respect to other research activities, this section first introduces the state of the art in hull form modeling which then serves as the basis to review the current research activities in the next section. The state of the art in hull form representation and modeling is thoroughly reviewed in Nowacki et al. [76]. Despite being introduced more than two decades ago, the concepts presented in this reference are still considered as be up to date as for instance the recent overview on computer-aided ship design given by Nowacki [75] shows.

B-spline surfaces are the standard for hull form representation as well as they are the standard representation for curved surfaces in computer-aided design and other disciplines where a mathematical precise representation matters. A comprehensive introduction to B-spline surfaces is given by Piegl and Tiller [89] or by Rogers [96] where the former is the reference source on this topic and the author of the latter is among the first to employ B-spline surfaces in the context of hull form modeling. There are different B-spline surface variants. The most general variant are non-uniform rational B-splines (NURBS) which is also the standard supported by most computer-aided design (CAD) systems used in practice today. B-spline surfaces are mathematically constructed from the tensor-product of two B-spline curves. Therefore they are precisely called tensor-product B-spline surfaces. This terminology is rather unusual, even in literature specifically dedicated to this topic, because B-spline surfaces are almost exclusively constructed in this way. Thus the explicit reference on the tensor-product is often omitted for the sake of simplicity. Within the scope of this thesis it is, however, necessary to include an explicit reference to the tensor-product when B-splines are discussed because it pursues another construction to define B-spline surfaces of arbitrary complexity which is referred to as generalized B-splines. Conventional tensor-product B-splines are limited to four-sided surfaces. As an option, one side of a tensor-product B-spline surface may collapse by constructing it of zero length. This geometrically looks like a three-sided surface, but remains a four-sided surface from a mathematical point of view. This limitation is a challenge in the field of hull form modeling because it is mostly impossible to represent a hull form only with a single four-sided surface. A hull surface is therefore usually composed of several tensor-product B-splines which altogether represent an overall smooth



(a) curve network (b) surface patches

Figure 2.1.: Principles of the curve-based design. The hull designer specifies a network of curves which is interpolated by a number of surface patches. On the left side the curve network is shown. The designer primarily works with points which are interpolated by the curves as illustrated for the main section curve. To precisely manipulate the curve's shape, the designer may specify additional constraints such as tangents or discontinuities. One patch is generated for each loop of the curve network as illustrated on the right side with highlighted patch boundaries. The patches are assumed to interpolate the curves and join continuously with their neighbors in order to represent an overall smooth surface.

surface. The term *smoothness* is defined precisely in Appendix G and the reader may refer to this material for clarification. To emphasize that a single surface represents just a subset of the entire hull surface in this setup, they are referred to as patches throughout this thesis.

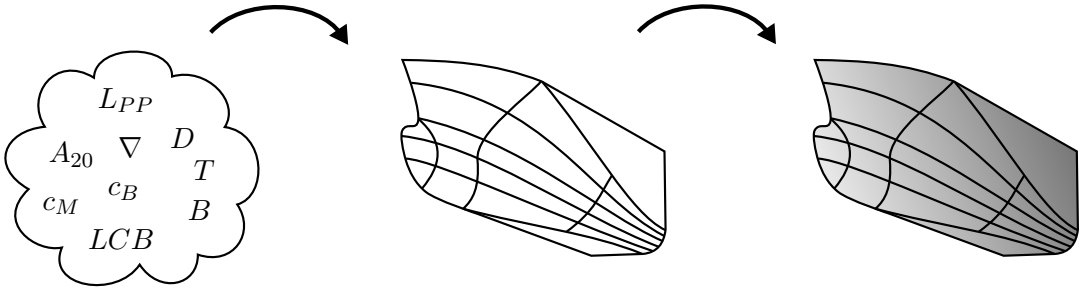
In the following, the explanation of the underlying surface representation is extended to include the methods to actually model a hull surface. In general two methods are differentiated: the curve-based design and the form parameter design. In the curve-based design, the hull designer specifies a network of curves in order to describe the hull surface. To define the network the designer is primarily working with points which are going to be interpolated by the curves. To precisely manipulate a curve's shape, the designer may formulate additional constraints such as prescribed tangents or even discontinuities at certain curve points. Finally, a number of tensor-product B-spline patches is automatically constructed, having to provide watertight transitions and tangent continuity between the patches. Refer to Figure 2.1 for a graphical illustration of the curve-based design.

One patch is generated for each loop of the curve network. A constraint resulting from the limitation of tensor-product B-splines to four-sided surfaces is that the loops of the curve network have to be defined by four (three) curve segments in order to be filled up with a patch. Some methods may accept arbitrary networks and automatically perform a preprocessing step which introduces further curves to the network in order to meet this constraint, but other methods require this constraint's consideration during modeling. The shape of the curves and

patches optionally is optimized by minimizing a fairness criterion. In simple words a fairness criterion is an integraly parameter that quantifies how much the surfaces deviates from the ideal surface and the surface is optimized by minimizing this parameter. A typical example for this kind of fairness optimization is to consider the surface as a thin plate and to minimize the deformation energy based on this model. The reader may refer to [107] for an overview of various fairness criteria and their application in the context of hull form modeling.

The main component of any curve-based modeling method is to construct a number of tensor-product B-spline patches to interpolate the given network of curves smoothly. This is a well-studied problem which is not limited to hull form modeling. The numerous methods presented in the literature essentially are classified by two characteristics: On the one hand the admissible topology of the curve network, and on the other hand the geometric continuity at which neighboring patches join each other. For a strictly regular network of curves the problem is easily tackled by Gordon surfaces [34]. Only a single surface is constructed interpolating all curves of the network at once. This represents a global interpolation method, which naturally yields good continuity characteristics, but changing one curve will always change the entire surface. This behavior is not desirable in the context of hull form modeling. Moreover it is not possible to generalize the method to irregular curve networks, i.e. curve networks where more than two curves intersect in a common node, as they are frequently encountered in practice. An alternative are Coons patches [25] where one patch is constructed for each loop of the curve network, representing a local interpolation method because changing a curve affects only the incident patches. At the first glance this seems to provide a reasonable solution for the interpolation of curve networks, but in practice it is difficult to construct tangent continuous patches even for strictly regular networks due to potentially incompatible twists at the patch corners. A method to construct patches for at least moderately irregular curve networks which join with tangent continuity is given by Sarraga [98]. The method allows to interpolate networks where three, four, or five curve may intersect in a common node, but the corner twist is forced to be zero and thus undesired flat spots may occur. Other methods drop the zero-twist assumption or allow an arbitrary number of curves to intersect in a common node of the network, see for example Peters [80] or Liu and Sun [62]. However, all methods are limited to tangent continuity, but in the field of hull design generally curvature continuity is required to consider the result as smooth. This problem is addressed in Ye and Nowacki [111] and Ye [110], but the methods are rather complex and have not gained wide acceptance in the field of hull form modeling. Finally, the method given by Westgaard and Nowacki [107] combines the curve interpolation methodology with the minimization of a fairing function to construct hull surfaces featuring tangent continuity across patch boundaries what fairly well represents the state of the art in curve-based hull design.

The form parameter design basically involves the same steps as the curve-based design. However, instead of directly manipulating the curve network based on points and other geometrical constraints, the designer specifies a number of principal parameters which describe



a) parameters b) curve network c) surface patches

Figure 2.2.: Principles of form parameter design. The designer specifies a number of parameters which describe the hull form such as the main dimensions or integral parameters like the displacement. The curve network is automatically derived from the parameters, and the surface patches are finally constructed from the curve network using the same methods as the curve-based hull form design.

the hull form. Examples are the hull's main dimensions or integral parameters like the hull's displacement. The curve network is derived from the parameters, and the patches are finally constructed from the curve network using the same methods as discussed above, refer to Figure 2.2 for clarification. The form parameter design is used to quickly generate a hull form in early design stages and to avoid the time-consuming process of manually defining a curve network.

Technically, the core of the form parameter design method is to translate the form parameters into geometrical constraints from which the curves are generated. This is a complex matter and naturally depends on the type of hull being modeled as well as the number of parameters being considered. The basics of this method are given in Nowacki et al. [76] and the work of Kim [47] successfully implemented the method in the context of complex hull forms. While the form parameter design method potentially saves a lot of time in early design stages where only a preliminary hull form is required, the method is generally not applicable in later design stages where a detailed design of the hull form is required, see also the discussion of Bole and Lee [14]. Whenever precise control of the hull surface is required, the curve-based design remains the method of choice. Other applications of the form parameter design are optimization studies. The parameters describing the hull form are systematically varied in order to find an optimal solution with reference to some optimization criteria. This is particularly promoted by Harries [38] and a framework is proposed for parameteric hull surface generation and systematic design explorations in the context of hydrodynamic optimizations.

Recent advances

Despite being the standard representation of curved surfaces in computer-aided design, tensor-product B-spline surfaces are far from being the ideal representation of hull surfaces. The challenges related to using tensor-product B-splines in the context of hull form representation and modeling are recently analyzed by Sharma et al. [102]. The authors identify the limitation of tensor-product B-splines to regular control meshes and thus four-sided surfaces as the reason why hull form modeling is still a laborious task and the surface quality in practice often is not sufficient. Four-sided surfaces are identified as being incompatible to the required complex hull form topology. Besides that, a lack of surface quality may easily lead to a failure of downstream analysis methods that require the hull form as an input. The latter is for example considered in detail by Bronsart et al. [17, 18, 19] and the studies provide evidence of the difficulties to manage those problems in practice. Sharma et al. [102] finally conclude that addressing the limitation of B-spline surfaces to regular control meshes and four-sided surfaces will be the key to improve hull form modeling in the future. This insight is the pivot of this thesis and shows both the motivation of this work as well as the necessity of this research. The study is complemented by Koelman and Veelo [49] with the same conclusions drawn by the authors, but additionally the authors briefly introduce potential solutions and suggest further research directions. In particular, they propose to investigate alternative surface representations in the context of hull form representation and modeling.

That hull form modeling is still a laborious task is rarely questioned, but the same is not true with respect to tensor-product B-splines. The research of the last two decades since Nowacki et al. [76] formulated the state of the art in hull form modeling mostly builds on top of this surface representation and just aims to develop better methods to handle the limitations of this surface representation in practice. Some recent publications, however, indicate that there is now a bit of momentum to tackle the root of the problem and to question the surface representation being used for hull form representation itself.

A method to improve the curve-based modeling built on top of tensor-product B-spline surfaces is given by Bole [9], also briefly described in Bole and Lee [14] and only recently resumed in Bole [13]. The proposed method combines the curve-based design with the form parameter design. It is based on the insight that the curve-based modeling provides the means to define a hull surface precisely, but usually takes too much time to be effectively employed in early design stages where only a preliminary hull surface is required. On the other hand, the form-parameter design represents a fast method suited to the needs of the early design stages with the hull surface almost instantaneous generated, but the method lacks the precise control on the surface's details required in later design stages. Both methods are brought down to a common denominator based on a hierarchical approach to hull form modeling. The hull surface is broken down into regions with the boundaries specified by the designer in terms of curves. Those curves do not only represent the boundaries of the regions, but also define their

connectivity, for instance that two regions shall meet at a knuckle or are considered to join smoothly. Effectively this represents a topological model of the hull surface which is referred to as form topology by the author. The method further includes different types of control curves and constraints which are tailored to the needs of hull form modeling. They do not only hide the details of B-spline mathematics, but allow to define geometric features of a hull form in terms of hull characteristics. The latter effectively introduces the power of parameter-based design to the curve-based modeling. The application of the method in the context of early design is specifically discussed in Bole et al. [15] and the interactive transformation of the hull form based on this method is described in Bole [10]. However, the method only changes the user interface of curve-based modeling from a rather geometric perspective of points, curves, and surfaces to a more technical perspective of hull form features which represent the design intent, see the discussion in Bole [13]. The technical implementation of this method, which is to have a curve network interpolated by a number of tensor-product B-spline patches altogether representing the hull surface, stays unchanged.

In Bole [11] the issues related to the surface quality obtained with the curve-based design are considered. The difficulties to ensure a sufficient surface quality are identified as being strongly related to the interpolation of curves with tensor-product B-splines and the challenge to define smooth patches in the presence of irregular curve networks. Interestingly, the idea to interpolate the curve network with surface patches is dropped and just sections, waterlines, and buttocks are created from the given curve network similar to the traditional lofting, effectively resulting in a lines representation of the hull form. The study analyzes that the method features a better performance when it comes to interactive modeling and is likely to result in better results in terms of fairness. Moreover, the modeling is simplified because the result is significantly less dependent on the actual topology of the curve network and thus the designer is relieved from the need to take the topology into account during modeling in order to achieve good results. Because a lines-based representation of the hull form is generally not sufficient for downstream processes, the author finally proposes to fit surface patches to the curves rather than strictly interpolating the data. This point is also discussed in Bole [12] where also the problem of poor surface quality is addressed and good results based on fitting rather than interpolation are reported.

Another contribution to the field of curve-based design is made by Koelman [50] with the essential results are briefly introduced in Koelman et al. [51]. Considering the conventional curve-based design methodology, the surface patches are constructed to interpolate the given curves and to join each other smoothly, but they remain entirely independent of each other. When one patch is subsequently modified, for instance in another modeling software, this does not affect the other patches because the information on how neighboring patches should connect to each other is neither kept with the patches nor considered within the patch's surface representation. Koelman [50] proposes to maintain this information within a topological data structure. A standard boundary representation (B-rep) is employed which includes vertices,

edges, faces, and a shell. The entities store links to other topological elements in order to represent their connectivity, but some entities are also linked to geometries with the edges being associated to curves and the faces being associated to the patches. The shell is finally the topological representation of the entire hull surface. The author refers to this concept as the hybrid model for ship hull representation (H-rep) and the practical application of this concept is discussed in Koelman [48]. While beyond question being a robust model for the patch-based hull form representation, it remains unclear how the curve-based modeling profits from this model as being claimed by the author. The main challenge related to the curve-based design is to construct surface patches which interpolate the given curves smoothly and to optimize the overall fairness of the hull surface and not to internally manage the topology of the curve network and the surface patches. The problems of curve interpolation and fairing are indeed considered in Koelman [50], but the presented solutions represent basically the state of the art as for example reported by Nowacki et al. [76] and the method is therefore unlikely to perform much different to the other alternatives.

The interpolation of curve networks with tensor-product B-spline patches is considered by Cho et al. [24] and a method is presented which is altered to make it applicable in hull form modeling. The method generates at least one tensor-product B-spline patch for each loop of the curve network while the patches join with tangent continuity. Moderately irregular curve networks are accepted by the method what means that the number of curves which are allowed to intersect at a node of the curve network may range between three and five and the number of curve segments defining a loop of the mesh is allowed to be within three and six. This provides a reasonable topological freedom for curve-based hull form modeling. If necessary, additional curves are automatically introduced to the network to match the topology with the limitation of tensor-product B-splines to four-sided surfaces. At irregular nodes of the network it is not always possible to interpolate the curves exactly and to simultaneously ensure tangent continuity across the patch boundaries. Therefore the interpolation condition is relaxed and a close approximation of the curves is accepted for those cases. An optional fairing of the curve network or the patches is not performed. While this basically represents the state of the art in curve network interpolation with some details optimized for hull form modeling, the method of Rhim et al. [95] improves the overall fairness of the hull surface based on fitting. Additional curves are derived from the input network and the surface patches are fitted to those curves rather than just interpolating the input curves as it is the case for the former method. Finally, in Oh et al. [77] so-called T-nodes are considered which allow to end curves anywhere in a network. This extends the capabilities of the curve-based design because it is possible to introduce local curves to the network which do not have to run across the entire hull in order to keep the overall topology valid. The work is only recently continued in Oh et al. [78, 79], but the studies are rather mathematical and not particularly focused on the application for curve-based hull form modeling.

The study of Lee et al. [58] employs subdivision surfaces in order to address the limitations

of tensor-product B-splines in the context of curve-based design. They are recognized as being able to represent smooth surfaces of arbitrary complexity and a method is presented to interpolate curve networks of arbitrary topology with a single subdivision surface. The algorithm presented by the authors is of similar complexity as the algorithms used to solve the same problem based on tensor-product B-splines, but the generated surface is almost everywhere curvature continuous. Only a few isolated spots are restricted just to normal continuity. This is generally a great step forward compared to the tensor-product B-splines setup. However, the study neither considers complex hull form characteristics such as knuckles or other hull form features, nor the integration of subdivision surfaces into the ship design process. The latter is an important point to consider because tensor-product B-splines are the industry standard for exchange of hull form data and it is a complex matter to convert subdivision surfaces to this surface representation.

The form parameter design is considered by Pérez et al. [92] and later continued in Pérez and Clemente [91] for simple hulls like sailing yachts or patrol boats. Based on the given parameters, a sectional area curve and the waterline are automatically generated. Along with a specification of the hull's center line and deck line given by the designer geometrically in terms of curves, the stations of the hull are generated. Finally a tensor-product B-spline is fitted to the stations in order to arrive at a surface representation applicable in downstream processes. The framework is extended in Pérez-Arribas and Péter-Cosma [93] to define hard-chined hulls which are primarily used for planing boats, but this still belongs to the category of simple hull forms. The parametric generation of complex hull forms is only recently reconsidered by McCulloch [67] who focuses on the robustness of this design method. The design space, which is spanned by the design parameters, usually includes parameter combinations which are not feasible or the result may reflect the parameters from a mathematical point of view, but the generated hull surface is not suitable for practical applications. The method of McCulloch [67] automatically narrows the design space to feasible designs in order to increase the robustness and to simplify the form parameter design. Generally, the method follows the standard approach to define a curve network from the parameters first, and generate the surface patches in a second step. The mapping of the input parameters to curves is also described in Birk and McCulloch [7]. The surface generation is based on truncated hierarchical B-splines introduced by Giannelli et al. [33] which allow to represent local features without having the entire surface necessarily defined at the same level of detail as it is the case for tensor-product B-splines. However, it is relatively easy to convert the result later to conventional tensor-product B-splines to facilitate the data exchange with subsequent design tools.

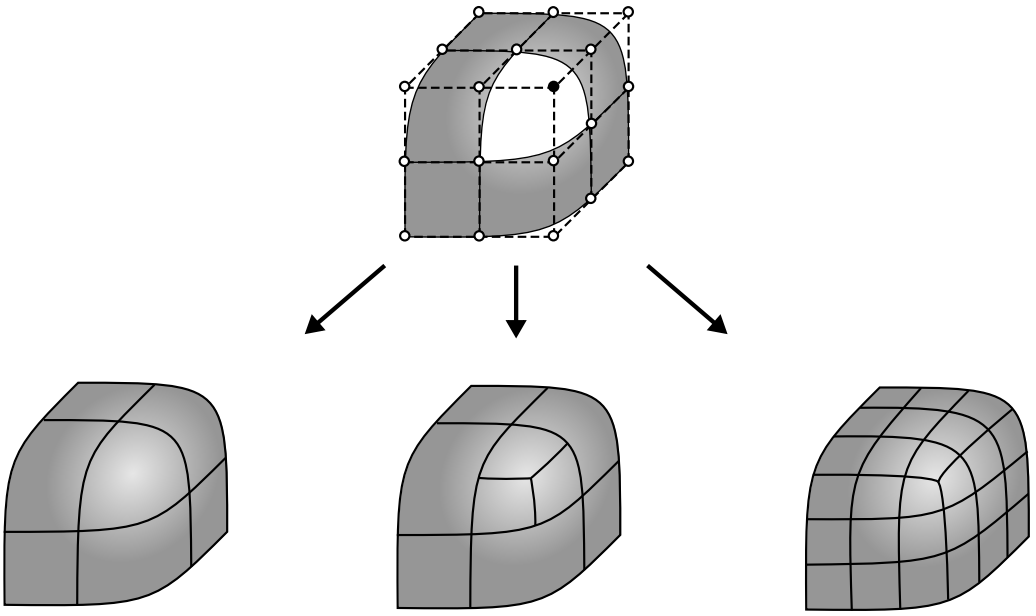
2.2. B-spline surfaces

The standard representation of smooth surfaces in computer-aided design are tensor-product B-splines. They are defined by a control mesh consisting of control points and the surface is a smooth approximation of the control mesh. Due to the mathematical construction based on the eponymous tensor-product, the control mesh is required to be regular and as a consequence the surface is always four-sided. The latter property represents a problem in practice because in complex real-world design scenarios, such as hull form modeling, the objects being designed are generally not representable by a single four-sided surface. The key to advance the theory of B-spline surfaces is therefore to have them defined on irregular control meshes in order to represent smooth surfaces of any complexity.

A key property of B-splines is that every control point affects the surface only locally. This means that any sufficiently large regular subset of a control mesh can be transformed into a tensor-product B-spline surface, even when the overall control mesh is irregular. This is shown in the top row of Figure 2.3 where the control mesh features a single irregular control point highlighted by a black dot. Only three edges are connected to this control point, referred to as valence $n = 3$, with four edges are considered as regular. All other interior control points are of valence $n = 4$, the edge control points are of valence $n = 3$ and the corner control points are of valence $n = 2$ what constitutes a regular control mesh. With only a single irregular control point and the rest of the control mesh being regular, a significant portion of the surface can be defined in terms of tensor-product B-splines. However, where the irregular control point affects the surface this construction is not applicable and the lack of definition results in a n-sided hole. To deal with B-spline surfaces defined on irregular control meshes is a well-studied problem. Basically three solutions are proposed in the literature: a single n-sided patch is constructed to fill the hole, the hole is filled by n four-sided patches, or a subdivision surface is used, see the bottom row of Figure 2.3. The first two solutions are referred to as patch constructions in this thesis and the corresponding literature is introduced in the next section. The third alternative, subdivision surfaces, is reviewed subsequently.

Patch constructions

Two kinds of patch constructions are differentiated: either a hole is filled with a single n-sided patch or the hole is filled with n four-sided patches, see Figure 2.3 (a) and (b) for clarification. A representative of the first kind is proposed by Loop and DeRose [63] and relies on a generalization of Bézier surfaces to n-sided surfaces described by the same authors in [66]. The surface representation allows to define biquadratic and bicubic B-spline surfaces on irregular control meshes what is designated as generalized B-spline surfaces. While the control meshes are significantly less restricted compared to the tensor-product setup, the control meshes are not allowed to be entirely general. For the biquadratic case, all faces of the control mesh have to be four-sided and therefore regular, whereas the control points



(a) n-sided patch (b) n tensor-product patches (c) subdivision surface

Figure 2.3.: The key to advance the theory of B-spline surfaces is to have them defined on irregular control meshes. Top row: the control mesh of a B-spline surface with one irregular control point (black dot). In the vicinity of the irregular control point, the surface cannot be defined in terms of tensor-product B-splines and a hole remains. Bottom row: the current state of research offers three solutions. Either the hole is filled by a single n-sided patch, the hole is filled by n four-sided patches, or a subdivision surface is used.

are allowed to be irregular. For the bicubic case the opposite applies: the faces may have any number of sides and are therefore allowed to be irregular, but in contrast the control points have to be regular. This allows to represent B-spline surfaces of any complexity, but the designer has to take into account the above-mentioned limitations of the control mesh topology. The individual patches are constructed to meet with normal continuity. Neither non-uniformity nor rational B-splines are considered, but the latter represents a straightforward extension. The work of Loop and DeRose [63] is particularly important to classify this thesis with respect to other literature because, among other aspects, the surface representation introduced in Chapter 4 of this thesis is also referred to as generalized B-splines. In contrast to this previously claimed generalization, the subsequently presented surface representation offers an unrestricted control mesh topology and the corresponding surface is almost everywhere curvature continuous and not only normal continuous. Recall that curvature continuity is required to consider a surface as smooth in the context of hull form design and consequently the generalization of Loop and DeRose [63] does not meet

the continuity requirements of this application, but the generalization of B-spline surfaces proposed in this thesis is sufficiently smooth in this context.

To construct a single n-sided patch that is smoothly connected to the other patches of the surface seems to be generally easier than to construct n four-sided patches with same properties due to the fact that generalizations of B-splines to n-sided patches are usually designed exactly for this scenario. However, n-sided patches are not widely supported by computer-aided design systems and therefore not widely available in practice. Hence, the remainder of this section focuses on the second kind of patch constructions which is more relevant in practice, namely to fill holes with n four-sided patches as shown in Figure 2.3 (b).

Wijk [108] constructs bicubic patches over irregular control meshes. The topology of the control mesh nevertheless is limited. All faces are required to be quadrilateral and the control points are either limited to valence three and four, or alternatively all control points are limited to odd valences. The patches are constructed to meet with normal continuity. The generalization of biquadratic surfaces to irregular control meshes is considered by Peters [81], but it requires the use of cubic triangular patches in the vicinity of irregular control points. The method is improved in Peters [83] and allows to parameterize the entire surface in terms of four-sided patches. In regular regions quadratic patches are used, whereas in irregular regions cubic patches are employed. However, the problem is only considered as well-solved on the basis of triangular patches, but the restriction to four-sided patches affects the continuity and shape characteristics of the result. The latter point is addressed in Peters [82] where biquadratic patches are employed to improve shape and continuity characteristics for purely four-sided patch constructions. All constructions are designed to provide a C^1 parametrization of the surface which constitutes a normal continuous surface. For purely four-sided patch constructions, however, high-degree parametrizations are required to ensure normal continuity. The normal continuous parametrization's complexity of control meshes of arbitrary topology with bicubic tensor-product B-splines is analyzed in Peters and Fan [87]. It is shown that at least two interior double knots per edge are required to ensure normal continuity for general input meshes. In Fan and Peters [30] these results are utilized to provide explicit formulas to compute the control points of normal continuous patch constructions in the vicinity of irregular control points based on bicubic tensor-product B-splines.

The problem of curvature continuous patch constructions is tackled for instance in Peters [84]. Triangular Bézier patches of degree six are employed in the vicinity of irregular control points. Where the control mesh is regular, the triangular patches may be replaced by bicubic tensor-product B-splines. A curvature continuous patch construction based on tensor-product Bézier patches is presented by Peters [86]. The algorithm constructs $4n$ patches to fill the hole around an irregular control point of valence n which are of degree 3×5 next to the irregular control point and the remaining patches being standard bicubic Bézier patches. Another curvature continuous patch construction is presented by Loop and Schaefer [65], where only n biseptic B-spline patches are used to fill the hole around an irregular control

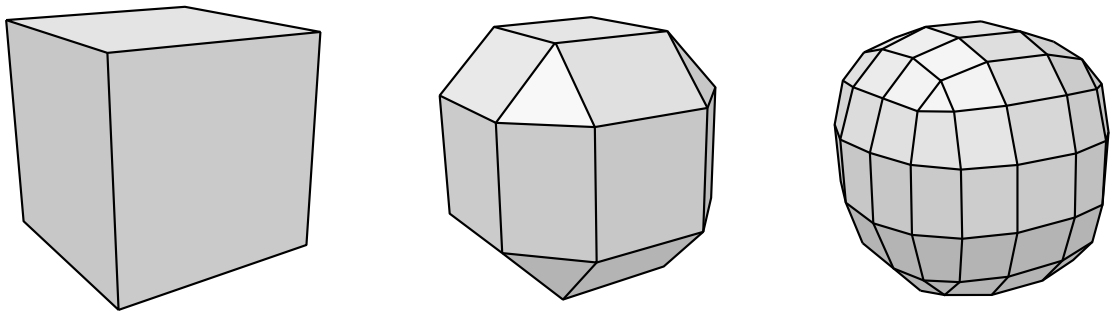


Figure 2.4.: The classical view on subdivision is as a method to refine polygon meshes. Two steps of Doo-Sabin subdivision of a simple box ($k = 0$) are illustrated. It usually takes only a few subdivision steps to obtain a polygon mesh that looks like a smooth surface. Although flat shading is used in this figure, the last polygon mesh ($k = 2$) looks already quite smooth.

point of valence n. The high degree introduces an additional fairness optimization potential of the patches by an energy functional's minimization. An alternative to improve the fairness of patch constructions are guided spline surfaces. First a guide surface is constructed which prescribes the local shape of the surface and avoids unwanted curvature oscillations as usually introduced by local constructions. The guide is not part of the final surface representation, but sampled by a finite number of tensor-product patches, see Karčiauskas and Peters [41] for an overview. In Karčiauskas and Peters [42, 43] the guide is sampled by biquintic patches. The hole is filled with $4n$ patches with n being the valence of the irregular control point. In Karčiauskas and Peters [45] biseptic patches are used to curvature continuously sample the guide with only n patches. The trade-off between the polynomial degree of the patches and the surface quality is discussed in detail by Karčiauskas and Peters [46]. In Karčiauskas and Peters [44] the formal requirement of curvature continuity is dropped to normal continuity for sampling a guide surface with bicubic tensor-product B-splines only. It is demonstrated, however, that the result is still close to curvature continuity as well as the overall fairness remains comparable to the previous cases.

Subdivision surfaces

Subdivision surfaces can be viewed from two complementary viewpoints. The first point of view considers subdivision as a method to refine polygon meshes. The subdivision algorithm describes the refinement rules for the mesh and usually takes only a few subdivision steps to obtain a polygon mesh that looks like a smooth surface. This is the classical view on subdivision surfaces and illustrated in Figure 2.4. The mesh remains a polygon mesh as long as not infinitely subdivided. The linear nature of polygon meshes is generally not qualified for

the representation of smooth surfaces and therefore this first view on subdivision surfaces is only of limited suitability for a generalization of B-spline surfaces to irregular control meshes.

The second point of view originates from the knot refinement of B-spline surfaces. Knot refinement is a well-known operation for tensor-product B-splines to increase the number of control points without changing the surface. For a given refinement strategy, this results in rules to calculate the control points for an equivalent control mesh of larger size. This rules are naturally limited to regular control meshes, but the most popular subdivision algorithms generalize those rules in the presence of irregular control meshes. The classical view on subdivision surfaces utilizes the result to obtain a convenient refinement algorithm for polygon meshes of all kind, but the second view is strictly limited to control meshes of B-spline surfaces. With the control mesh repeatedly refined, the regular regions of the control mesh become larger and allow to define a growing portion of the surface in terms of conventional tensor-product B-splines. This constitutes the third alternative to define B-spline surfaces on irregular control meshes as shown in Figure 2.3 and is exactly the view on subdivision surfaces being promoted in this thesis.

The algorithm of Doo and Sabin [27, 28] is among the first contributions to the subdivision theory, which generalizes biquadratic tensor-product B-splines to irregular control meshes. The algorithm of Catmull and Clark [23] additionally generalizes bicubic tensor-product B-splines. The latter is the most important subdivision algorithm and often employed in theory and practice. In addition, Stam [104] and Zorin and Schröder [114] provide generalizations for B-spline surface of arbitrary degree. The subdivision algorithms are stationary what means that the refinement rules do not depend on the refinement level, but are the same for each refinement step. However, the algorithms are restricted to uniform B-splines only.

Non-uniform B-splines are addressed by Sederberg et al. [100] for biquadratic and bicubic surfaces. The algorithms are non-stationary, but the utilization of certain constraints for the knot vectors yields a stationary algorithm, see Huang and Wang [40] and Sederberg et al. [101]. The latter algorithm requires irregular control points sufficiently isolated from non-uniform features what affects the topological freedom of the control mesh. A special type of control point, the T-point, gives additional freedom for local refinement of the control mesh. Another algorithm that generalizes non-uniform bicubic B-spline surfaces is described by Müller et al. [68] and later improved in Müller et al. [69]. In the vicinity of irregular control points uniform subdivision is used and non-uniform subdivision is restricted to regular parts of the control mesh. In fact, both algorithms blend between uniform and non-uniform subdivision. The first algorithm is non-stationary and the second algorithm is stationary. Finally, Cashman et al. [20, 22, 21] describes a stationary algorithm for non-uniform B-spline surfaces of any odd degree.

The main application of non-uniform B-splines is the boundary behavior of open surfaces. A reasonable boundary behavior of B-spline surfaces employs multiple knot lines. They are also useful to realize interior creases of B-spline surfaces. However, none of the subdivision algo-

rithms for non-uniform B-splines permits multiple knot lines across irregular knots. Modeling features based on non-uniform B-splines are therefore limited to regular parts of the control mesh. An alternative are local modifications of the subdivision algorithm in order to introduce similar modeling features that are not constrained to certain topological requirements of the control mesh. A common modification is given by Hoppe et al. [39]. This modification introduces creases to Catmull–Clark subdivision surfaces that may pass extraordinary points, meet with an arbitrary number of other creases in a common point, or fade out at any point on the surface. An extension of semi-sharp creases is described by DeRose et al. [26] and the normal behavior at extraordinary points is improved by Biermann et al. [5]. A generalization of these features to B-spline subdivision of arbitrary degree is described by Stewart and Foisy [105]. A framework to define features along user-defined curves and not only control mesh edges is described by Biermann et al. [6]. While local modifications of the subdivision algorithm are simple to implement they are not compliant with the tensor-product B-spline theory and provide only limited control on high-order properties such as cross-feature curvature. To model features compliant to the tensor-product B-spline theory based on multiple knot lines in the presence of irregular knots is considered in Kosinka et al. [54, 53]. The concepts are extended to semi-sharp creases in Kosinka et al. [52].

Subdivision surfaces generally inherit their smoothness properties from their tensor-product B-spline parents. However, at irregular control points, the smoothness depends on the subdivision algorithm and all major algorithms provide only normal continuity. The continuity analysis is mathematically involved and at least for low-degree generalizations of B-spline surfaces higher continuity is impossible, see the full analysis given by Peters and Reif [88]. Some researchers ensure curvature continuity by blending the surface locally with a sufficiently smooth low-degree polynomial in the vicinity of irregular control points, see [112, 61, 3].

Important subdivision algorithms such as the predominant algorithm of Catmull and Clark [23] are from the late 70s, but although the algorithms are already introduced as generalizations of tensor-product B-splines at this time, the full potential is only known since an algorithm for the exact evaluation of surface properties is given by Stam [103] in 1998. Before this seminal work, subdivision algorithms are only considered as a method to refine polygon meshes. It usually takes only a few subdivision steps to obtain a polygon mesh which looks like the underlying smooth B-spline surface upon rendering, but as long as the mesh is not subdivided ad infinitum, it is still a polygon mesh. This constitutes the first of the two complementary views on subdivision which is referred to as subdivision surfaces. The linear nature of polygon meshes is generally not qualified for the representation of smooth surfaces in the context of engineering and does not allow to evaluate points or derivatives on the surface. The algorithm introduced by Stam [103] allows to evaluate exact surface properties without the need to subdivide the control mesh. This is the key to construct smooth B-spline surfaces based on subdivision and constitutes the second view which is referred to as generalized B-splines in this thesis. The algorithm is slightly improved by Yamaguchi [109] with

respect to consistent normal directions at irregular points, and the exact evaluation of surface boundaries is described by Lacewell and Burley [56]. The evaluation speed is improved by Bolz and Schröder [16] based on a table driven evaluation strategy where as much data as possible is precomputed. While the evaluation of Stam [103] and its derivatives represent an algorithm, an explicit parameterization for the algorithm of Catmull and Clark [23] is given by Lai and Cheng [57]. The evaluation of non-uniform generalizations with non-stationary subdivision algorithms is considered by Wang et al. [106] and the exact evaluation of the piecewise-smooth subdivision surfaces is given by Zorin and Kristjansson [113]. A well-known problem with the parametrization of Stam [103] are diverging derivatives at irregular knots. The length of the derivative vectors rapidly increases and they are undefined at the irregular knot itself. However, in Boier-Martin and Zorin [8] another parametrization is used that is everywhere differentiable. The same is achieved by Antonelli et al. [3] who blend the surface with a well-defined low-degree polynomial in the vicinity of the irregular knot featuring bounded derivatives everywhere.

The state of the art in subdivision theory provides the means to define smooth B-spline surfaces of any degree on irregular control meshes. The theory is considered as settled for the uniform setup, but the non-uniform setup remains subject of further research. It is possible to compute exact surface properties such as points or derivatives anywhere on the surface without the need to subdivide the control mesh explicitly. In contrast to patch constructions, the complexities to join individual patches smoothly are avoided and the surface generally inherits its smoothness properties from its tensor-product relatives. Each irregular control point, however, maps to a single point on the surface which is limited to normal continuity, but using polynomial blending even high-order continuity is possible at those spots although this is more of a workaround and real high-order continuity at irregularities remains an open problem. From the author's point of view, subdivision surfaces represent the most advanced theory for B-spline surfaces on irregular control meshes.

The work of Antonelli et al. [3] shows the integration of subdivision surfaces in an existing CAD system. CAD systems are typically based on the abstraction of curves and surfaces being represented in parametric form and a closed shell composed of a connected set of surface patches constitutes the boundary of a solid. Subdivision surfaces are successfully implemented as a specialization of this abstraction and are fully compliant to the range of CAD modeling tools such as intersections or boolean operations provided by the system.

Another modeling tool which is particularly important for hull form modeling is the interpolation of curve networks. Creases based on the local modification of the subdivision algorithms as described by Hoppe et al. [39] or Biermann et al. [5] represent a simple form of curve interpolation. A crease is defined by a sequence of control points and the surface interpolates exactly the B-spline curve defined by those points. While this provides a simple framework to interpolate networks of B-spline curves with a closed surface, high-order smoothness across the curves is not easily possible. To interpolate networks of curves smoothly with

Doo-Sabin subdivision surfaces is considered by Nasri [71] and the method is extended to Catmull-Clark subdivision surfaces in Nasri and Abbas [72]. The method is based on the idea that one is able to define curves on a surface which are aligned with a well-known subset of the control mesh referred to as polygonal complex. Modifying the control points of the polygonal complex, it is possible to satisfy interpolation constraints such as prescribed curves or even high-order constraints such as a given cross-tangent or cross-curvature. The method is initially limited to regular curve networks with only two curves intersecting in a common node, but later extended to irregular curve networks in Nasri [74] for Doo-Sabin subdivision surfaces and in Abbas and Nasri [2] for Catmull-Clark subdivision surfaces. Another application of polygonal complexes is the interpolation of a sequence of non-intersecting cross sections, referred to as lofting, as described in Nasri and Abbas [73] as well as in Nasri et al. [70]. While the method originates from the idea to satisfy interpolation constraints with a given control mesh, in Abbas [1] a method is described to automatically construct a control mesh from the curve network. An application is the generation of hull forms as described by Lee et al. [58] which is based on polygonal complexes.

Schaefer et al. [99] describe a method to interpolate irregular networks of cubic B-spline curves with a single Catmull-Clark subdivision surface. The method requires a modification of the subdivision algorithm to ensure the curve's interpolation at irregular nodes of the curve network and parts of the control mesh are calculated based on the minimization of a fairness criterion. The method of Levin [59] blends between normal subdivision and boundary sampling in order to satisfy user-defined constraints what is referred to as combined subdivision. An application of this method to the interpolation of curve networks is described in Levin [60], but the method is limited to regular curve networks.

Conclusion

Patch constructions which are based on the idea to construct n four-sided patches in the vicinity of irregular control points as illustrated in Figure 2.3 (b) are by default compatible to computer-aided design systems and many methods are based on tensor-product B-splines meeting the industry standard for the exchange of surface data. Curvature continuous patch constructions as given for instance by Loop and Schaefer [65] or Karčiauskas and Peters [45] provide the best results in terms of surface fairness and meet the formal requirement of curvature continuity which is required to consider a surface as smooth in the context of hull form design. The disadvantage of the above methods is, however, the complexity of the algorithms and their implementation.

Subdivision surfaces, in contrast, have almost the same continuity and fairness characteristics, but are significantly more simple in theory and practice. The formal lack of curvature continuity at a few isolated spots is usually less of a problem in practice. If required for the application, however, this can be achieved by a curvature continuous blending with a low-degree

polynomial which is conceptually easy and gives similar results as the curvature continuous patch constructions. Subdivision surfaces are therefore considered as a reasonable trade-off between simplicity of the underlying theory and good continuity characteristics. Therefore they are considered as the most advanced generalization of B-spline surfaces to irregular control meshes currently known. What represents a challenge, though, is the conversion of subdivision surfaces to conventional tensor-product B-spline patches for data exchange.

This page intentionally left blank.

3 Tensor-product B-spline surfaces

A comprehensive introduction to tensor-product B-splines can be found in for instance [89, 96, 31, 32]. Tensor-product B-splines are the most popular variant of B-spline surfaces and are the standard surface representation in ship design. This chapter introduces the general characteristics of B-spline surfaces and reviews the particularities of tensor-product B-splines. Especially, the limitation of tensor-product B-splines to four-sided surfaces is explained and it is discussed how this affects the representation of complex surfaces. This is the necessary background to show in Chapter 4 why generalized B-splines are ordinary B-spline surfaces on the one hand, but on the other hand are not limited regarding to the surface complexity.

The limitation of tensor-product B-splines to four-sided surfaces also explains why hull form modeling nowadays is still dominated by the curve-based design, although this method is well-known for a lack of surface quality and is generally a laborious method as pointed out in Chapter 2 with reference to previous research. To identify the limitation of tensor-product B-splines to four-sided surfaces as a main reason for the dominance of curve-based design is a new idea in the field of hull form modeling and represents a controversial point of view in this domain. This chapter's derivation and justification of this position therefore is an essential result of this thesis.

3.1. Definition

A tensor-product B-spline is a function

$$\mathbf{x} : \Sigma \ni (u, v) \mapsto \mathbf{x}(u, v) \in \mathbb{R}^3 \tag{3.1}$$

as illustrated in Figure 3.1 where Σ is called the *domain* which is a subset of $\mathbb{R}^2$ and $\{\mathbf{x}(u, v)|(u, v) \in \Sigma\}$ is the image of $\mathbf{x}$, usually referred to as the *surface*. This formal definition of tensor-product B-splines is not very substantial, but important to understand what differentiates tensor-product and generalized B-splines. In the case of tensor-product B-splines, the domain is limited to rectangular subsets of $\mathbb{R}^2$, denoted by Σ, but for generalized B-splines significantly more complex domains are possible. To clarify the meaning of Equation 3.1, the domain is uniformly sampled in Figure 3.1, illustrated by the grid, and the points on the surface are the image of the grid's nodes.

In order to implement the formal definition of tensor-product B-splines, the standard form

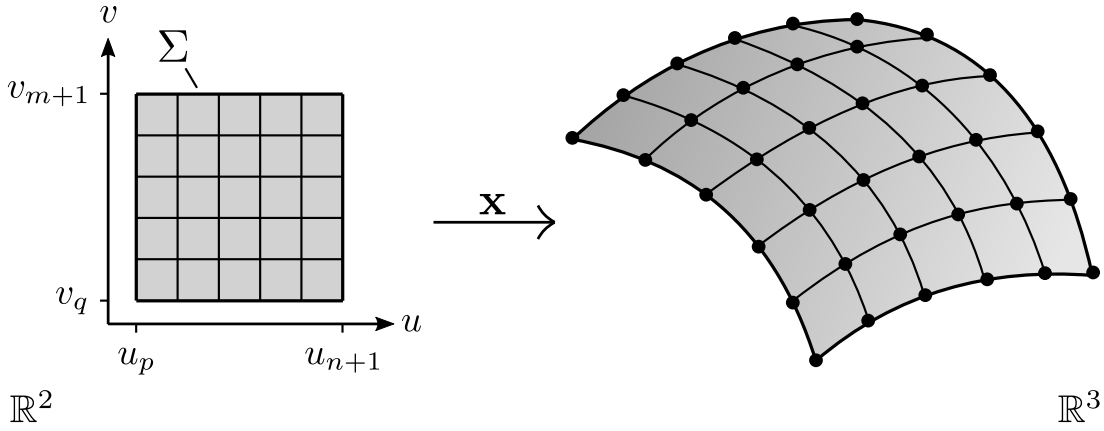


Figure 3.1.: A tensor-product B-spline surface $\mathbf{x}$ is a function from $\Sigma \subset \mathbb{R}^2$ to $\mathbb{R}^3$. The domain always is a rectangular subset of $\mathbb{R}^2$ as shown on the left side. Therefore the image of $\mathbf{x}$, in other words the surface, is always four-sided. For clarification, the domain is uniformly sampled indicated by the grid lines. The points on the surface are the image of the grid's nodes.

of $\mathbf{x}$ reads

$$\mathbf{x}(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)\mathbf{q}_{ij} \tag{3.2}$$

where $N_{i,p}$ and $N_{j,q}$ are the basis of functions of degree p and q respectively, and $\mathbf{q}_{ij}$ are the control points. A tensor-product B-spline is defined on $(n+1)\times(m+1)$ control points with $n \geq p+1$ and $m \geq q+1$. A bicubic B-spline surface is for instance defined on at least 4×4 control points.

The basis functions are given by the Cox-de Boor recursion

$$\begin{aligned} N_{i,0}(u) &= \begin{cases} 1 & \text{if } u_i \leq u < u_{i+1} \\ 0 & \text{otherwise} \end{cases} \\ N_{i,p}(u) &= \frac{u-u_i}{u_{i+p}-u_i}N_{i,p-1}(u) + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1}(u) \end{aligned} \tag{3.3}$$

where the parameter values u_i are called knots, p is the degree, and $n+1$ is the number of control points. The knots are specified in the knot vector $U = [u_0, u_1, \ldots, u_{n+p+1}]$. For the knots only $u_i \leq u_{i+1}$ is assumed. When all knots are uniformly spaced, $u_{i+1} - u_i = \text{const.}$ for $0 \leq i < n+p+1$ the knot vector is called uniform and non-uniform otherwise. A special case are k equal knots in a row what is referred to as a knot multiplicity of k. The most common application is to set $p+1$ knots equal at the ends of the knot vector and keep all other knots

uniformly spaced, for convenience normalized to the unit interval

$$U = [\overbrace{0, \ldots, 0}^{p+1}, \underbrace{\frac{1}{n-p}, \frac{2}{n-p}, \ldots,}_{n-p-1} \overbrace{1, \ldots, 1}^{p+1}]. \tag{3.4}$$

This is called an open uniform knot vector. The knot vector V for the second basis function $N_{j,q}(v)$ is defined accordingly. Only when both knot vectors are uniform, the surface is referred to as a uniform B-spline surface and in all other cases it is considered as non-uniform.

The actual choice of the knot vector has a large impact on the basis functions. This is discussed in detail in the standard literature and beyond the scope of this material. It should be noted, however, that Equation 3.4 represents the standard form for the knot vectors being used in practice. The boundary of the surface depends only on the outer control points and is not affected by the interior control points (unless cross-boundary derivatives are taken into account). Uniform B-spline surfaces are less popular in practice because of a more complex boundary behavior. Other forms of non-uniform knot vectors are used for specialized applications such as the exact representation of conic sections or to improve the interpolation of data points. To conclude the discussion of the tensor-product B-splines, some essential properties of B-spline basis functions are noted:

1. A basis function $N_{i,p}(u)$ is a piecewise-defined degree p polynomial, one for each knot span $[u_i, ui + 1)$. The pieces are connected to each other at the knots u_i. While the polynomials are naturally C^∞, the basis functions are only C^{p-k} at the knots, where k is the knot multiplicity.

2. The product of two basis functions $N_{ij}(u, v) = N_{i,p}(u)N_{j,q}(v)$ is a bivariate polynomial with the continuity characteristics inherited from its operands. One control points $\mathbf{q}_{ij}$ of a tensor-product B-spline is associated to exactly one bivariate basis function N_{ij}.

3. The sum of all basis functions $\sum_{i=0}^{n} \sum_{j=0}^{m} N_{ij}(u, v) = 1$ for all $(u, v) \in \Sigma$. As a consequence, each point on a B-spline surface is an affine combination (a weighted mean) of the control points $\mathbf{q}_{ij}$.

4. The basis functions are generally $N_{ij}(u, v) \neq 0$ for only a subset of the domain. Therefore each point of the surface $\mathbf{x}(u, v)$ generally depends on only a few control points associated to non-zero basis functions.

5. The domain of the basis functions is defined over is $\Sigma = [u_p, u_{n+1}] \times [v_q, v_{m+1}]$ and therefore always rectangular as illustrated in Figure 3.1.

The practical meaning of these properties for the characteristics of tensor-product B-splines are discussed in the next section.

The matrix notation of Equation 3.2 reads

$$\mathbf{x}(u,v) = \begin{bmatrix} N_{0,p}(u)N_{0,q}(v) \\ N_{0,p}(u)N_{1,q}(v) \\ \vdots \\ N_{n,p}(u)N_{m,q}(v) \end{bmatrix} \cdot \begin{bmatrix} \mathbf{q}_{00} \\ \mathbf{q}_{01} \\ \vdots \\ \mathbf{q}_{nm} \end{bmatrix} = \begin{bmatrix} N_{00}(u,v) \\ N_{01}(u,v) \\ \vdots \\ N_{nm}(u,v) \end{bmatrix} \cdot \begin{bmatrix} \mathbf{q}_{00} \\ \mathbf{q}_{01} \\ \vdots \\ \mathbf{q}_{nm} \end{bmatrix} = B\mathbf{Q} \tag{3.5}$$

which is particularly important for two reasons: on the one hand it clarifies the second property from the above list, namely, that one control point $\mathbf{q}_{ij}$ is associated to one bivariate basis function $N_{ij}(u,v)$. On the other hand this form is used in Chapter 4 to develop the theory of generalized B-splines and the reader needs to know the matrix notation of tensor-product B-splines to follow the derivation.

Last but not least, rational B-splines are considered by projecting the control points to homogeneous coordinates first, apply Equation 3.2, and finally project the result back to Cartesian coordinates. This is also the way to consider rational B-splines for generalized B-splines, see Sederberg et al. [100].

3.2. Important characteristics

The required parameters to define a B-spline surface are: the knot vectors U and V, the degrees p and q, and the control points $\mathbf{q}_{ij}$. The knot vectors are rarely used for modeling and usually left in the standard form as specified in Equation 3.4. The other two parameters are relevant for modeling, but the degree is usually either set once at the beginning of the modeling process or sometimes even prescribed by the modeling environment. What remains are the control points as the main parameter to define a B-spline surface.

A B-spline surface approximates the control mesh. This directly follows from the properties of the B-spline basis functions discussed in the previous section. Each point on a B-spline surface is an affine combination of the control points. In particular this means that neither all control points have the same impact on the point nor that each point on the surface necessarily depends on all control points. In contrast, a point on a B-spline surface generally depends on only a few control points. The closer a control point, the greater is the influence on the surface. The described B-spline surface's characteristic of control mesh approximation is illustrated in Figure 3.2. Two B-spline surfaces having the same degree and being defined on the same knot vectors are shown. The left side of Figure 3.2 introduces the terminology and shows the initial control mesh. On the right side, two control points close to the front side of the surface are moved to another position. All other control points remain at their initial positions. This changes the surface, but the greatest change is clearly at the front side close to the control points being moved and gradually fades out towards the rear side. The close

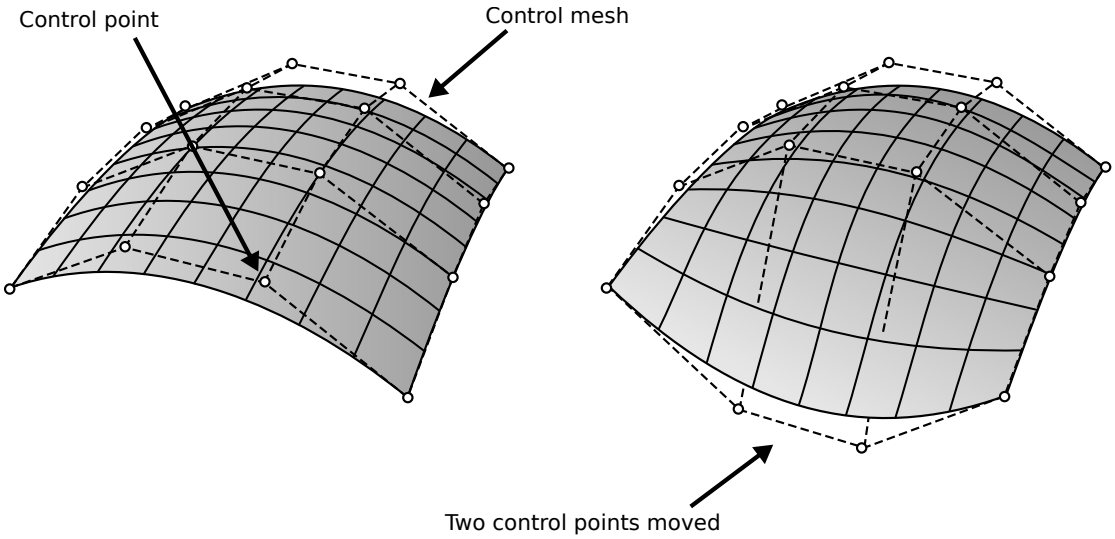


Figure 3.2.: Two B-spline surfaces are shown. Both surfaces are of degree $p = q = 3$ and defined on the same knot vectors $U = V = [0, 0, 0, 0, 1, 1, 1, 1]$. What differentiates both surfaces is only the control mesh. In comparison to the control mesh on the left side, the two control points depicted by the arrow are located at another position on the right side. The greatest change is in the front of the surface and gradually fades out towards the rear boundary.

link between the control mesh and the surface is considered as favorable for modeling. The surface stays close to the control points and is therefore predictable by the designer. Moreover, moving a single control point changes the surface only locally and a small movement implies only a small change of the surface.

A B-spline surface is not only a close, but also a smooth approximation of the control mesh. With reference to the precise definition of the term *smoothness* in Appendix G, this is the point where the second modeling parameter, the degree, becomes relevant. A surface is called smooth when the segments and their connections are G^n continuous where n depends on the application context. For the sake of simplicity, let a B-spline surface have the same degree p in both directions. Then it is a piecewise-defined polynomial whose segments are parametrically C^{p-k} continuously joined with each other. As a knot multiplicity $k > 1$ would only be introduced to intentionally reduce the continuity, it can safely be ruled out from this discussion and the B-spline surface is effectively considered to be C^{p-1} with p being the degree what implies G^{p-1}. To put it in simple terms, the geometric continuity is equivalent to p, and the degree of a B-spline surface is usually chosen to meet the application's requirements to consider the surface as smooth. For instance, when the application requires at least curvature continuity (G^2), as being the case for hull form modeling, the degree of a B-spline surface is chosen to be $p = 3$. The degree of a B-spline surface is generally not chosen to be higher than required because this enlarges the impact of a control point on the B-spline surface. In order

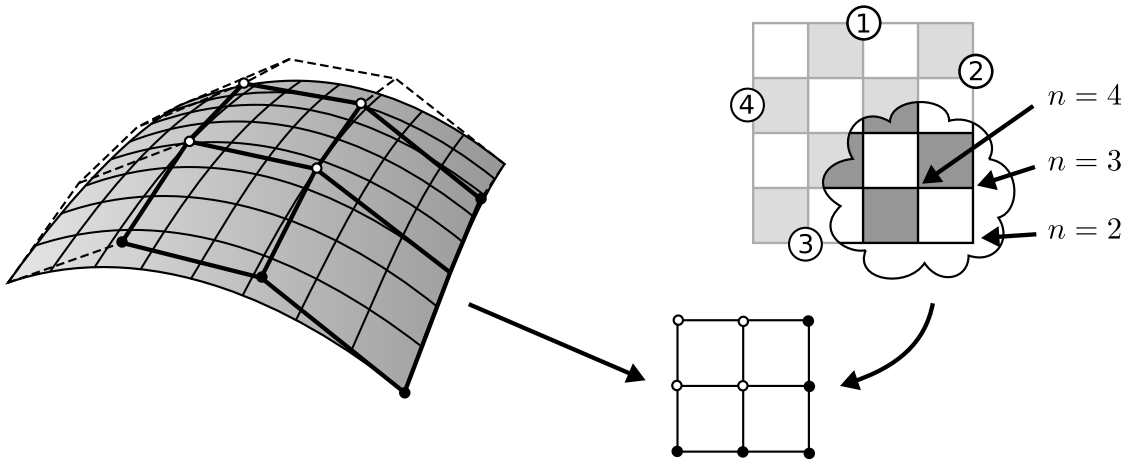
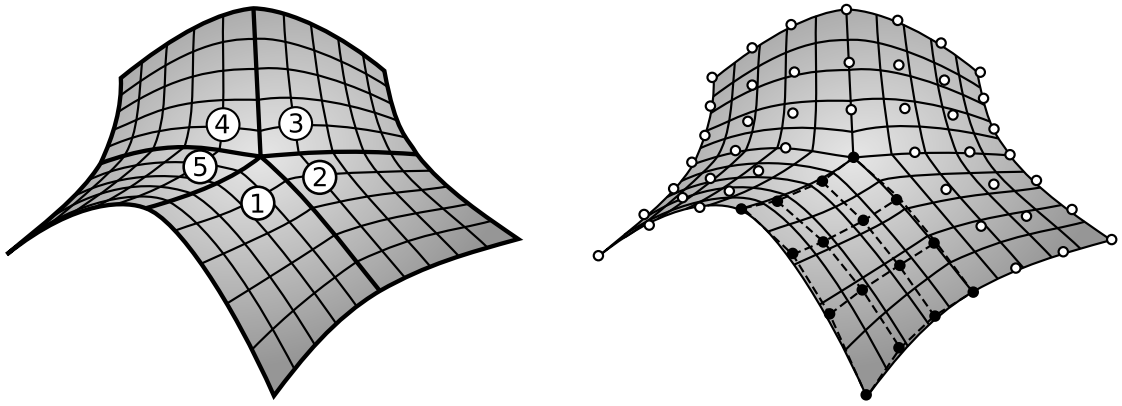


Figure 3.3.: The characterization of regular control meshes. A B-spline surface and the corresponding control mesh are shown on the left side. A part of the control mesh is highlighted. For demonstration purposes this part is schematically reproduced and compared to a chessboard. Both have the same topology, i.e. each interior node is of valence $n = 4$, each boundary node is of valence $n = 3$ and each corner node is of valence $n = 2$ with the term valence referring to the number of edges connected to a node. This is precisely the topology required for a regular control mesh.

to achieve a small local influence of the control points and thus simplify modeling, the degree of a B-spline surface is kept as low as possible.

In the previous section it is clarified that a tensor-product B-spline surface has always a rectangular domain. An immediate consequence is, that the image and therefore the surface, is always four-sided. It is possible to make the control points of one side coincident what causes the corresponding surface boundary to collapse into a single point, but this just looks like a three-sided surfaces and remains a four-sided surface from a mathematical point of view. A consequence is for instance that the derivatives are not uniquely defined in this point.

Another consequence of the tensor-product construction is that the control mesh is always regular. Referring to Equation 3.2, the tensor-product construction is characterized by the double sum and the control points represent a regular grid of $(n+1) \times (m+1)$ control points. The topological characterization of a regular control mesh is illustrated in Figure 3.3. On the left side a B-spline surface with the corresponding control mesh is shown. A part of the control mesh is highlighted by bold edges, schematically reproduced and compared to a chessboard. Neglecting the fact that none has the required amount of fields to actually play chess, it can nevertheless be stated that both, the control mesh and the chessboard, have the same pattern - or more precisely - topology. Each interior node has the valence $n = 4$, each boundary node has the valence $n = 3$ and each corner node has the valence $n = 2$ where



(a) Patches (b) Control points

Figure 3.4.: Complex surfaces can be decomposed into four-sided patches as shown on the left side. As the goal is to join the patches smoothly a reasonable degree is assumed for the patches, say $p = q = 3$ for the illustrated example. This, however, results in the surface being defined by at least 80 control points as shown on the right side. It is obviously a disproportionate effort to define so many points for the illustrated simple surface, especially since all points are constrained by the continuity requirements and cannot be placed freely.

the term *valence* refers to the number of edges connected to the node. This is precisely the topology of the control mesh which is required for a tensor-product B-spline surface. It does not matter from how many fields the chessboards actually is made of, but the topological requirements of the nodes do only allow to construct chessboards which are made of $n \times m$ fields, or $(n+1) \times (m+1)$ nodes respectively and, hence, it is always four-sided. The same applies to tensor-product B-spline surfaces.

3.3. Complex surfaces

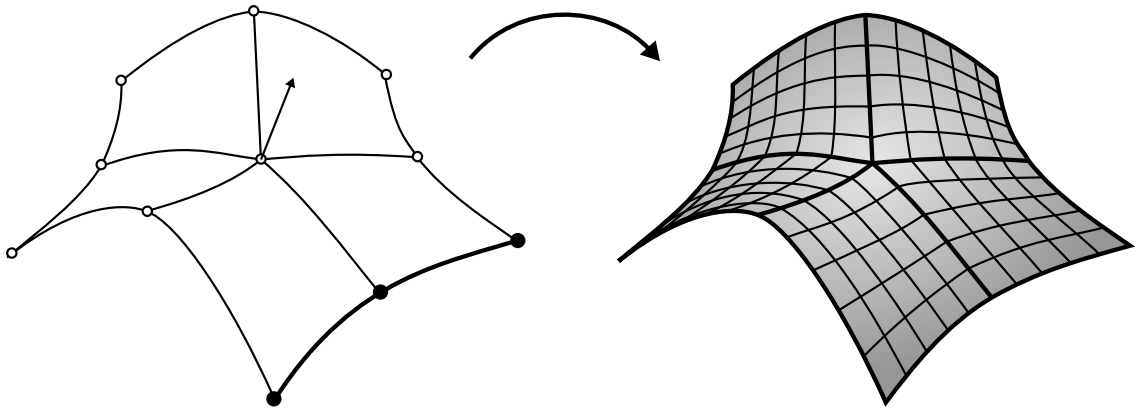
Having clarified in the last section that a tensor-product B-spline surface is always four-sided, the question is how to deal with non-four-sided surfaces as they are frequently encountered in hull form design and other applications. A natural approach is to consider a five-sided surface as the next more complex case. As shown in Figure 3.4 a five-sided surface can be decomposed into four-sided surfaces. The surface is decomposed into five segments. Each segment is four-sided and therefore representable in terms of a tensor-product B-spline. The segments have a common point in the center and join pairwise along a common side. In order to emphasize the role of the individual segments as being just a part of the overall surface, they are referred to as *patches*.

Any surface with $n \geq 3$ sides can be decomposed into n four-sided patches. At the first

glance this looks like a straightforward solution to deal with the limitation of tensor-product B-splines within the scope of complex surfaces. The decomposition into patches affects, however, the applicability of the control points for surface modeling. The patches are supposed to represent an overall smooth surface and therefore a reasonable degree is chosen for each patch, say $p = q = 3$ for the example illustrated in Figure 3.4. A bicubic B-spline surface requires at least 4×4 control points to be defined. Taking into account that the surface is decomposed into five bicubic patches, this totals up to 80 control points as illustrated on the right side in Figure 3.4. Aside of being an excessive number of control points to define such a simple surface, a further consequence is that it becomes hard to model a fair surface. The simple example illustrated in Figure 3.4 is overdetermined by the control points what easily introduces unwanted oscillations to the surface when the control points are not carefully placed. Moreover, most control points cannot be placed freely because they are constrained by the continuity requirements at the patch boundaries (for the example in Figure 3.4 not even a single control point can be placed freely). To formulate the continuity constraints requires a deep knowledge on B-spline theory and involves to solve a linear system. Both circumstances are clearly not suitable for direct modeling.

Since the control points are a poor choice for modeling as soon as a surface is decomposed into several patches, another modeling approach is needed. This is usually the curve-based modeling as depicted in Figure 3.5. Instead of working with an excessive number of control points, designers model the intended patch boundaries. The curve network shown on the left side is generated from only eleven points which are interpolated by the curves and a single vector in the center which specifies the intended normal of the surface. Not all constraints are relevant for each network curve. The bold curve, for instance, depends only on the three points highlighted in black. For this example, the number of constraints to work on is approximately one magnitude smaller than the number of control points required. The patches are automatically generated to interpolate the curves and to join continuously based on the methods reviewed in Chapter 2. The result is, however, usually limited to G^1 continuity what is not considered as smooth in the context of hull form design.

In fact, the method illustrated in Figure 3.5 is nothing else than the curve-based design introduced in Chapter 2 which is still the most popular modeling method in hull form design. It is, however, shown that the curve-based design is primarily a consequence of the limitation of tensor-product B-splines to four-sided surfaces, instead of being a modeling philosophy required by the application context.



(a) Curve network (b) Surface patches

Figure 3.5.: The curve-based modeling requires the designer to specify a curve network to define the patches. As shown on the left side, the curves are manipulated by a number of interpolated points and other constraints such as prescribed normals. The result is shown on the right side. The patches are automatically constructed to interpolate the curves and to join continuously. The methods are usually limited to G^1 continuity which prevents the result to be considered as smooth for applications such as hull form modeling.

This page intentionally left blank.

4 Generalized B-spline surfaces

This chapter develops the theory of generalized B-splines. On the one hand, they are just conventional B-spline surfaces defined on a control mesh which is smoothly approximated, but on the other hand they are neither restricted to rectangular domains nor to regular control meshes. Therefore B-spline surfaces of arbitrary complexity can be defined on a single, potentially irregular, control mesh. The theory of generalized B-splines originates from the field of subdivision surfaces. The basis of generalized B-splines are subdivision algorithms which generalize the knot refinement for B-spline surfaces in the presence of irregular control meshes. The most important contributions are certainly the subdivision algorithms of Catmull and Clark [23] and Doo and Sabin [28]. A next essential contribution is the evaluation algorithm of Stam [103] which allows to evaluate exact surface properties anywhere on a subdivision surface. Finally, the work of Peters and Reif [88] is mentioned. Although the goal of this monograph is to develop the mathematical tools to analyze the continuity characteristics of a subdivision algorithm, it is motivated by the theory of generalized B-splines as well as much of the concepts and the notation are adopted from this source.

4.1. Introduction

A generalized B-spline surfaces is a function

$$\mathbf{x} : \mathbf{S} \ni (u, v, i) \mapsto \mathbf{x}(u, v, i) \in \mathbb{R}^3 \tag{4.1}$$

as illustrated in Figure 4.1. The domain $\mathbf{S}$ is composed of a set of indexed cells Σ_i. Each cell is a rectangular subset of $\mathbb{R}^2$ equivalent to the domain of a tensor-product B-spline. In fact, this is why the same notation Σ is used in both cases. The limitation of a generalized B-spline to a certain cell of its domain

$$\mathbf{x}_i : \Sigma_i \ni (u, v) \mapsto \mathbf{x}(u, v, i) \in \mathbb{R}^3 \tag{4.2}$$

is called a patch. This is in analogy to the usual setup in hull form design and other applications where complex surfaces are represented by a collection of surface patches. However, the patches are now formally treated as subsets of a single surface rather than being independent surfaces on their own. For clarification in Figure 4.1 the surface, that is the image of the entire domain $\mathbf{S}$ under $\mathbf{x}$, is illustrated as a shaded representation. The image of the cell

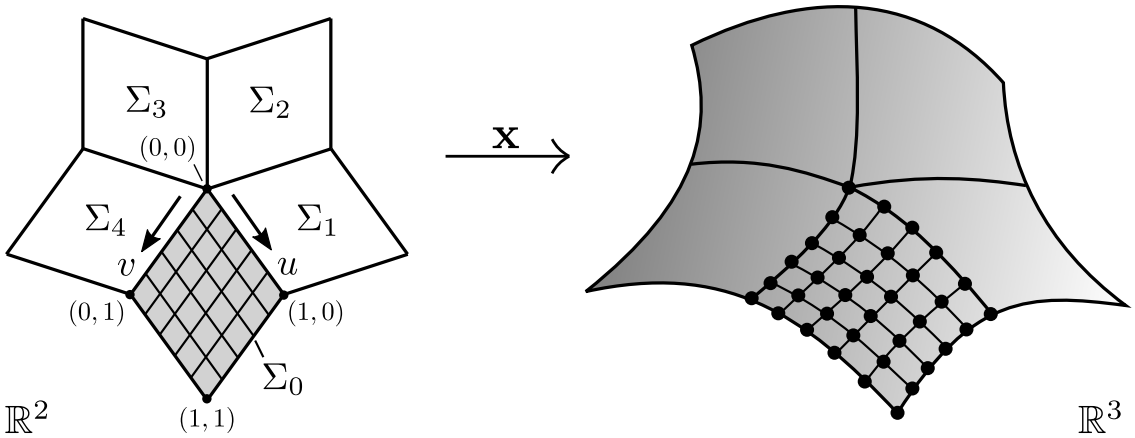


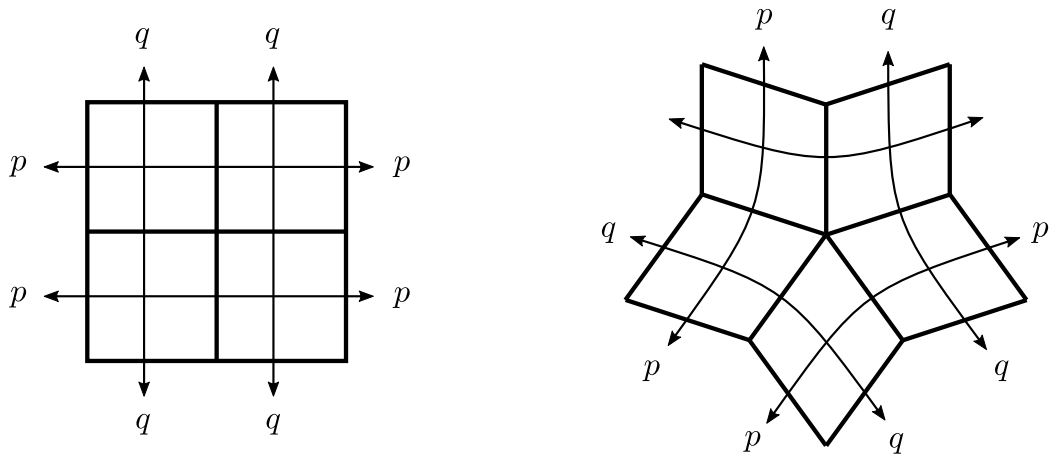
Figure 4.1.: A generalized B-spline surface $\mathbf{x}$ is a function from $\mathbf{S}$ to $\mathbb{R}^2$. The domain is composed of a set of indexed cells Σ_i which are a rectangular subset of $\mathbb{R}^2$ equivalent to the domain of a tensor-product B-spline. The limitation of a generalized B-spline to a certain cell Σ_i of its domain is called a patch. As the domain $\mathbf{S}$ of a generalized B-spline is significantly less restricted compared to the domain of a tensor-product B-spline, the image under $\mathbf{x}$, respectively the surface, is so as well. The given example represents for instance a five-sided B-spline surface, where the curves are the image of the cell boundaries and the points are the image of the uniformly sampled cell Σ_0 as indicated by the corresponding grid.

boundaries under $\mathbf{x}$ is depicted as black curves. In addition, one cell is uniformly sampled as illustrated by the grid, and the points on the surface are the image of the grid's nodes.

With reference to the definition of tensor-product B-splines in Equation 3.1, a generalized B-spline as defined in Equation 4.1 is conceptually very similar. The less restrictive domain definition allows, however, to define surfaces of arbitrary complexity such as the five-sided surface shown in Figure 4.1. The following material shows how all patches are defined simultaneously by the same control mesh, smoothly connected to each other by construction, and each point on the surface being an affine combination of the control points nearby.

4.2. Assumptions

This section introduces the fundamental assumptions made for the theory of generalized B-splines: the degree is supposed to be the same in all directions, only uniform B-splines are covered, the control mesh topology is limited to manifolds and the subdivision rules are assumed to be stationary. These assumptions are explained next.



(a) Regular case (b) Irregular case

Figure 4.2.: For tensor-product B-spline surfaces it is possible to have different degrees p and q in both directions which is generally not possible for generalized B-splines. The degree of two adjacent cells of the domain $\mathbf{S}$ has to match at the common boundary and therefore it is propagated across the entire domain as indicated by the arrows. For a regular domain this does not represent a problem and complies with the tensor-product setup. For irregular domains, however, this result in inconsistent degrees. For instance, the two lower sides are consistently chosen to be of degree p and q respectively, but this results in an inconsistent degree of the upper side. Generalized B-splines are therefore assumed to have the same degree in any direction, referred to as *isotropic degree*.

Isotropic degree

The degree has to be consistent along the common side of two adjacent cells of the domain. As a consequence, the degree is propagated through the entire domain which may result in inconsistently defined surfaces for irregular domains as illustrated in Figure 4.2. For a regular domain, the degree is chosen to be p for one side and q for the other side. As indicated by the arrows, the degree is propagated through the entire domain and the opposite sides are fixed to p and q respectively. This complies with the tensor-product B-spline setup and all four sides of the surfaces are consistently defined to be either of degree p or q. The situation is different for irregular domains. The two lower sides of the domain are consistently defined to be of degree p and q respectively. Due to the propagation of the degree throughout the domain, the upper side involves both degrees. It is, however, unclear how B-spline surfaces of varying degrees can be defined on a single control mesh. Therefore generalized B-splines are prior assumed to be of the same degree in any direction. This is referred to as *isotropic degree assumption* for generalized B-splines.

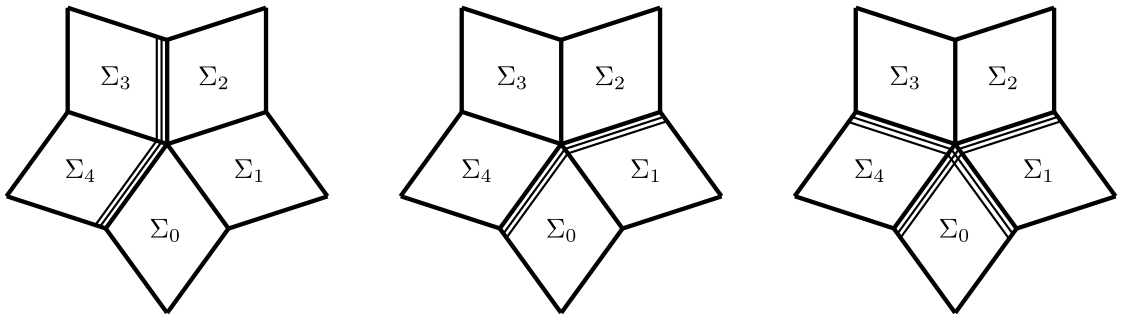


Figure 4.3.: The application of a triple knot line is illustrated. For a quadratic B-spline surface this would result in a crease which is one of the most popular non-uniform B-spline features. The triple knot lines always runs across the entire domain. The direction of the triple knot line, and thus the surface crease, depends on the cell the triple knot is defined for, see the two left illustrations. Moreover, certain configurations are impossible such as a triple knot line separating Σ_0 from the rest of the domain as the triple knots are required to propagate through Σ_3 and Σ_4 as well, see the right illustration.

Uniform knot spacing

The theory of generalized B-splines applies to non-uniform B-splines, but the additional complexity is only of little use. The degree propagation illustrated in Figure 4.2 also applies to non-uniform knot vectors. Let p and q represent two non-uniform knot vectors. It is generally covered by the theory of generalized B-splines to define non-uniform patches $\mathbf{x}_i$ based on these knot vectors. The knot vector is, however, the same for opposite sides of the cells and has to be same for any two adjacent cells. Therefore the knot vectors are propagated through the domain as being the case for the degree. This imposes some significant constraints on the choice of knot vectors for a generalized B-spline surface which is best illustrated for one of the most popular applications of non-uniform knot vectors to have multiple knot lines in order to create interior creases as shown in Figure 4.3. Consider a crease between the patches $\mathbf{x}_0$ and $\mathbf{x}_4$ corresponding to the cells Σ_0 and Σ_4 of the domain. The crease has to run through the entire surface because the knot vector has to be the same for any two adjacent cells. While this represents a limitation on its own, the actual path of the crease on the surface depends on whether the designer chooses to locate the multiple knot in Σ_0 or Σ_4 which results either into a left-hand or right-hand turn of the crease. Clearly this requires a rather complex user interaction and a profound background on B-spline theory to be successfully employed in practice. Certain configurations are impossible to realize such as to separate the patch $\mathbf{x}_0$ by a crease from the rest of the surface as the corresponding multiple knot lines around Σ_0 are required to propagate through Σ_4 and Σ_1 what unintentionally separates $\mathbf{x}_4$ and $\mathbf{x}_1$ as well.

Non-uniform B-splines on irregular control meshes are still a subject of research. The

subdivision algorithm described by Cashman et al. [20] is certainly the most advanced non-uniform algorithm currently known, but it does not allow multiple knot lines to run through irregular knots as discussed in Figure 4.3. The recent work of Kosinka et al. [54, 55, 52, 53], however, is not a substitute for a formal generalization of non-uniform B-splines at irregular knots, but provides at least a method to handle multiple knot lines in this situation.

Since the layout of non-uniform domains is rather restricted and the corresponding development of non-uniform subdivision algorithms is still subject of research, the theory of generalized B-splines is currently limited to uniform B-splines. This is referred to as the *uniform knot spacing assumption* for generalized B-splines, but to drop this assumption is naturally a next step to advance this theory.

Manifold control meshes

The basis of generalized B-splines are control meshes of arbitrary topology. The term *valence* generally refers to the number of adjacent elements in a mesh. For the elements of a mesh the following definitions are used: the vertex valence n_v is the number of incident edges, the edge valence n_e is the number of incident faces, and the face valence v_f is the number of incident vertices or equivalently edges. For the control meshes of generalized B-splines any vertex valence $n_v \geq 3$ is valid for interior vertices, $n_v \geq 2$ is required for boundary vertices, and $n_f \geq 3$ is required for the faces. This minimum valence required for these entities only excludes meaningless configurations and the control mesh topology is basically unrestricted from this point on. The edge valence, however, is restricted to $n_e = 2$ for interior edges and $n_e = 1$ for boundary edges. This is the only major limitation of generalized B-splines in this context and restricts the control mesh topology to manifold meshes.

Stationary subdivision

The subdivision algorithm describes how a refined control mesh is computed from an input mesh. It includes a specification of the topological refinement operation for the mesh and the rules to calculate the new control points. A subdivision algorithm is called stationary when both are the same for each subdivision step. For generalized B-splines it is assumed that the subdivision algorithm is stationary with the exception of an optional initial control mesh subdivision.

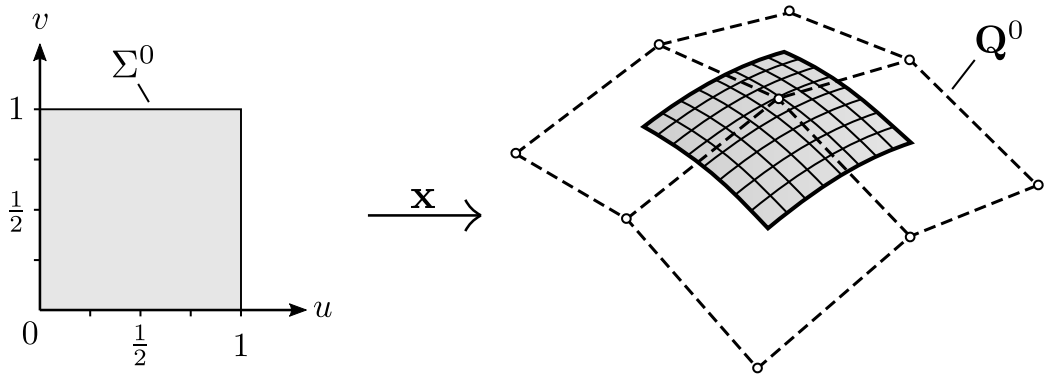
4.3. Subdivision

The key to develop and understand the theory of generalized B-spline surfaces is subdivision. This is an extension of the knot refinement as known from the tensor-product B-spline theory, which is naturally only defined on regular control meshes, to irregular control meshes as faced in the context of generalized B-splines. Knot refinement refers to a redefinition of the knot

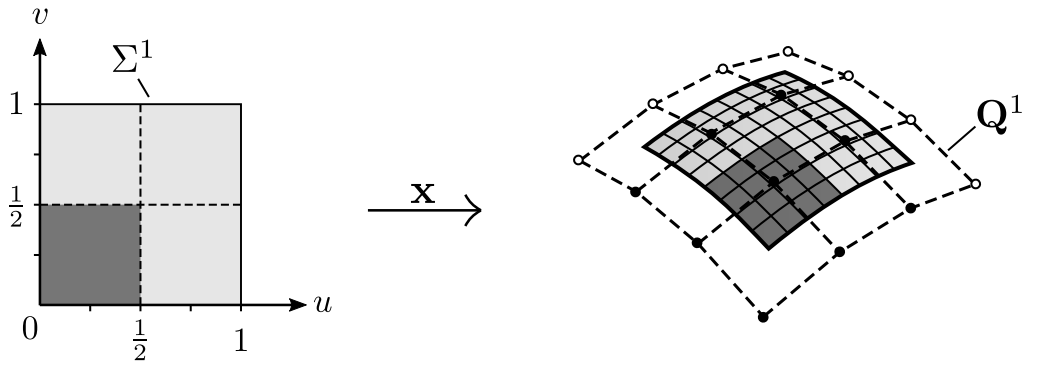
vectors of a tensor-product B-spline to increase the number of control points without changing the surface. The latter is an important characteristic of this operation as it manipulates the knot vector and therefore the control mesh, but the operation does not affect the surface's geometrical properties. The surface stays the same before and after the knot refinement. Therefore the refined control mesh is said to be equivalent to the original control mesh.

For a given refinement strategy of the knot vectors, this operation translates into rules to directly calculate the control points of the equivalent control mesh. The refinement strategy specifies how many knots at which locations are inserted. To derive a subdivision algorithm, the following refinement strategy is essential: New knots are introduced halfway between the existing knots and the knot vectors are finally clipped to the original domain. A comprehensive review of this knot refinement strategy for uniform tensor-product B-splines is given in Appendix H and the reader may refer to this material for clarification. The relevant insight required to understand subdivision is that this particular refinement strategy results in a successive quartering of the domain as illustrated in Figure 4.4. The top row shows a B-spline surface which is defined over the domain Σ^0. For the particular example illustrated, the domain is initially not subdivided by internal knots, but this is only for clarity and not required by the theory. For convenience the domain is assumed to be the unit square $\Sigma^0 = [0, 1]^2$. One refinement step introduces a single internal knot in each direction at $u = v = \frac{1}{2}$ what represents the above-mentioned quartering of the domain. The domain is split into four subsets of equal size. A second refinement step introduces now two additional internal knots at $u = v = \frac{1}{4}$ and $u = v = \frac{3}{4}$ respectively. This represents another quartering of the domain into now sixteen subsets of equal size. In general, the domain is subdivided into 4^k subsets of equal size with this refinement strategy where k represents the number of refinements. The overall size of the domain is not affected with $\Sigma^0 = \Sigma^1 = ... = \Sigma^k$.

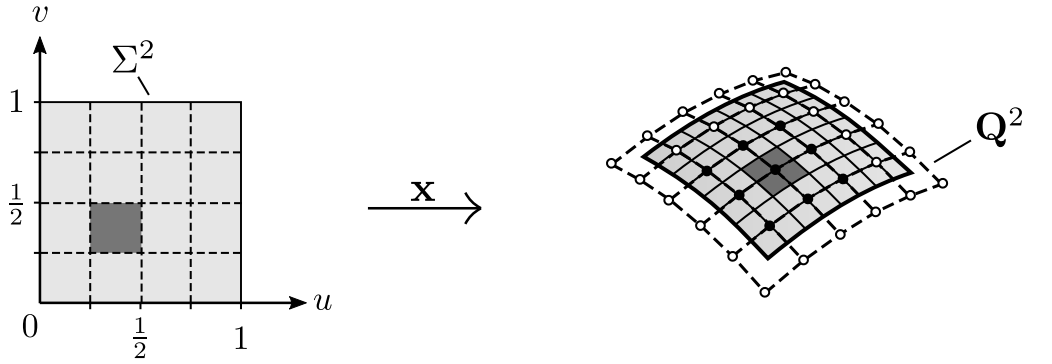
Knot refinement does not only subdivide the domain, but the surface can be understood as being accordingly decomposed into segments. The knot refinement increases the size of the knot vectors and correspondingly grows the control mesh and the number of basis functions. A basis function is generally only non-zero for a small interval. For the initial setup shown in Figure 4.4 without any interior knots and assuming a uniform biquadratic B-spline, all nine basis functions are non-zero over the entire domain Σ^0. Therefore any point of the surface depends on the entire control mesh $\mathbf{Q}^0$. The first refinement introduces a single internal knot in each direction and raises the number of control points and basis functions to 16. At any point of the domain, however, still only nine of them are non-zero and any point on the surface depends only on the associated subset of control points of the control mesh $\mathbf{Q}^1$. How this subset is put together changes at the internal knot lines. For the situation shown in Figure 4.4(b), the surface is therefore understood as being decomposed into four segments. Each of them is defined on a subset of nine control points as illustrated for one segment. The reader may also refer to Appendix H for a comprehensive analysis of the segmentation for this particular case. The next refinement subdivides the domain into 16 subsets and the control



(a) Initial setup



(b) 1st refinement



(c) 2nd refinement

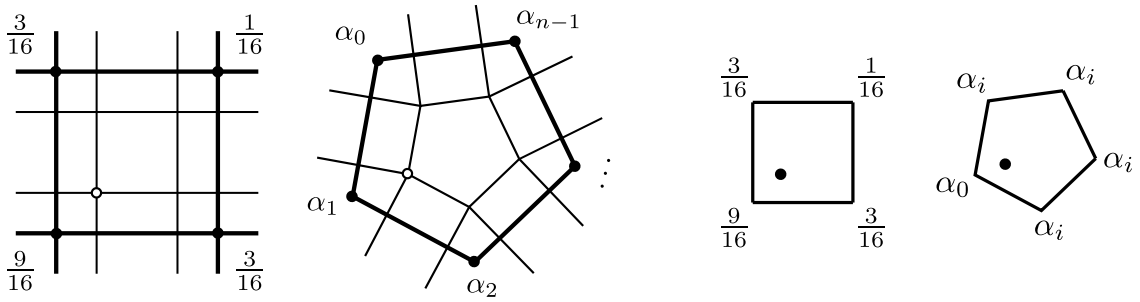
Figure 4.4.: Knot refinement is a well-known operation for tensor-product B-splines to increase the number of control points without changing the surface. The illustrated refinement strategy introduces new knots halfway between the existing knots and thus performs a one-to-four subdivision of the domain. Knot refinement does not only subdivide the domain, but the surface can be understood as being accordingly decomposed into segments with each segment defined on only a subset of the control mesh which is for one example depicted by the black control points.

mesh $\mathbf{Q}^2$ grows to a size of 36 control points. This does not affect, however, the number of non-zero basis functions at any point of the domain which is still nine what only depends on the B-spline degree.[1] The surface is accordingly decomposed into 16 segments, each of them defined on a subset of nine control points as illustrated for one segment in Figure 4.4(c).

The increasing decomposition of the surface into smaller segments which are defined on only a subset of the control mesh due to the repeated knot refinement is crucial to understand generalized B-splines. Without specifically considering irregular control meshes yet, it is already possible to explain the fundamental mechanics of generalized B-splines. Consider one control point as being exceptional, say the topmost control point shown in Figure 4.4. The term *exceptional* just means that the surface cannot be evaluated when an exceptional control point has to be considered as it will happen for irregular control points in the subsequent material. For the initial case with the surface defined on the domain Σ^0, this prevents the evaluation of any surface points or derivatives as they are everywhere dependent on the entire control mesh $\mathbf{Q}^0$ including the exceptional point. A single knot refinement, however, suffices to decompose the surface into four segments with three of them are defined independently of the exceptional control point. While the surface is initially not defined at all, now the majority of the surface is well-defined in terms of a tensor-product B-spline and further knot refinements increase this portion. Based on a suitable formalization, finally each point on the surface can be calculated without the need to perform the knot refinement explicitly.

Now that the refinement strategy is clarified from the perspective of the domain, the control mesh is considered in detail. The refined control mesh $\mathbf{Q}^{k+1}$ can be calculated to be equivalent to the previous mesh $\mathbf{Q}^k$. This is shown for the illustrated example in Appendix H where closed-form expressions for the control points of $\mathbf{Q}^1$ in terms of $\mathbf{Q}^0$ are presented. As the control mesh grows exponentially, this is a tedious endeavor as the number of expressions grows accordingly and their actual form depends on the chosen indexing. Fortunately, the expressions have a repetitive pattern which is not only repeated within a single refinement step, but also through the entire sequence of refinements. This allows a concise notation of the rules to compute the refined control mesh which is independent of the actual indexing and referred to as a refinement mask. For the example illustrated in Figure 4.4, the rules are particularly simple and depicted on the left side of Figure 4.5(a). For each face's corner of the original mesh, one control point is calculated as an affine combination of the original control points defining that face. The weights are applied as illustrated. Afterwards the control points are connected to the refined control mesh by defining one face for each vertex, edge and face of the original control mesh. Although this is only an example, it shows the two relevant characteristics of the refinement rules which generally apply to all B-spline refinements: The

[1] More precisely *at most* nine basis functions are non-zero at any point of the domain. For the particular case being discussed, the situation is actually different at the knot lines where only six basis functions are non-zero and at the intersections of the knot lines where only four basis functions are non-zero. For the sake of simplicity, however, this is omitted in the discussion and does not affect the conclusions.



(a) Subdivision rules (b) Short notation

Figure 4.5.: Generalization of the knot refinement for uniform biquadratic B-splines to irregular control meshes. Left: For each face's corner of the original control mesh, illustrated by the bold edges, one control point is calculated as the affine combination of the original control points defining the face. The weights are $\alpha_0 = \frac{1}{4} + \frac{5}{4n}$ and $\alpha_i = \frac{3+2\cos(2i\pi/n)}{4n}$ with n being the face valence. The original weights as being calculated from the knot refinement are reproduced for $n = 4$. The control points are connected to a refined control mesh, illustrated by the thin edges, by defining one regular face for each vertex, edge and face of the original control mesh. This generalization is known as the Doo-Sabin subdivision algorithm. Right: Short notation in the form of subdivision masks which only gives the weights and omits the topological refinement of the control mesh.

refined control points are an affine combination of a local neighborhood of the original control points and there is a well-defined topological refinement.

A *subdivision algorithm*[2] is a generalization of the refinement rules to irregular control meshes. This is first clarified in the context of the running example. A possible generalization of the refinement rule to irregular faces is shown on the right side of Figure 4.5 (a). Still one control point is calculated for each corner of the face, but the weights are now formulated for faces of any valence and not only regular faces, but the original refinement rules are reproduced for the regular case. This generalization is commonly known as the Doo-Sabin subdivision algorithm [27, 28]. Other generalizations are possible, but it is not subject of this material to introduce the different algorithms. The reader may refer to the literature review for an overview. In contrast, this material specifies the requirements for subdivision algorithms which are compliant with the construction of generalized B-spline surfaces. These are introduced in the next section and afterwards important characteristics of subdivision algorithms are reviewed.

[2] A subdivision algorithm is also referred to as a subdivision scheme in the literature. As being defined in this thesis, both terms can be used interchangeably.

Requirements

A subdivision algorithm suitable to define a generalized B-spline surface must meet the following requirements:

1. All control points are calculated as an affine combination of the original control points. This means in particular that the weights to calculate a control point sum up to one.

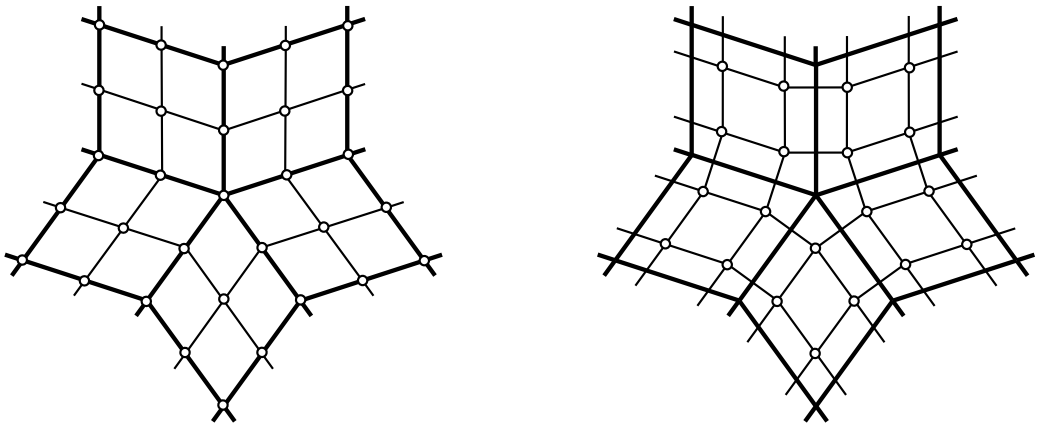
2. The subdivision rules must reproduce the above introduced knot refinement for the corresponding tensor-product B-spline when the control mesh is locally regular. A control mesh is locally regular when the relevant subset of the control mesh to calculate a control point is regular.

3. The subdivision rules must be stationary. This means that the same rules are applied when the control mesh is locally similar. A control mesh is locally similar when the relevant subset of the control mesh to which the subdivision rules apply is topologically the same.

Characteristics

A number of characteristics which apply to all subdivision algorithms suitable to define generalized B-splines can be identified. A subdivision algorithm generally consists of two components: the weights to calculate a control point and the topological refinement of the control mesh. In the context of generalized B-splines, all subdivision algorithms rely on only two topological refinements: the *face split* or the *vertex split* as illustrated in Figure 4.6. Which refinement method is employed depends on the degree of the generalized B-spline surface. Since an isotropic degree is assumed, it is safe to just differentiate surfaces of odd and even degree. Odd surfaces are generalized B-splines of degree $p = 1, 3, 5, ...$ and even surfaces are generalized B-splines of degree $p = 2, 4, 6, ...$ Subdivision algorithms for odd surfaces are always based on a face split. One control point is created for each vertex, edge, and face of the original control mesh. Each edge point is connected to its two adjacent vertex and face points what effectively results in a one-to-four split of the control mesh as shown in Figure 4.6(a). Subdivision algorithms for even surfaces are based on the vertex split. For each face's corner of the original control mesh one control point is created. The control points are connected to each other to form one face for each vertex, edge, and face of the original control mesh as illustrated in Figure 4.6(b). As the topological refinement is given by the degree of the B-spline surface, a short notation of the subdivision as illustrated in Figure 4.5 can be used. This notation omits the topological refinement and only specifies the weights to calculate the control points.

The two split types have a slightly different behavior regarding the topology of the refined control mesh. A face split of a mesh of arbitrary topology results in a mesh whose faces are all



(a) Face split (b) Vertex split

Figure 4.6.: Depending on whether the degree of a generalized B-spline surface is odd or even, the control mesh is topological refined in two different ways. For surfaces of odd degree, the subdivision algorithm is based on a face split. One control point is created for each vertex, edge and face of the original control mesh and connected to represent a one-to-four split of the faces. For B-spline surfaces of even degree, a vertex split is performed. For each vertex of the control mesh n control points are created with n referring to the number of incident faces. One face is created for each vertex, edge, and face of the original mesh. The original mesh is depicted by the bold edges, the refined mesh by the thin edges, and the newly created control points by the empty points.

regular, but the irregular vertices of the input mesh are reproduced in the new mesh. All newly created vertices are, however, regular. Repeating the face split therefore isolates the irregular vertices by an increasing number of regular vertices. A vertex split, in contrast, results in a mesh whose vertices are all regular, but for irregular vertices and faces of the input mesh, irregular faces reproducing their respective valence are created. Similar as before, repeating the vertex split will isolate irregularities from each other by an increasing number of regular faces. Both split types share the same characteristic of isolating irregularities for one thing, and for another thing to produce locally the same mesh topology after a finite number of initial splits. Both is illustrated in Figure 4.7. Two successive subdivisions are shown for one and the same input mesh. The top illustrates the face split and the bottom row illustrates the vertex split. The input mesh has two irregular vertices of valence $n = 5$ shared by the center faces which are just one edge apart from each other. The face split reproduces these vertices, but they are separated by an increasing number of regular faces. After the first subdivision they are already separated by two layers of regular faces and after the second subdivision by four layers. The vertex split initially turns the irregular vertices into irregular faces.

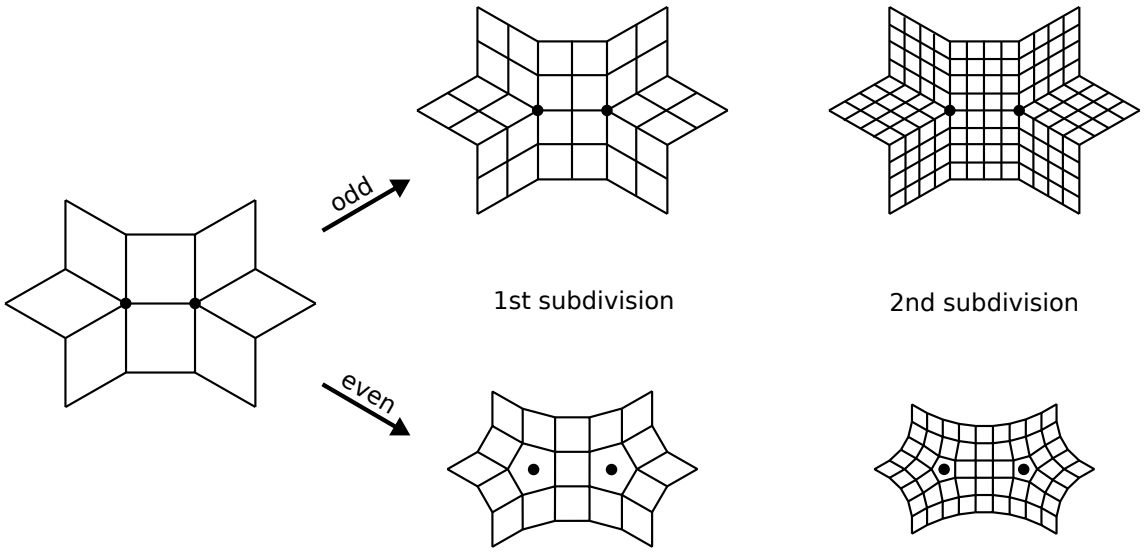


Figure 4.7.: The upper row shows two subdivisions for generalized B-splines of odd degree and the lower row shows the same for generalized B-splines of even degree. The difference between odd and even degrees is the topological refinement of the control mesh: for odd B-splines a face split is performed and for even B-splines a vertex split. After the first subdivision, the same local topology of the control mesh is reproduced for all successive subdivision steps. The irregularities are never removed by subdivision, but they are isolated by an increasingly regular subset of the control mesh.

Therefore the first subdivision not only refines the mesh, but also changes its topology. The second subdivision, however, refines the control mesh what isolates the irregularities from each other, but maintains the topology from the first subdivision. The isolation of irregularities which applies to subdivision algorithms used for generalized B-splines is the key characteristic to define generalized B-splines. The irregularities are not removed by subdivision, but they are surrounded by a growing regular subset of the mesh.

4.4. Domain

Though most material is centered around the control mesh, the domain is what differentiates a generalized B-spline from a tensor-product B-spline. The domain of a tensor-product B-spline is limited to a rectangular subset of $\mathbb{R}^2$ and therefore the surface is always four-sided. A generalized B-spline, in contrast, can have a significantly more complex domain what is the basis to define n-sided surfaces of any complexity. This section shows the construction of domains for generalized B-splines and clarifies how the domain relates to the surface and the control mesh.

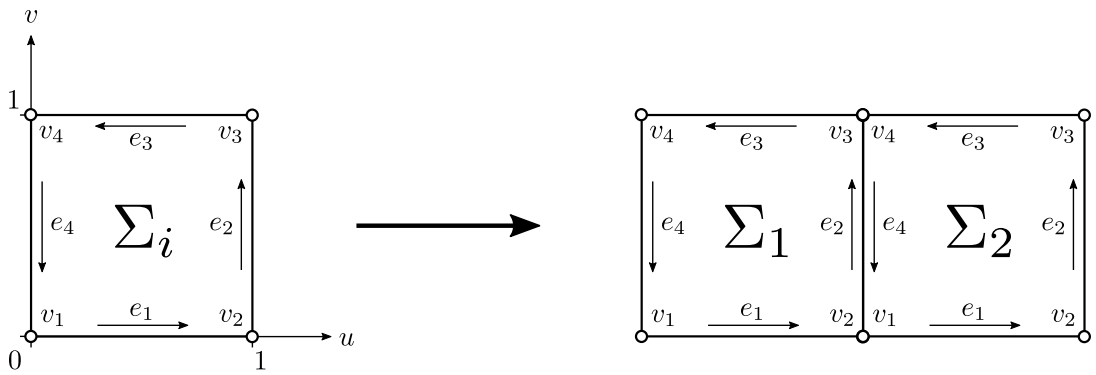


Figure 4.8.: A cell Σ_i is decomposed into the following topological elements: four vertices v_i representing the corners of the cell, and four oriented edges e_i representing the cell's boundary. For convenience $\Sigma_i = [0,1]^2$ is assumed. To construct a domain from the cells, they are topologically joined as shown on the right side by identifying of the corresponding and vertices and edges.

Definitions

The domain $\mathbf{S}$ of a generalized B-spline surface is composed of a set of indexed cells Σ_i. Each cells is a rectangular subset of $\mathbf{R}^2$ equivalent to the domain of a tensor-product B-spline. For convenience, the cells are assumed to be the unit square $\Sigma_i = [0,1]^2$. This is not required by the theory of generalized B-splines which generally allows any rectangular domain, but considerably simplifies the practical implementation of generalized B-splines. It is neither required to consider the parameter range of the individual cells upon evaluation of the patches, nor has it been chosen properly to maintain the consistency between adjacent cells. In fact, it is the assumption $\Sigma_i = [0,1]^2$ that hides the domain in practice behind the control mesh.

A cell Σ_i is decomposed into vertices and oriented edges as follows:

$$\begin{aligned} v_1 &= (0,0), & e_1(t) &= (t,0), \\ v_2 &= (1,0), & e_2(t) &= (1,t), \\ v_3 &= (1,1), & e_3(t) &= (1-t,1), \\ v_4 &= (0,1), & e_4(t) &= (0,1-t) \end{aligned}$$

with $t \in [0,1]$ as shown in Figure 4.8. The first edge starts at v_0 and goes to v_1, the second edge starts at v_1 and goes to v_2, and so on. The edges are consistently ordered to be counter-clockwise for all cells. To construct a domain, the cells Σ_i are joined together as shown on the right side of Figure 4.8. This means that the edges are pairwise tied by $e_i(t) = e_j(1-t)$ as well as the vertices $v_i = v_{j+1}$ and $v_{i+1} = v_j$ which bound the edges are identified. Any two edges tied together have always opposite orientations and are defined on the same interval.

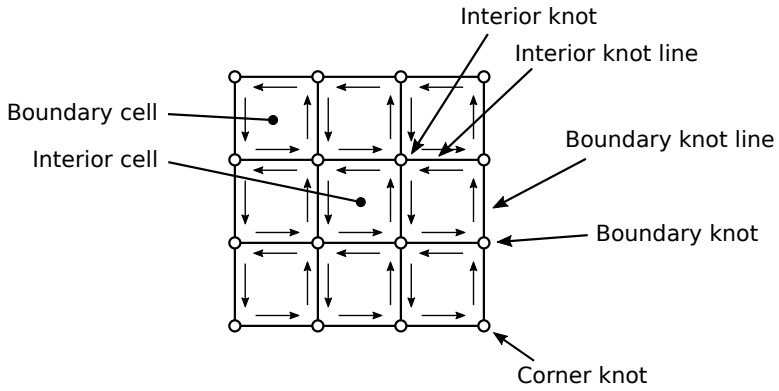


Figure 4.9.: Once the cells are joined together to form a domain $\mathbf{S}$ for a generalized B-spline, the edges are called knot lines and the vertices are called knots in analogy to the tensor-product B-spline theory. Interior and boundary knots and knot lines are differentiated as illustrated. A knot which is only incident to single cell is referred to as a corner knot. The special significance of the corner knot is that a n-sided generalized B-spline surface is defined over a domain $\mathbf{S}$ with n corners.

The former results from the edge orientation chosen consistently for all cells and the latter is given by the assumption $\Sigma_i = [0, 1]^2$.

Once the domain is joined together, the edges are called *knot lines* and the vertices are called *knots*. This terminology is in compliance with the tensor-product B-spline theory where knots mark certain locations in the domain and knot lines subdivide the domain. Several types of knots and knot lines are differentiated depending on whether they are on the boundary or at the interior of the domain, and whether they are regular or irregular. Only two types of knot lines exist: interior and boundary knot lines. An interior knot line is incident to exactly two cells and a boundary knot line is incident to only one cell. Other types of knot lines are not possible. As with the knot lines, interior and boundary knots are differentiated. A boundary knot connects two boundary knot lines and is connected to one or more interior knot lines. A corner knot is a boundary knot which connects two boundary knot lines, but is not connected to an interior knot line and is therefore the only knot type which is incident to only a single cell of the domain. Finally, a boundary cell is bounded by at least one boundary knot line, whereas an interior cell is bounded by interior knot lines only. For clarification, the terminology related to the domain of a generalized B-spline is illustrated in Figure 4.9.

The number of knot lines intersecting in a common knot is called knot valence n. A regular interior knot has a valence of $n = 4$ and a regular boundary knot has a valence of $n = 3$. For all other valences, the knot is considered as irregular. A corner knot has always a valence of $n = 2$ and any surface with $n \neq 4$ corners requires at least one irregular knot. A domain which has only regular knots is referred to as a regular domain.

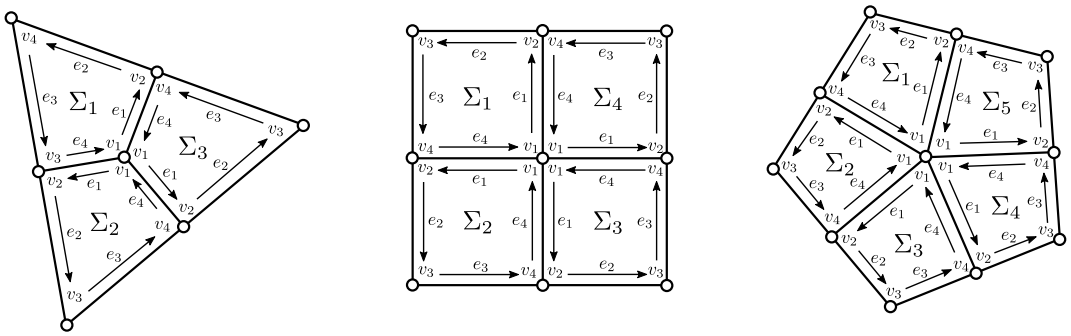


Figure 4.10.: To construct a domain **S** from the cells Σ_i the following principle applies: only two cells may intersect along a common knot line, but an arbitrary number of cells is allowed to intersect in common knot. The knot valence n is the number of cells which intersect in the knot. For interior knots a valence of $n = 4$ and for boundary knots $n = 2$ is considered as regular. Other valences constitute an irregular knot. From left to right: A domain with an interior knot of valence $n = 3$ to define a three-sided surface, a regular domain to define a four-sided surface, and a domain with an interior knot of valence $n = 5$ to define a five-sided surface.

Construction

To construct a domain **S** from the cells Σ_i the following principle applies: only two cells may intersect along a common knot line, but an arbitrary number of cells is allowed to intersect in a common knot. A valence of $n \geq 3$ is required for all knots with the exception of corners which always have a valence of $n = 2$. A number of examples to illustrate the construction of domains for generalized B-splines is shown in Figure 4.10. The left domain has a single interior knot of valence $n = 3$ and all other knots are regular boundary knots with three of them are considered as corners. Therefore the domain defines a three-sided generalized B-spline surface. A regular domain is shown in the middle. All knots are regular and the domain defines a four-sided surface. This could equally well be represented in terms of a single tensor-product B-spline surface. The right domain has an interior knot of valence $n = 5$ and defines a five-sided surface. Please note that the cells of a domain are topologically joined together with each cell $\Sigma_i = [0, 1]^2$ representing the unit square in $\mathbb{R}^2$. For illustration, the domain **S** has to be *embedded* into $\mathbb{R}^2$ which for one thing covers more than just the unit square and for another thing requires to distort the cells when irregular knots are used. Usually, the cells are kept as close as possible to a square, refer for example to Figure 4.3 for an illustration of a domain with an interior knot of valence $n = 5$. In Figure 4.10, however, another embedding is chosen for the same domain which highlights the corners, but this requires a larger distortion of the individual cells and is only used in this illustration for clarification.

In practice, the domain of a generalized B-spline is not explicitly constructed, but derived

from the control mesh as shown in Figure 4.11. This requires the following structure of the control mesh:

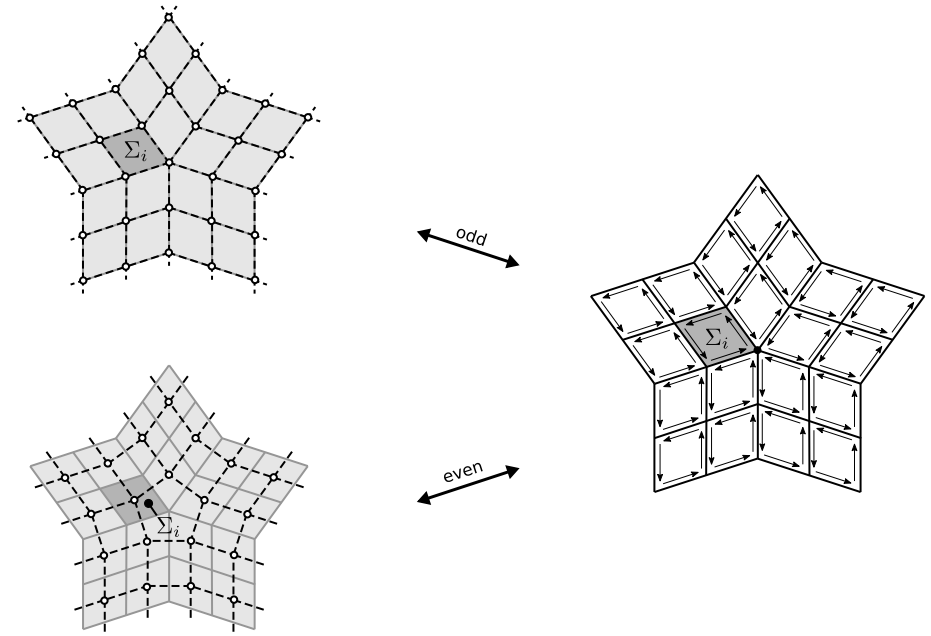
- Even degrees: All faces of the control mesh have to be regular ($n_f = 4$), but the vertices are allowed to have any valid valence $n_v \geq 2$.
- Odd degrees: All vertices of the control mesh have to regular with $n_v = 4$ for interior vertices, $n_v = 3$, for boundary vertices and $n_v = 2$ for corners. The faces are allowed to have any valid valence $n_f \geq 3$.

A single subdivision step which performs a vertex split in the case of even surfaces and a face split in the case of odd degrees, turns any manifold control mesh $\mathbf{Q}^0$ into the required structure to derive the domain. Therefore a domain is said to be in one-to-one correspondence with the once subdivided control mesh $\mathbf{Q}^1$.

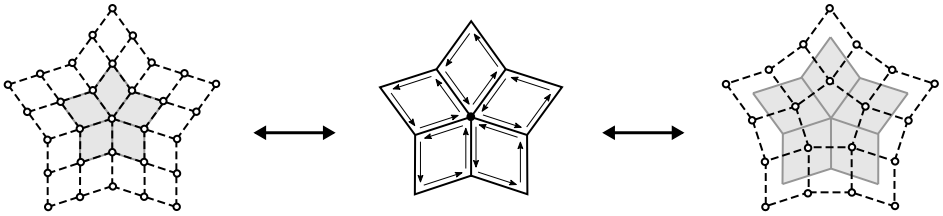
How the domain corresponds to the control mesh depends on the degree. For even degrees, one cell exists for each control point. For each face of the control mesh with the valence n_f a knot of the same valence exists. This is referred to as the *vertex-centered correspondence* of the domain and the control mesh. For odd degrees, one cells exists for each face of the control mesh. For each control point of the control mesh with the valence of n_v a knot of the same valence exists. This is referred to as the *face-centered correspondence* of the domain and the control mesh. It is emphasized that the domain is derived from the control mesh, but conversely the control mesh could be derived from the domain what constitutes the above-mentioned one-to-one correspondence.

How the control mesh and the domain correspond to each other close to the boundary depends on how the subdivision algorithms handles the boundaries. Uniform B-spline surfaces have a complex boundary behavior compared to open uniform B-spline surfaces which are commonly used for modeling. The boundary of a uniform B-spline surface does not only depend on the outer control points, but is affected by $p+1$ outermost layers of control points. In practice, this looks like the surface shown in Figure 4.4 where the outer control points do not intuitively coincide with the boundary of the surface. When a subdivision algorithm reflect these uniform boundary behavior, the outermost $p \div 2$[3] layers of the domain are simply omitted with p is the degree of the surface. A common alternative is the construction of a more convenient boundary behavior based on ghost points, see for instance the description in Rogers [96]. The outer control points are only partially visible for the designer while the invisible control points are referred to as the ghost points. The ghost points are automatically positioned to let the boundary of the surface only depend on the outermost visible control points. This closely mimics the boundary of open uniform B-spline surfaces and is the standard boundary behavior which is realized by the most common subdivision algorithms for generalized B-splines. For this particular case, the domain includes the boundary cells. Both cases are illustrated in Figure 4.11 for the degrees $p = 3$ (left side) and $p = 2$ (right side).

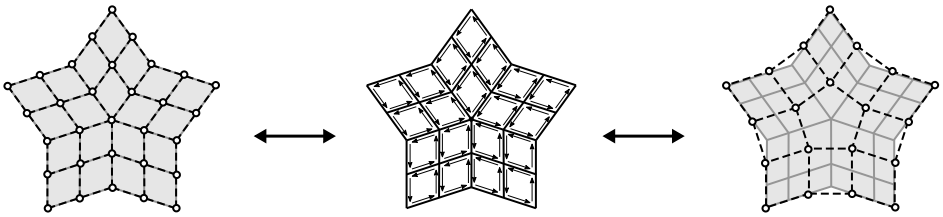
[3]The symbol $\div$ denotes the integer division with $1 \div 2 = 0$, $2 \div 2 = 1$, $3 \div 2 = 1$, $4 \div 2 = 2$,...



(a) Interior



(b) Uniform boundary



(c) Standard boundary

Figure 4.11.: The domain is in one-to-one correspondence with the once subdivided control mesh. For even degrees, one cell exists for each control point. For each face of the control mesh with the valence n_f a knot of the same valence exists. For odd degrees, one cells exists for each face of the control mesh. For each control point of the control mesh with the valence of n_v a knot of the same valence exists. How the boundary is treated depends on the boundary behavior. For standard uniform B-splines the $p \div 2$ outer layers of cells are simply omitted. This is illustrated for the degrees $p = 2$ and $p = 3$. The standard boundary behavior mimics open uniform B-splines and includes the boundary cells.

4.5. Patches

Recall that the limitation of a generalized B-spline to a certain cell Σ_i of its domain is referred to as a patch. When Σ_i has only regular knots, the patch is called a *regular patch* and when the cell has at least one irregular knot, it is called an *irregular patch*. Each patch is defined on only a subset $\mathbf{Q}_i$ of the once subdivided control mesh $\mathbf{Q}^1$. The following material shows how regular patches are evaluated in terms of a conventional tensor-product B-spline and how the same is realized for irregular patches. The latter, however, is limited to patches defined over cells that are incident to only a single irregular knot and therefore control meshes which have only a single irregular control point. Taking into account a general degree $p \geq 1$ and only a single preliminary subdivision of the control mesh so far, this is certainly not always the case and motivates the definition of quadrants. Quadrants separate the knots of the cell and ensure that the evaluation of irregular patches will face only a single irregular knot at a time, but they do not imply that the corresponding control mesh contains only a single irregular control point. Where necessary, the control mesh $\mathbf{Q}_i$ is therefore further subdivided until the last condition is met.

From the mathematical viewpoint, it is significantly easier to require just a sufficient number of preliminary subdivision steps and only then to derive the domain before decomposing the surface into patches. While conceptually simple, this has a serious flaw: the number of patches grows exponentially which is well-known to slow down surface modeling algorithms. For the example specifically considered in this thesis, the representation of hull forms with a cubic generalized B-spline surface, a typical control mesh has about 100 faces what results in the surface being decomposed into 400 patches. This is already a rather large number for this application, but from the author's experience fortunately does not represent a problem in the usual surface modeling systems. A second subdivision step, however, is required for this particular example to guarantee that all irregularities are sufficiently isolated from each other to be evaluated. Using the natural approach to subdivide the control mesh first and only then define the domain to decompose the surface into patches results in 1 600 patches. Choosing a higher degree requires even more preliminary subdivisions and quickly raises the number of patches to several tens of thousands what is likely to affect the surface processing. Therefore the additional effort to limit the preliminary subdivisions to one and to ensure a sufficient isolation of the irregularities by quadrants and an on-demand subdivision of the individual subsets of $\mathbf{Q}_i$ is reasonable. On the other hand, the number of patches could be further minimized by identifying regular regions first and exclude them from the preliminary subdivision or later to combine arrays of $m \times n$ patches into a single patch. Since a few hundred patches are less of a problem in practice, these optimizations are regarded as implementation issues rather than over-complicating the underlying theory.

Quadrants

A patch $\mathbf{x}_i$ of a generalized B-spline surface is defined on $\mathbf{Q}_i$ which is a subset of the once subdivided control mesh $\mathbf{Q}^1$ that includes all control points which have an impact on the patch. As shown next, it is possible to define a patch on an irregular mesh, but this is limited to a single irregularity, i.e. one irregular face for even degrees or one irregular vertex for odd degrees. This implies that the corresponding cell of the patch is limited to only a single irregular knot. Since the domain is in one-to-one correspondence with the once subdivided control mesh $\mathbf{Q}^1$, it is, however, possible that a cell has more than one irregular knot and consequently the control mesh $\mathbf{Q}_i$ contains more than one irregularity. A second preliminary subdivision of the control mesh, though, would guarantee that each cell of a domain has at most one irregular knot. A natural solution would be to subdivide the control mesh twice before the domain is derived, but this would further increase the number of patches into which a generalized B-spline surface is decomposed by a factor of four. Instead, each patch split into four quadrants with each of them including at most one irregular knot and is therefore defined on a control mesh which includes at most one irregularity as illustrated in Figure 4.12. The next paragraph makes quadrants precise.

A quadrant $\mathbf{x}_{i,m}$ with $m \in \{1,2,3,4\}$ of a patch $\mathbf{x}_i$ is defined over only a quarter $\Sigma_{i,m}$ of the cell Σ_i. A patch is mapped to its quadrants by

$$\begin{aligned}
\mathbf{x}_i(u,v) &= \mathbf{x}_{i,1}(2u,2v) && \text{for } (u,v) \in \Sigma_{i,1} = [0,\tfrac{1}{2}] \times [0,\tfrac{1}{2}] \\
\mathbf{x}_i(u,v) &= \mathbf{x}_{i,2}(2v,2-2u) && \text{for } (u,v) \in \Sigma_{i,2} = [\tfrac{1}{2},1] \times [0,\tfrac{1}{2}] \\
\mathbf{x}_i(u,v) &= \mathbf{x}_{i,3}(2-2u,2-2v) && \text{for } (u,v) \in \Sigma_{i,3} = [\tfrac{1}{2},1] \times [\tfrac{1}{2},1] \\
\mathbf{x}_i(u,v) &= \mathbf{x}_{i,4}(2-2v,2u) && \text{for } (u,v) \in \Sigma_{i,4} = [0,\tfrac{1}{2}] \times [\tfrac{1}{2},1]
\end{aligned} \tag{4.3}$$

where each quadrant is supposed to be defined on the control mesh $\mathbf{Q}_{i,m}$. This is the subset of the at least once subdivided patch's control mesh $\mathbf{Q}_i$ that has an impact on the mth quadrant. While $\mathbf{Q}_i$ is subdivided at least once, further subdivisions may be required until each quadrant subset contains only a single irregularity. As each quadrant's domain $\Sigma_{i,m}$ includes at most one irregular knot, this condition is met within a finite number of subdivisions.

An example is illustrated in Figure 4.12. The control mesh $\mathbf{Q}_i$ of the patch consists of two irregular faces which belong to irregular knots of Σ_i at $(1,0)$ and $(1,1)$. In order to define the quadrants the cell is split into four quarters. $\Sigma_{i,2}$ and $\Sigma_i, 3$ each containing a single irregular knot and $\Sigma_{i,1}$ and $\Sigma_i, 4$ including only regular knots. The control mesh $\mathbf{Q}_i$ is once subdivided and the quadrants are defined on the subsets depicted by the black points. $\mathbf{Q}_{i,1}$ and $\mathbf{Q}_{i,4}$ are regular meshes and $\mathbf{Q}_{i,2}$ and $\mathbf{Q}_{i,3}$ are irregular meshes with only a single irregular face. In the case that the latter condition would not have been met yet, the control mesh $\mathbf{Q}_i$ is just further subdivided until the condition of only a single irregularity for $\mathbf{Q}_{i,m}$ is fulfilled. The

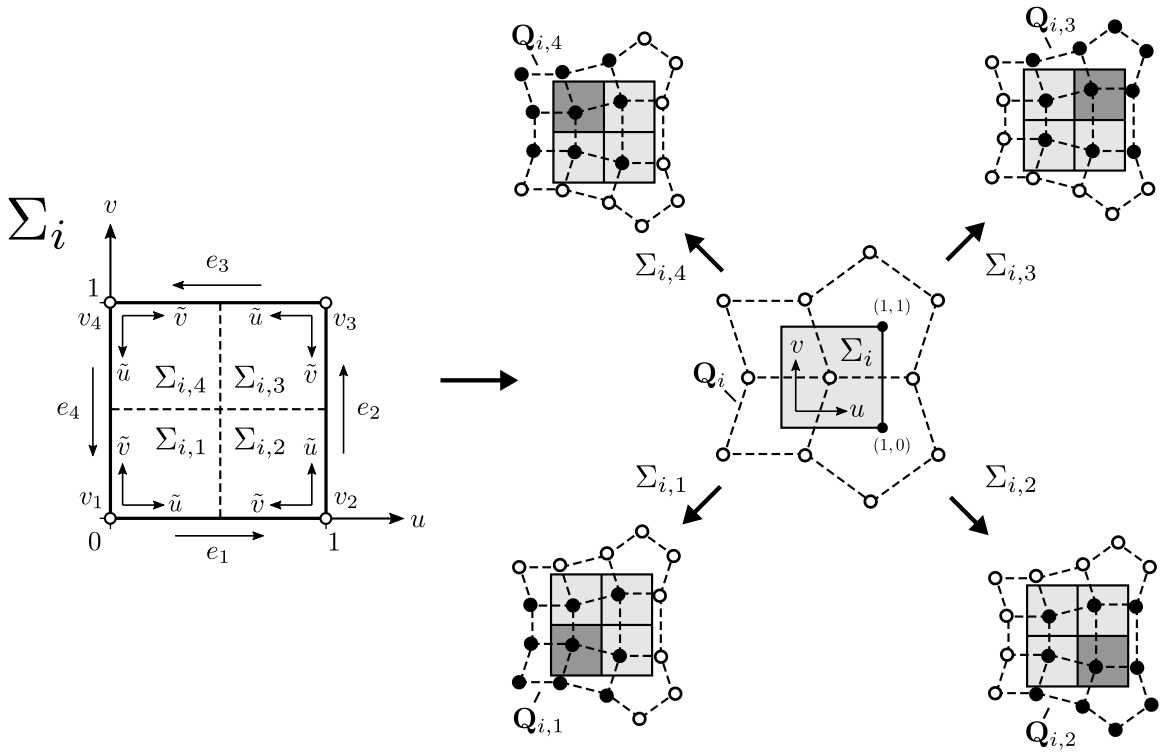


Figure 4.12.: A patch $\mathbf{x}_i$ is subdivided into four quadrants $\mathbf{x}_{i,m}$ with $m \in \{1,2,3,4\}$ to separate potentially irregular knots from each other. A quadrant is defined over $\Sigma_{i,m}$ which represents a quarter of the patch's cell Σ_i and includes at most one irregular knot. A quadrant is defined on subsets $\mathbf{Q}_{i,m}$ of the at least once subdivided control mesh $\mathbf{Q}_i$ of the patch. Additional subdivisions may be required until each subset $\mathbf{Q}_{i,m}$ contains only a single irregularity, i.e. a single irregular face for even degrees or single irregular vertex for odd degrees.

actual definition of $\mathbf{x}_{i,m}$ depends on whether a quadrant's mesh $\mathbf{Q}_{i,m}$ is regular or irregular and is described for both cases next.

Regular quadrants

A quadrant which is defined on a regular control mesh $\mathbf{Q}_{i,m}$ is defined in terms of a tensor-product B-spline with

$$\mathbf{x}_{i,m}(u,v) = \begin{bmatrix} N_0(u)N_0(v) \\ N_0(u)N_1(v) \\ \vdots \\ N_s(u)N_t(v) \end{bmatrix} \cdot \begin{bmatrix} \mathbf{q}_{00} \\ \mathbf{q}_{01} \\ \vdots \\ \mathbf{q}_{st} \end{bmatrix} = B(u,v)\mathbf{Q}_{i,m} \tag{4.4}$$

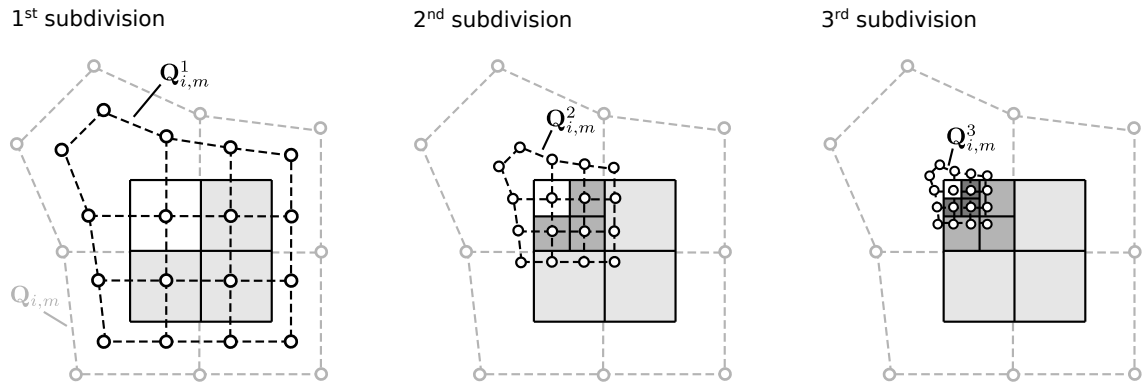


Figure 4.13.: Repeated subdivision is utilized to define a growing portion of an irregular quadrant in terms of a regular subsets of the control meshes $\mathbf{Q}^k_{i,m}$. The first subdivision splits the quadrant into four segments and yields the control mesh $\mathbf{Q}^1_{i,m}$. Three of them are defined on regular subsets of the control mesh as depicted in light gray what already allows to represent a large portion of the quadrant in terms of tensor-product B-splines. A second step of subdivision performs a one-to-four split of the remaining irregular segment with again three of resulting segments are defined on regular subsets of $\mathbf{Q}^2_{i,m}$. This process is repeated for all subsequent subdivisions.

for all $(u, v) \in \Sigma$. It is emphasized that a regular quadrant is defined over the unit square $\Sigma = [0, 1]^2$ and not over $\Sigma_{i,m} \subset \Sigma$. The required mapping from the subset $\Sigma_{i,m}$ to the unit square is carried out by Equation 4.3.

Irregular quadrants

A quadrant which is defined on an irregular control mesh $\mathbf{Q}_{i,m}$ cannot be simply defined in terms of a tensor-product B-spline. However, a step of subdivision splits an irregular quadrant into four smaller segments which are mostly defined on a regular subset of the control mesh. Repeated subdivision results in a growing portion of an irregular quadrant to be defined in terms of regular control meshes as illustrated in Figure 4.13. The first step of subdivision splits the initial quadrant into four segments of equal size and refines the control mesh $\mathbf{Q}_{i,m}$ accordingly with three segments are now defined on regular subsets of the refined mesh $\mathbf{Q}^1_{i,m}$ and only one segment is still defined on an irregular control mesh, refer also to Figure 4.15 for clarification. Naturally, this allows to evaluate a significant portion of the quadrant in terms of tensor-product B-splines. A second step of subdivision splits the remaining segment again what results in exactly the same situation as before with three segments are defined on regular subsets of $\mathbf{Q}^2_{i,m}$. Consequently another portion of the initial quadrant can be defined in terms of tensor-product B-splines. This process is repeated for all further steps of subdivision what allows to evaluate an ever-growing part of the quadrant. The next paragraph makes this

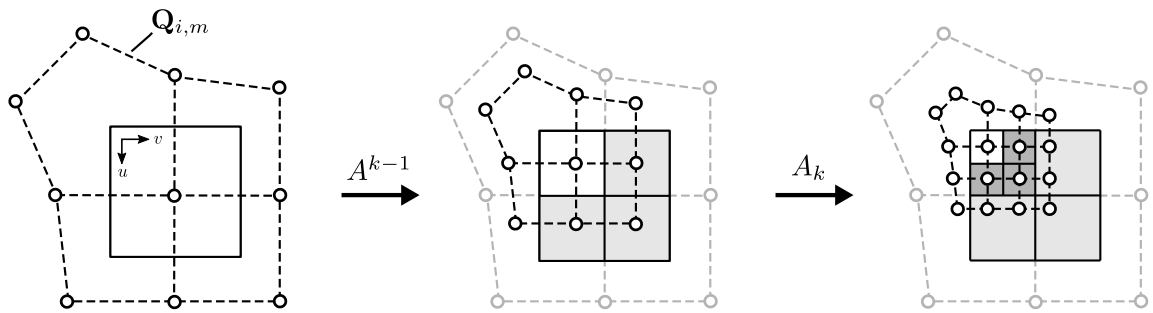


Figure 4.14.: The subdivision matrix can be formulated in two different ways: either it computes the new control mesh for only a single segment or all four segments at once. The subdivision matrix A computes only the control mesh for the segment which is still defined on the irregular control mesh. The extended subdivision matrix A_k computes the control mesh for all four segments generated by a subdivision.

definition of irregular quadrants precise.

The refinement of the control mesh based on subdivision is represented by

$$\mathbf{Q}_{i,m}^{k+1} = A\mathbf{Q}_{i,m}^{k}$$

with $\mathbf{Q}_{i,m}^{k+1}$ is the refined mesh, $\mathbf{Q}_{i,m}^{k}$ is the original mesh, and A is the subdivision matrix. The subdivision matrix reflects the refinement rules for the control mesh as given by the subdivision algorithm. The latter is called stationary when the rules are the same for all subdivisions and consequently the matrix A is also the same for all steps. As this is always assumed for generalized B-splines, the k-times subdivided control mesh

$$\mathbf{Q}_{i,m}^{k} = A^{k}\mathbf{Q}_{i,m}$$

is directly given in terms of the initial control mesh $\mathbf{Q}_{i,m}$. A subdivision step performs a one-to-four split and refines the control mesh accordingly. The subdivision matrix can be formulated in two different ways: either it computes the new control mesh for only a single segment or all four segments at once. With reference to Figure 4.14, the subdivision matrix A computes only the control mesh for the irregular segment. The final subdivision, however, uses the extended subdivision matrix A_k which computes the control mesh for all four segments. The control points of the kth subdivision level are therefore given by

$$\mathbf{Q}_{i,m}^{k} = A_k A^{k-1}\mathbf{Q}_{i,m} \tag{4.5}$$

in terms of the initial control mesh $\mathbf{Q}_{i,m}$ which defines the quadrant.

Three of the four segments which result from the kth subdivision step are defined on regular

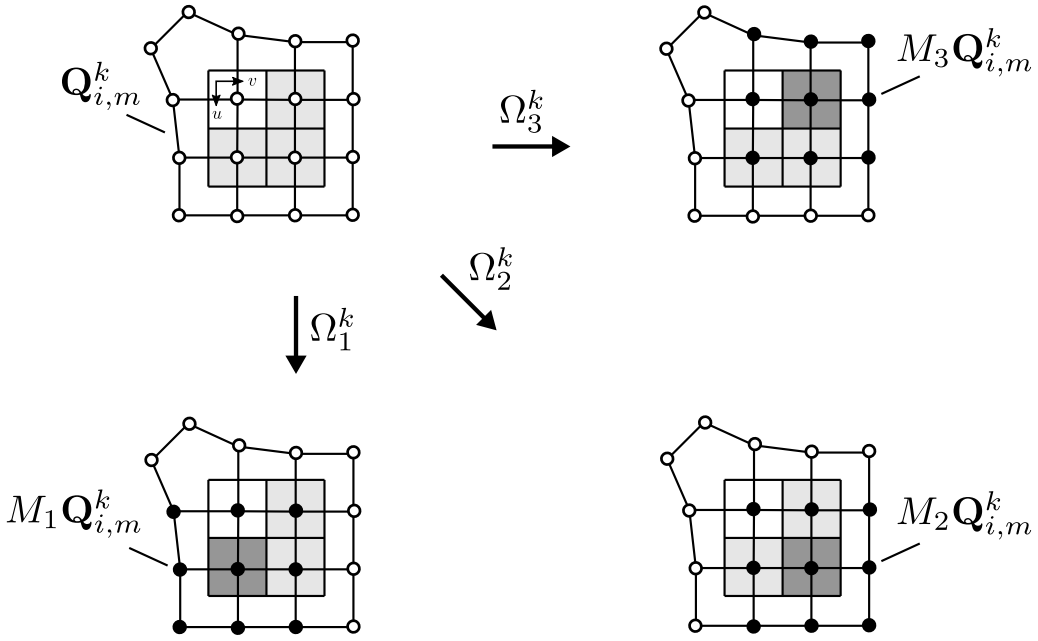


Figure 4.15.: A step of subdivision performs a one-to-four split of a segment which is defined on an irregular control mesh and refines the control mesh accordingly. As a result, three of four segments are defined on regular subsets of the refined control mesh $\mathbf{Q}^k_{i,m}$ as depicted by the black points. They are formally identified by the picking matrix M_n with $n \in \{1, 2, 3\}$.

subsets of $\mathbf{Q}^k_{i,m}$ and can be defined in terms of tensor-product B-splines. The matrix notation of the regular segments is

$$\mathbf{x}^k_{i,m,n}(u, v) = B_n M_n \mathbf{Q}^k_{i,m}$$

with $n = 0, 1, 2$ identifies the three individual segments. The matrix M_n picks the appropriate subset from the control mesh $\mathbf{Q}^k_{i,m}$ as illustrated in Figure 4.15. B_n refers to the matrix notation of the basis functions with the parameter values are scaled according to

$$\begin{aligned} B_0 &= B(2^k u - 1, 2^k v) \\ B_1 &= B(2^k u - 1, 2^k v - 1) \\ B_2 &= B(2^k u, 2^k v - 1) \end{aligned} \tag{4.6}$$

for the kth subdivision level. This parameter scaling is required because an irregular quadrant, like a regular quadrant, is defined over the unit square $\Sigma_i = [0, 1]^2$. The segments $\mathbf{x}^k_{i,m,n}$, however, cover only a fraction of Σ which is referred to as Ω^k_n. Keeping the whole definition of irregular quadrants consistent requires to formulate the basis functions B_n individually for

all Ω_n^k, but fortunately they can equally well be formulated once for Σ and are then scaled according to Equation 4.6 in order to fit with Ω_n^k. Utilizing Equation 4.5 the expression for the individual segments now reads

$$\mathbf{x}_{i,m,n}^k(u,v) = B_n M_n A_k A^{k-1} \mathbf{Q}_{i,m} \tag{4.7}$$

where $\mathbf{Q}_{i,m}$ are the initial control points defining the quadrant. All other terms of this equation are well-known in advance and only dependent on the subdivision rules and the valence of the irregularity.

Computing the power of the subdivision matrix A^{k-1} is an expensive operation. This is an important point to consider for practical implementations of generalized B-splines. Fortunately, the cost of this operation is significantly reduced by the eigendecomposition of A. Recall that

$$A\mathbf{x}_i = \lambda_i \mathbf{x}_i$$

relates an eigenvector $\mathbf{x}_i$ of a matrix A to its eigenvalue λ_i. For all eigenvalues and -vectors of A at once, this relation is represented in matrix notation by

$$AX = X\Lambda$$

with $X = [\mathbf{x}_0, \mathbf{x}_1, ...]$ containg the eigenvectors $\mathbf{x}_i$ of A and $\Lambda = \text{diag}(\lambda_0, \lambda_1, ...)$ being a diagonal matrix which contains the corresponding eigenvalues. A quadratic matrix A of size $N \times N$ is said to be diagonalizable if it has exactly N linear independent eigenvectors in which case the inverse of X exists and

$$A = X\Lambda X^{-1}$$

is called the *eigendecomposition* of A. The advantage of this form is that computing the powers of A is significantly simplified because

$$A^k = X\Lambda^k X^{-1} \tag{4.8}$$

and since Λ is a diagonal matrix, the power is just $\Lambda^k = \text{diag}(\lambda_0^k, ..., \lambda_n^k)$ and therefore the computation of A^k is considerably simplified to the powers of the individual eigenvalues. To utilize the eigendecomposition for the purpose of generalized B-splines, naturally requires that the subdivision matrix A is diagonalizable. A subdivision matrix $A \in \mathbb{R}^{N \times N}$ is a real quadratic matrix which maps a control mesh of N control points to another control mesh of the same size, refer to Figure 4.14 for clarification. The first requirement of being a square matrix is therefore generally given, but the second requirement of N linear independent eigenvectors does not automatically follow from this particular structure of subdivision matrices. Most popular subdivision algorithms, fortunately, have subdivision matrices which are diagonalizable and therefore the theory of generalized B-splines as presented here is based on this

assumption. This implies, however, no loss of generality as the Jordan normal form results in a similar decomposition to compute arbitrary matrix powers A^k and works for all real subdivision matrices. The reader is pointed to Appendix I for the necessary background on the eigendecomposition and Jordan decomposition to clarify the computation of matrix powers.

Using Equation 4.8, the expression for the segments reads

$$\mathbf{x}^k_{i,m,n}(u,v) = B_n M_n A_k X \Lambda^{k-1} X^{-1} \mathbf{Q}_{i,m} \tag{4.9}$$

with all terms left and right of Λ can be multiplied in advance and just the powers of the eigenvalues have to additionally computed compared to Equation 4.4 which defines regular quadrants and therefore the computational costs of both definitions are roughly the same. The *central point* $(0,0)$ is finally given by

$$\mathbf{x}_c = \lim_{k\to\infty} B_n M_n A_k X \Lambda^{k-1} X^{-1} \mathbf{Q}_{i,m} \tag{4.10}$$

what requires to compute the limit of the matrix power Λ^{k-1}. The limit exists for all subdivision algorithms of practical interest for generalized B-splines as the eigenvalues of A are generally $\lambda_0 = 1 > \lambda_i$ for $i \geq 1$.

What remains is to link the parameter value (u,v) to the subdivision level k. A quadrant $\mathbf{x}_{i,m}$ is defined over the unit square $\Sigma = [0,1]^2$. The definition of irregular quadrants is based on its decomposition into a sequence of regular segments $\mathbf{x}^k_{i,m,n}$. Each of them represents only a subset of $\mathbf{x}_{i,m}$ and naturally belongs only to the corresponding subset of Σ which is referred to as Ω^k_n and given by

$$\begin{aligned}
\Omega^k_0 &= [\frac{1}{2^k}, \frac{1}{2^{k-1}}] \times [0, \frac{1}{2^k}] \\
\Omega^k_1 &= [\frac{1}{2^k}, \frac{1}{2^{k-1}}] \times [\frac{1}{2^k}, \frac{1}{2^{k-1}}] \\
\Omega^k_2 &= [0, \frac{1}{2^k}] \times [\frac{1}{2^k}, \frac{1}{2^{k-1}}]
\end{aligned} \tag{4.11}$$

for $n = 1,2,3$. For any parameter $(u,v) \in \Sigma$, the corresponding subdivision level k is computed first by

$$k = \lfloor \min\{-\log_2(u), -\log_2(v)\} \rfloor \tag{4.12}$$

where $\lfloor\rfloor$ denotes the floor function. The segment is finally calculated

$$n = \begin{cases} 0 & 2^k u < \frac{1}{2} \\ 2 & 2^k v < \frac{1}{2} \\ 1 & \text{otherwise} \end{cases} \tag{4.13}$$

from the parameter values u and v as well as the subdivision level k. Note that the calculation

of n requires the subdivision level k and therefore both values are calculated in this order.

The final expression for an irregular quadrant reads

$$\mathbf{x}_{i,m}^{k}(u,v) = \begin{cases} B_n M_n A_k X \Lambda^{k-1} X^{-1} \mathbf{Q}_{i,m} & \text{if} \quad (u,v) \in \Omega_n^k \\ \mathbf{x}_c & \text{if} \quad (u,v) = \mathbf{0} \end{cases} \tag{4.14}$$

where $\mathbf{Q}_{i,m}$ are the initial control points. All other terms are known. The practical implementation is straightforward. For any $(u,v) \neq \mathbf{0}$ the Ω_n^k of the parameter values is identified by Equations 4.12 and 4.13. Afterwards Equation 4.14 is evaluated. The full path from the initial control mesh $\mathbf{Q}^0$ that defines a generalized B-spline to the definition of irregular quadrants is summarized for clarification of the concepts and the notation in Figure 4.16.

4.6. Continuity

The continuity characteristics of a generalized B-splines are inherited from the corresponding tensor-product B-spline. When the control mesh is locally regular, the continuity characteristics of a generalized B-spline of degree p are exactly the same as of a uniform tensor-product B-spline of the same degree, i.e. the surface is C^{p-1} continuous. In the vicinity of irregularities, however, this is generally not true. There, the continuity characteristics depend on the eigenstructure of the subdivision matrix A. The full analysis of subdivision matrices is a complex matter and not scope of this thesis. Fortunately, for many subdivision algorithms applied in the context of generalized B-splines, the simplified analysis given below provides a solid background on the continuity.

Let $\lambda_0, \lambda_1, ...$ be the eigenvalues of the subdivision matrix A which are ordered according to $\lambda_i \geq \lambda_{i+1}$. The minimum requirement is that

$$\lambda_0 = 1 > \lambda_i \tag{4.15}$$

for $i \geq 1$ and λ_0 is usually referred to as the dominant eigenvalue. Basically $\lambda_0 = 1$ is needed for a generalized B-spline to be position continuous which can easily be seen from Equation 4.10 to calculate the central point. The limit exists only for this particular eigenstructure. Fortunately, this can safely be assumed for subdivision matrices of generalized B-splines as the control points of the refined control mesh are calculated as affine combinations of the original control points. This results in all rows of the matrix A to sum up to one, what is referred to as a Markov matrix which is well-known to have this particular eigenstructure. For normal continuity it is further required that

$$\lambda_0 = 1 > \lambda_1 = \lambda_2 = \lambda > \lambda_i \tag{4.16}$$

with $i \geq 3$. The two eigenvalues λ_1 and λ_2 are referred to as the subdominant eigenvalue λ. To

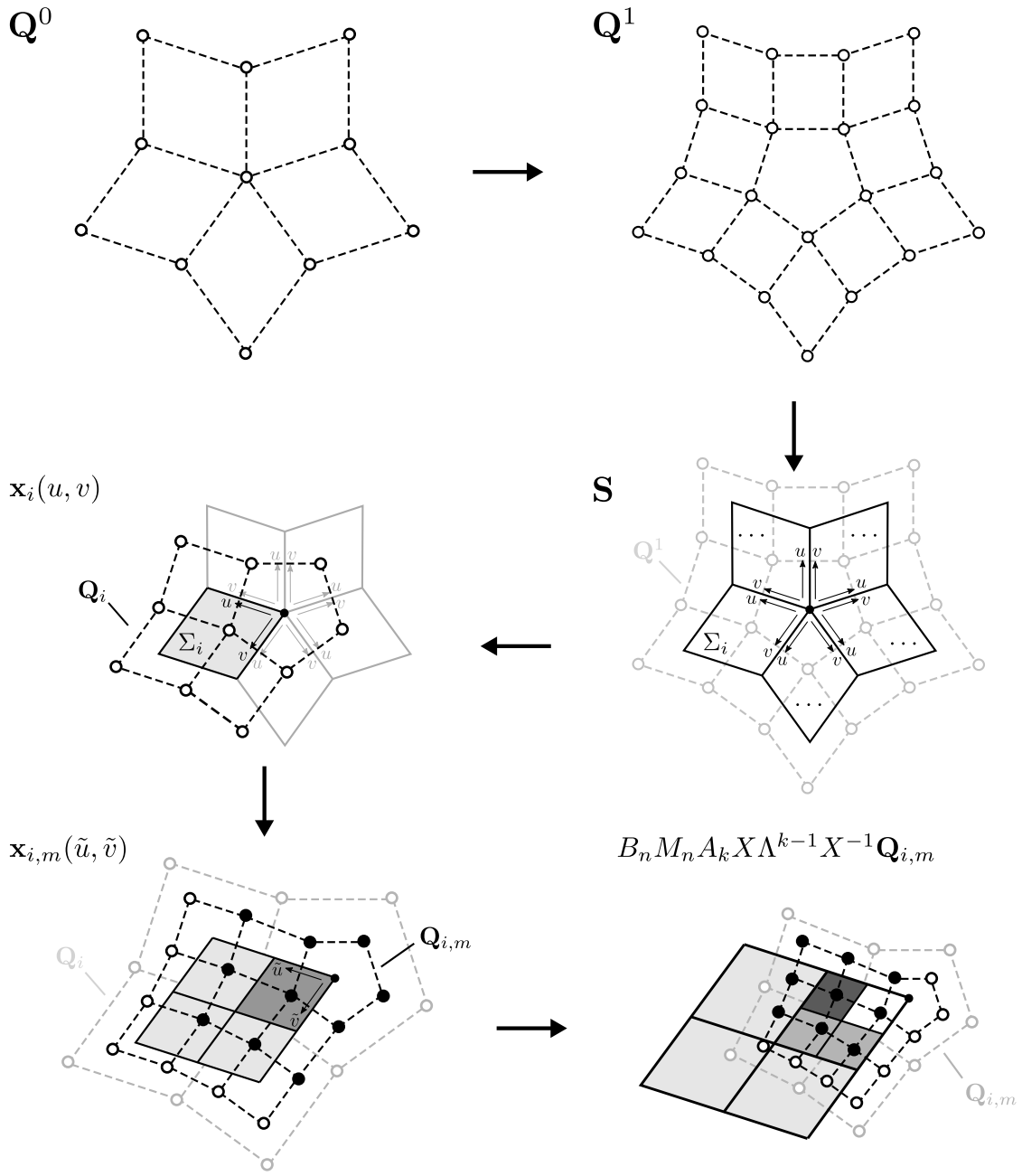


Figure 4.16.: A generalized B-spline is defined over the control mesh $\mathbf{Q}^0$ which is once subdivided to the control mesh $\mathbf{Q}^1$ in order to derive the domain and to define the patches $\mathbf{x}_i$ of the surface. Each patch is decomposed into four quadrants $\mathbf{x}_{i,m}$ which are defined on $\mathbf{Q}_{i,m}$ which may include a single irregularity. These are further decomposed into a sequence of tensor-product B-splines based on subdivision what is formalized by the given expression. The illustrated example shows a generalized B-spline of degree $p = 2$.

construct a normal continuous subdivision algorithm is much more difficult and certainly not all algorithms employed for generalized B-splines have this eigenstructure, but nevertheless for the most popular algorithms such as Catmull and Clark [23] or Doo and Sabin [27] normal continuous variants are well-known.

Curvature continuity, in contrast, is still subject of research and not really solved yet. The findings of Reif [94] respectively Prautzsch and Reif [90] show that this requires a degree $p \geq 6$ for the generalized B-spline for curvature continuity to be possible at all.[4] Instead at least a favorable second-order behavior of next to irregularities is desired. The next best alternative to curvature continuity is bounded curvature which requires the following eigenstructure

$$\lambda_0 = 1 > \lambda_1 = \lambda_2 = \lambda > \lambda_3 = \lambda_4 = \lambda_5 = \lambda^2 > \lambda_i \tag{4.17}$$

of the subdivision matrix A with $i \geq 6$ and where the next three eigenvalues are equal to the square of the subdominant eigenvalue λ^2. However, bounded curvature continuity is also difficult to realize, but a minimal optimization is to keep $\lambda_3 \approx \lambda^2$ close to the square of the subdominant eigenvalue. This is only a simplified representation of the second-order behavior of subdivision algorithms and the reader should refer to Sabin et al. [97] for more information.

The most comprehensive continuity analysis of generalized B-splines is given in the monograph of Peters and Reif [88]. While close to the theory of generalized B-splines as introduced in this thesis, the material is, however, mathematically involved.

4.7. Implementation

The evaluation of a generalized B-spline involves the calculation of $\mathbf{x}(u, v, i)$. The implementation of an evaluation algorithm based on the theory of generalized B-splines is straightforward. A number of optimizations, though, may increase the evaluation speed. The basic implementation issues are briefly reviewed in this section.

Precomputation

The calculation of $\mathbf{x}(u, v, i)$ is basically mapped to the quadrant's definitions. For a regular quadrant this involves only the evaluation of the corresponding tensor-product B-spline, but irregular quadrants require a number of additional matrix multiplications when just Equation 4.14 is implemented. Matrix multiplications are an expensive operation and since the matrices in Equation 4.14 depend only on the local topology of the quadrant's control mesh which in practical applications does not change for a large number of evaluations, they can be mostly avoided by precomputation of certain terms. These terms are

$$\phi_n(u, v) = B_n(u, v) M_n A_k X \tag{4.18}$$

[4] More generally, the degree estimate for a subdivision algorithm to be C^k continuous is $p \geq 2(k + 1)$.

and

$$\hat{\mathbf{Q}}_{i,m} = X^{-1}\mathbf{Q}_{i,m} \tag{4.19}$$

with $n = 1, 2, 3$ which can be precomputed for each quadrant as soon as the topology of $\mathbf{Q}_{i,m}$ is known. Then Equation 4.14 simplifies to

$$\mathbf{x}^k_{i,m}(u,v) = \begin{cases} \phi_n(u,v)\Lambda^{k-1}\hat{\mathbf{Q}}_{i,m} & \text{if} \quad (u,v) \in \Omega^k_n \\ \mathbf{x}_c & \text{if} \quad (u,v) = \mathbf{0} \end{cases} \tag{4.20}$$

The first term $\phi_n(u,v)$ represents a bivariate polynomial for each control point in $\hat{\mathbf{Q}}_{i,m}$. It is recommended to implemented this as a separate method with the coefficients of the polynomials are precomputed and the result, which is a vector of real numbers, is calculated by the method for the parameters u and v. Taking further into account that Λ is a diagonal matrix, the evaluation further simplifies to:

$$\mathbf{x}^k_{i,m}(u,v) = \sum_j \phi_j \lambda_j^{k-1} \hat{\mathbf{q}}_j \tag{4.21}$$

for $(u,v) \in \Omega^k_n$ and

$$\mathbf{x}_c = \phi_0 \hat{\mathbf{q}}_0 \tag{4.22}$$

where ϕ_j are the elements of $\phi(u,v)$, λ_j are the eigenvalues in Λ, and $\hat{\mathbf{q}}_j$ are the control points in $\hat{\mathbf{Q}}_{i,m}$. This is both simple to implement and fast to calculate.

Features

Features such as special rules for boundaries and the like are commonly used to improve the capabilities of subdivision algorithms. They are based on a local modification of the subdivision rules and possibly the basis functions. Therefore the quadrants which are affected by features are based on different subdivision matrices A and A_k and where required also different forms of the basis functions B_n. It is recommend to implement an evaluator for the quadrants which is parameterized with these matrices. When the quadrants are derived from the initial control mesh, for each patch of the surface four evaluators are instantiated while the actual matrices are determined from the local control mesh topology with additional flags and tags to define the features.

Naturally, not the plain form of the matrices should be used, but the precomputed form as introduced before. Therefore the evaluator is in fact parameterized by the coefficients of $\phi_n(u,v)$, the eigenvalues λ_j, and the matrix $\mathbf{X}^{-1}$ rather than A, A_k, and B_n.

Derivatives

The derivatives of an irregular quadrant are

$$\partial_u^\mu \partial_v^\nu \mathbf{x}_{i,m}^k(u,v) = 2^{k(\nu+\mu)} \partial_u^\mu \partial_v^\nu B_n(u,v) M_n A_k X \Lambda^{k-1} X^{-1} \mathbf{Q}_{i,m} \tag{4.23}$$

where ∂_u^μ denotes the μth derivative in u, ∂_v^ν is the νth derivative in u, and the order of differentiation is $\mu+\nu$. When approaching the central point $(0,0)$, the length of the derivative vectors rises quickly and at the central point they are not defined at all. A practical solution, however, is

$$\partial_u^\mu \partial_v^\nu \mathbf{x}_{i,m}^k(0,0) \approx \mathbf{x}_{i,m}^k(\epsilon,\epsilon) \tag{4.24}$$

where ϵ is a sufficiently small number close the machine tolerance.

Eigenstructure

The theory of generalized B-splines is built around the eigendecomposition of the subdivision matrix A. The implementation of a generalized B-spline surface therefore requires the computation of the eigenstructure of the subdivision matrices. For popular subdivision algorithms such as the Catmull-Clark algorithm, analytic solutions are known, see for instance the article of Stam [103]. While the analytic solution for the eigenstructure promises the best accuracy, good results are obtained from the numerical solution as well. Especially since the eigenvalues and -vectors are represented numerically sooner or later anyway.

The author's implementation uses the open-source C++ library *Eigen* to compute the eigenstructure numerically. Due to the limited precision of floating-point numbers, the result is often complex even when it is known to be real from the analytic solution. The imaginary part is, however, small compared to the real part. Good results are obtained with the method to perform all calculations in complex space and to chop off the imaginary part from the final solution. In theory, any two adjacent patches have exactly the same points along their common boundary independent of the actual patch that is being used to compute the point. This property is used to estimate the numerical inaccuracy introduced by this implementation. Common points are computed along common patch boundaries using both patches respectively and the actual distance between both points is considered as the numerical error introduced by the implementation. For the author's implementation the error is about 5e−15 for range of valences $n \leq 32$ and therefore considered as insignificant for practical applications in the context of engineering.

4.8. Example

An example of a cubic generalized B-spline surface is shown in Figure 4.17. The illustrated example is based on the subdivision algorithm described in Chapter 5 and it is defined in

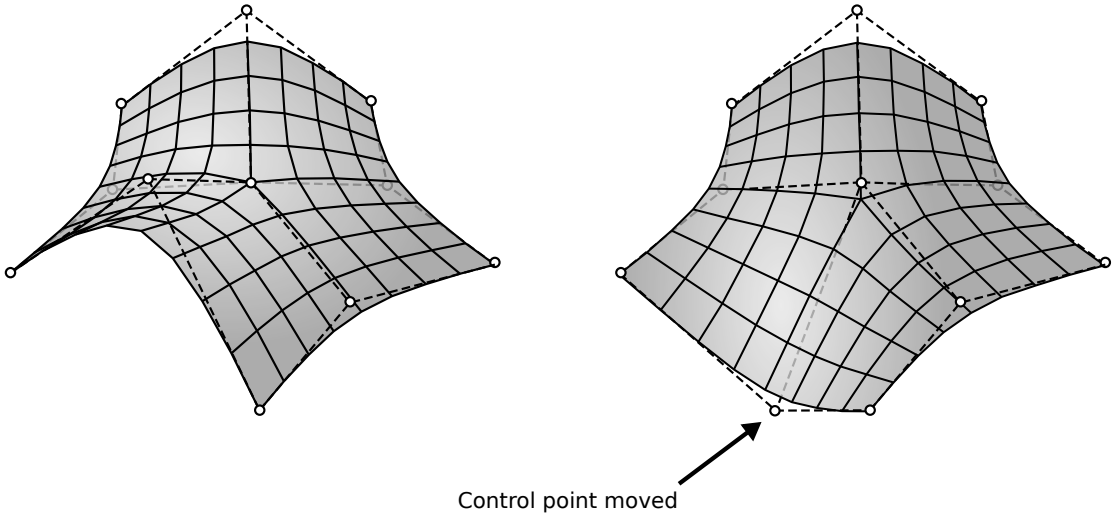


Figure 4.17.: Example of a cubic generalized B-spline surface. The surface is defined on a simple control mesh with only one irregular control point of valence $n = 5$ in the center and the total number of control points is $N = 11$. The result is a five-sided B-spline surface. On the right side, one control point is moved in order to demonstrate that the surface behaves just as a normal B-spline surface with the largest change is in the front, close to the control point being moved, and fades out towards to rear part.

terms of the theory developed before. It is defined on a control mesh with only one irregular control point of valence $n = 5$ in the center with $N = 11$ control points in total. The result is a five-sided B-spline surface.

The first intention of this example is to demonstrate that generalized B-spline surfaces behave just like tensor-product B-splines. Therefore on the right side of Figure 4.17, one control point is moved to another position. As a consequence, the surface is changed, but the largest change is close to the point and fades out with increasing distance. This is exactly the fundamental behavior of B-spline surfaces as identified in Chapter 3. From this point of view, generalized B-splines are just like tensor-product B-splines, but not limited to four-sided surfaces only as illustrated by the example for a five-sided surface.

The second intention refers to representation of complex surfaces. In Chapter 3 it is shown that a n-sided surface can be decomposed into n four-sided surfaces. While this seems to be a reasonable solution to represent complex surfaces with tensor-product B-splines, the modeling based on control points is practically impossible as a consequence of this decomposition. In Figure 3.4 this is illustrated for basically the same surface as being shown in this example. In contrast, however, the generalized B-spline can be defined in terms of its control mesh without being affected by an excessive number of control points or complex continuity constraints.

4.9. Bibliographical notes

- The idea of generalized B-splines originates from the field of subdivision surfaces which cover more than just B-spline surfaces. However, the most popular subdivision surfaces are closely related to B-spline theory and special credit is given to the ground-breaking advances made by Catmull and Clark [23] and Doo and Sabin [28, 27] which for one thing created a new research field, namely subdivision surfaces, and for another thing are the fundamental basis of generalized B-splines.

- For a long time, subdivision surfaces were thought to be limited to polygon meshes, but Stam [103] showed that it is possible to evaluate exact surface properties anywhere on the surface and therefore to represent truly smooth surfaces based on subdivision. This insight is one important key to the theory of generalized B-splines and the central definition of irregular quadrants is entirely based on the results of this work.

- While generalized B-splines would not be possible without the evaluation algorithm of Stam [103], the algorithm alone does not lead to generalized B-splines. The essential part which is not covered is the domain. The formal treatment of subdivision surfaces as being a map from a domain to a surface is contributed by Peters and Reif [88]. Although the intention of this monograph is to provide a comprehensive continuity analysis of subdivision surfaces, this results in the definition of generalized splines which are the basis and motivation of generalized B-splines.

- A variation of subdivision surfaces which is closely related to the idea to generalize tensor-product B-splines are non-uniform subdivision surfaces. In particular, the contributions of Sederberg et al. [100], Sederberg et al. [101], and Cashman et al. [20] are mentioned as they reportedly strive for a generalization of non-uniform rational B-splines (NURBS). These works are, however, focused on the underlying subdivision algorithm rather than providing a formal description of generalized B-spline surfaces.

5 Hull surface representation

5.1. Requirements

A surface representation has to meet a variety of requirements to be qualified for ship hull form design. These requirements are classified into three categories. The first two categories cover the geometrical hull form features that need to be represented. The third category, on the other hand, deals with an appropriate modeling behavior to improve the design process of the hull form.

Surface

Most ship hull forms are complex surfaces. In hull form design, they are decomposed into several regions with different requirements and functions. The geometric classification includes flat and curved regions. The decomposition of hull forms into regions is shown in Figure 5.1 for the example of a container vessel.

- **Flat regions.** For the ease of production and cost efficiency some parts of a hull form are flat. A flat side, a flat bottom, and a flat transom are commonly found.
- **Curved regions.** Major parts of a hull form are curved. This covers uniaxially curved regions such as the bilge area of a parallel middle body and biaxially curved regions such as the forebody and the afterbody. The surface is required to be curvature continuous in these regions.
- **Cylindrical regions.** A special case of curved regions are cylindrical parts of a hull form. The most important example is the bilge area of a parallel middle body. It is often supposed to be the surface of a quarter cylinder.

Features

Features define the appropriate transitions of neighboring regions of a hull form. Three different transitions are required in hull form design: knuckle curves, tangent curves, and smooth curves. An example is shown in Figure 5.1.

- **Knuckle curves.** A hull surface is only position continuous across a knuckle curve. Normals and curvature of the surface do not agree along the knuckle. A variant is an angle curve with a prescribed angle between the normals.

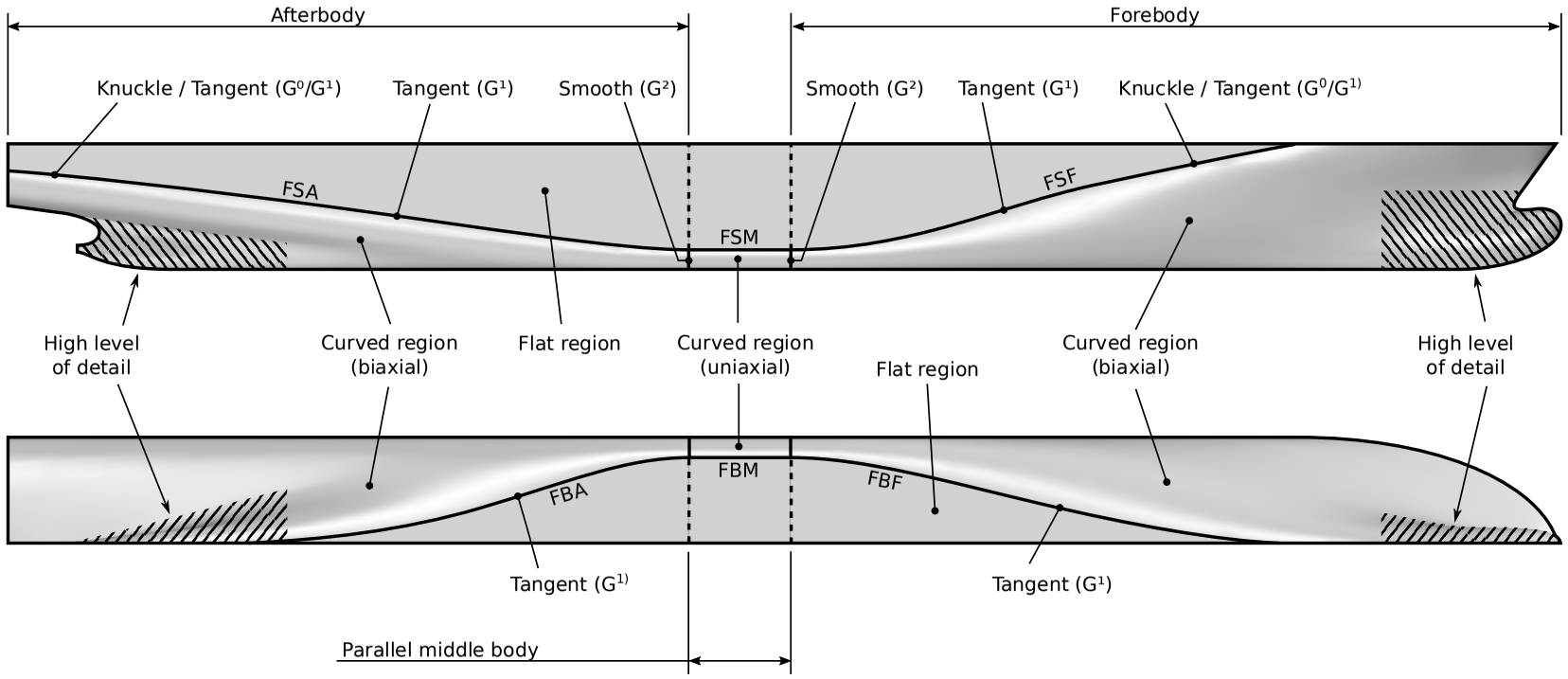


Figure 5.1.: The hull form is decomposed into five regions: a biaxially curved forebody and afterbody, a flat side and bottom, and a uniaxially curved bilge area at the parallel middle body. This region is often supposed to be the surface of a quarter cylinder. The flat of side (FSA, FSM, FSF) is usually a tangent curve with a normal continuous transition (G^1). Close to the deck it may also take the form a knuckle curve what requires only position continuity (G^0). The flat of bottom (FBA, FBM, FBF) is generally a tangent curve over the entire length. The bilge area of the parallel middle body is bounded by smooth curves. Its transition to the forebody and afterbody is characterized by curvature continuity (G^2). The hatched regions of the hull form are considerably more complex than the other parts of this hull.

- **Tangent curves.** A hull surface is only tangent continuous across a tangent curve. It requires the normals to agree, but the normal direction may change along the tangent curve. The curvature generally disagrees on both side of a tangent curve.
- **Smooth curves.** A hull surface is curvature continuous across a smooth curve. In addition to the positions and the normals, the curvature matches along the curve.

Modeling

A common pitfall in ship design is a surface representation that is able to represent hull forms properly, while the definition of the hull surface is a complex matter. The essential properties of a surface representation which are advantageous for modeling are: a simple definition of the form, a robust fairing method, and an easy access to different levels of detail.

- **Simple definition.** A simple hull surface definition does not require any particular background on the mathematics of the surface representation. A further property of a simple definition is a direct link between the input a hull designer provides and the final hull surface. The hull designer instantly needs to know how the input has to look like to obtain the intended shape.
- **Robust fairing.** Fairing is a time-consuming process in all state-of-the-art hull modeling systems. The main reason is the missing link between the input and the result. Most systems are based on a curve network to define the hull surface, but a fair curve network does not necessarily lead to a fair surface. Moreover, the result is sensitive to even small changes of the curve network. Robust fairing calls for direct link between the quality of the input and the quality of the results. The quality of the result is required to be insensitive to small changes of the input.
- **Level of detail.** A major aspect in hull form modeling is to deal with different levels of details. Usually, large areas of a hull form are characterized by a rather simple geometry and only little detail. Other parts feature a lot of details and require a complex geometry. A surface representation that is suitable for hull form modeling has to manage different levels of details. That is to provide only a few degrees of freedom in simple areas, but more degrees of freedom in complex areas.

5.2. Generalized B-spline surface

This study analyzes the application of generalized B-splines for hull form representation and modeling. Based on the subdivision algorithm of Catmull and Clark [23] in conjunction with extensions proposed by Biermann et al. [5], a cubic generalized B-spline surface is constructed which is capable to represent important hull form features.

Feature definition

Features of the surface are defined based on tags which are applied to the vertices and edges of the control mesh. The choice of vertex and edge tags is interdependent.

- **Vertex tags:** Vertices of the control mesh are tagged either *smooth*, *crease* or as *corner*. Vertices which are incident to exactly two crease edges must be tagged as a crease or a corner. Vertices which are incident to three or more creases edges are always tagged as corner vertices.
- **Edge tags:** Edges of the control mesh are tagged as either *smooth* or *crease*. Boundary edges are always creases and interior edges are smooth by default. Edges which are incident to crease or corner vertices must be tagged as creases.

Subdivision rules

The subdivision rules are given for control meshes of arbitrary topology. The control mesh is topologically refined based on the face split. After one subdivision step, it only consists of quadrilateral faces. The following text gives the general form of the subdivision rules which apply to control meshes of any topology and Figure 5.2 shows a simplified form of these rules being valid for quad meshes only. The latter is the basis to construct the subdivision matrices A and A_k.

Face points: For each face $\mathcal{F}_i$ of the control mesh $\mathcal{M}^k$ a new control point

$$\mathbf{f}_i = \frac{1}{n} \sum_{i=1}^{n} \mathbf{q}_j \tag{5.1}$$

is computed as the average of the n control points $\mathbf{q}_j \in \mathcal{F}_i$ defining the face. After one subdivision step, all faces are defined by exactly four control points and therefore the rule is simplified to

$$\mathbf{f}_i = \frac{1}{4} \left(\mathbf{q}_1 + \mathbf{q}_2 + \mathbf{q}_3 + \mathbf{q}_4 \right) \tag{5.2}$$

with $\{\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{q}_4\} \in \mathcal{F}_i$ and the weights are simply $\frac{1}{4}$ for all control points as illustrated in Figure 5.2 for a number of cases.

Edge points: For each edge $\mathcal{E}_i$ of the control mesh $\mathcal{M}^k$ a new control point $\mathbf{e}_i$ is computed based on the tags of the edge and the incident vertices. For a *smooth edge* a new control point

$$\mathbf{e}_i = \omega_1 \mathbf{q}_1 + \omega_2 \mathbf{q}_2 + \frac{1}{4} \left(\mathbf{f}_1 + \mathbf{f}_2 \right) \tag{5.3}$$

is computed as the weighted average of the control points $\mathbf{q}_1, \mathbf{q}_2 \in \mathcal{E}_i$ defining the edge and the face points $\mathbf{f}_1$ and $\mathbf{f}_2$ of the two faces $\mathcal{F}_1$ and $\mathcal{F}_2$ incident to the edge $\mathcal{E}_i$. The choice of weights ω_1 and ω_2 depends on the tags of $\mathbf{q}_1$ and $\mathbf{q}_2$. If both control points are (not)

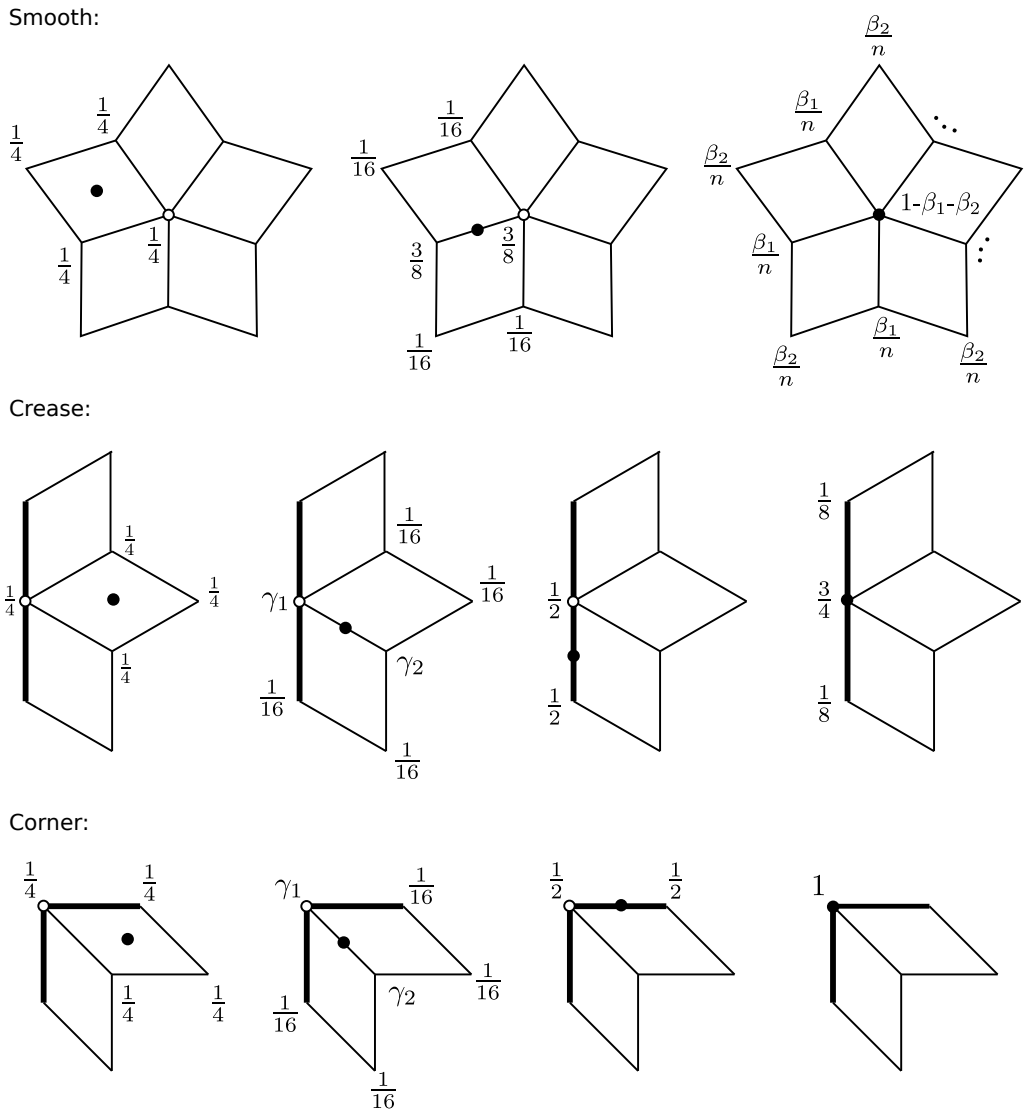


Figure 5.2.: Subdivision rules for a cubic generalized B-spline surface which may have creases and corners as features. The rules depend on the tags of the edges and vertices. Bold lines denote crease edges, thin lines smooth edges, and the vertex tag belongs to the highlighted point (empty circle). The black point is the control point to compute.
Top row: Subdivision rules in the vicinity of smooth interior vertices. The vertex point depends on the weights $\beta_1 = \frac{3}{2n}$ and $\beta_2 = \frac{1}{4n}$ with n being the vertex valence which is $n = 5$ for the shown control mesh.
Middle row: Subdivision rules in the vicinity of crease vertices. The weights to compute smooth edge points are $\gamma_1 = \frac{1}{2}\cos^2(\frac{\pi}{2n})$ and $\gamma_2 = \frac{1}{2}\sin^2(\frac{\pi}{2n})$ with n being the sector valence which is $n = 3$ for the shown control mesh.
Bottom row: Subdivision rules in the vicinity of corner vertices. The weights to compute smooth edge points are $\gamma_1 = \frac{1}{2}\cos^2(\frac{\pi}{4n})$ and $\gamma_2 = \frac{1}{2}\sin^2(\frac{\pi}{4n})$ with n being the sector valence which is $n = 2$ for the shown control mesh.

smooth the weights are $\omega_1 = \omega_2 = 1/4$. If $\mathbf{q}_1$ is a crease and $\mathbf{q}_2$ is smooth, the weights are $\omega_1 = \frac{1}{2}\cos^2(\frac{\pi}{2n})$ and $\omega_2 = \frac{1}{2}\sin^2(\frac{\pi}{2n})$. Finally, if $\mathbf{q}_1$ is a corner and $\mathbf{q}_2$ is smooth, the weights are $\omega_1 = \frac{1}{2}\cos^2(\frac{\pi}{4n})$ and $\omega_2 = \frac{1}{2}\sin^2(\frac{\pi}{4n})$. Here n denotes the sector valence which is the number of faces between the two creases which enclose the edge $\mathcal{E}_i$. The simplified notation for quad meshes is shown in Figure 5.2. For a *crease edge* of the control mesh a new control point

$$\mathbf{e}_i = \frac{1}{2}\mathbf{q}_1 + \frac{1}{2}\mathbf{q}_2 \tag{5.4}$$

is computed as the average of the control points $\mathbf{q}_1, \mathbf{q}_2 \in \mathcal{E}_i$ defining the edge

The edge rules are proposed by Biermann et al. [5] to ensure normal continuity at creases. A simplified notation of the rules is chosen for this material and the weights of the corners are modified by assuming $\alpha = \frac{\pi}{2}$ as the angle between the two crease edges spanning the sector at corners. The reader may refer to the original source for clarification.

Vertex points: For each vertex $\mathcal{V}_i$ of the control mesh $\mathcal{M}^k$ a new control point is computed. The rule depends on the tag of the control point $\mathbf{q}_i \in \mathcal{V}_i$ associated to the vertex. For a *smooth vertex* a new control point

$$\mathbf{v}_i = \frac{1}{n^2}\sum_{i=j}^{n}\mathbf{f}_j + \frac{1}{n^2}\sum_{j=1}^{n}(\mathbf{q}_i + \mathbf{q}_j) + \frac{n-3}{n}\mathbf{q}_i \tag{5.5}$$

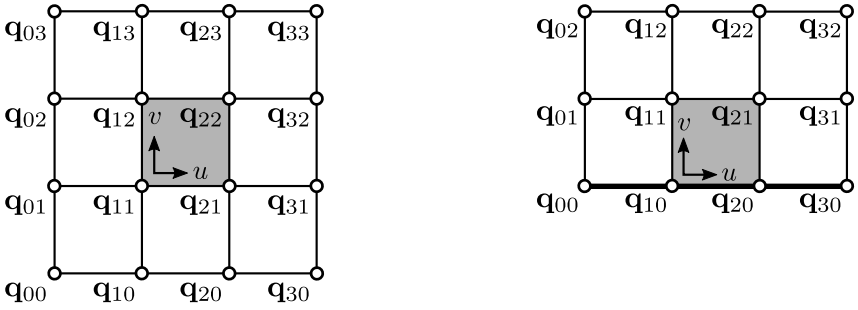
is computed with $\mathbf{f}_j$ are the face points of the faces incident to the vertex, $\mathbf{q}_j$ are the surrounding vertices. For a *crease vertex* a new control point

$$\mathbf{v}_i = \frac{1}{8}\mathbf{q}_{i\text{-}1} + \frac{3}{4}\mathbf{q}_i + \frac{1}{8}\mathbf{q}_{i+1} \tag{5.6}$$

is computed as the weighted average of the adjacent crease vertices $\mathbf{q}_{i\text{-}1}$, $\mathbf{q}_{i+1}$ and the position of the original vertex $\mathbf{q}_i$. A *corner vertex* interpolates the original vertex with $\mathbf{v}_i = \mathbf{q}_i$.

Basis functions

A generalized B-spline surface is patch-wise evaluated in terms of tensor-product B-splines. Two configurations are differentiated: interior patches which are defined on 4×4 control points and crease patches which are defined on only 4×3 control points as illustrated in Figure 5.3. The basis functions for the two patch types are defined in terms of the standard



(a) Interior patches (b) Crease patches

Figure 5.3.: A generalized B-spline surface is patch-wise evaluated in terms of tensor-product B-splines. Two configurations are differentiated: interior patches which are defined on 4×4 control points and crease patches which are defined on only 4×3 control points.

basis functions for uniform cubic B-splines which are

$$\begin{aligned} N_0(u) &= \frac{1}{6} - \frac{1}{2}u + \frac{1}{2}u^2 - \frac{1}{6}u^3 \\ N_1(u) &= \frac{2}{3} - u^2 + \frac{1}{2}u^3 \\ N_2(u) &= \frac{1}{6} + \frac{1}{2}u + \frac{1}{2}u^2 - \frac{1}{2}u^3 \\ N_3(u) &= \frac{1}{6}u^3 \end{aligned} \tag{5.7}$$

for $u \in [0, 1]$. The basis functions for the interior patches are

$$N_{ij}(u, v) = N_i(u)N_j(v) \tag{5.8}$$

for $i, j \in \{0, 1, 2, 3\}$. For the crease patches the basis functions are

$$\begin{aligned} N_{i0} &= N_i(u)\left(2N_0(v) + N_1(v)\right) \\ N_{i1} &= N_i(u)\left(-N_0(v) + N_2(v)\right) \\ N_{i2} &= N_i(u)N_3(v) \end{aligned} \tag{5.9}$$

for $i \in \{0, 1, 2, 3\}$. The corresponding matrix notation of the patches is given in Equation 3.5. The choice of the basis function for crease patches reflects the boundary behavior of ghost points. Utilizing this definition of the basis functions, however, avoids to compute the ghost points explicitly as proposed by Lacewell and Burley [56].

5.3. Discussion

The surface is generally curvature continuous everywhere, but at irregular knots it is only normal continuous. It is therefore well-suited to represent ship hull forms. It is possible to represent uniaxially and biaxially curved regions as well as flat regions within a single surface. An exact representation of cylindrical regions is not possible.

The representation of hull form features is limited compared to the requirements given in Section 5.1. A knuckle curve is introduced to the surface with a chain of crease edges. A crease may end anywhere in the mesh to fade out knuckles smoothly. Creases provide an intuitive way to define knuckles, but the Gaussian curvature is always zero on a knuckle. This is potentially unfavorable in hull form modeling and affects the implementation of tangent curves. Several knuckle curves may meet at a common point, but the angle between two consecutive knuckles is limited to $\alpha \leq 180°$. Tangent curves are realized on the generalized B-spline surface similar to knuckles. A chain of crease edges identifies the tangent curve. In addition, all faces incident to the crease edges are restricted to a common plane. This guarantees a common normal vector on the crease. However, the curvature component across a crease is always zero. The reason is that the extensions of Biermann et al. [5] are essentially equivalent to the end conditions for uniform cubic B-splines based on ghost points. The nature of ghost points is best explained in the curve setting. A uniform cubic B-spline curve is extended by an additional control point on both sides. These points are neither visible nor are they accessible for the user. Hence they are called ghost points. The automatically chosen position of the ghost points ensures that the curve starts (ends) at the first (last) visible control point and the curve is tangent to first (last) span of the control polygon. Both properties are desirable for modeling and mimic the behavior of open uniform B-spline curves with multiple knots at the ends of the knot vector. However, a third property of this construction is that the curvature at both ends of the curve is always zero. The extensions of Biermann et al. [5] are essentially equivalent to these end conditions in the surface setting and therefore the curvature component across features and the boundary is always zero. An example where only unsatisfactory results can be obtained using this construction of feature curves is shown in Figure 6.4.

In comparison to the general definition of a tangent curve this implementation is quite limited. As the curvature is always zero across a tangent curve the surface is G^2 continuous as opposed to the general definition which requires only G^1 continuity. Furthermore, it is not possible to change the normal direction along the tangent curve. Nevertheless, this construction is useful to realize two major applications of tangent curves in hull form modeling: the flat of side and the flat of bottom. A smooth curve on the generalized B-spline surface is simply identified by a chain of smooth edges.

5.4. Bibliographical notes

- Sections 5.1 and 5.3 were previously published by the author in:

  Sebastian H. Greshake and Robert Bronsart. Application of subdivision surfaces in ship hull form modeling. *Computer-Aided Design*, 100:79 – 92, 2018. ISSN 0010-4485. doi: https://doi.org/10.1016/j.cad.2018.03.004

- Section 5.2 is a revised and extended version of the material previously published by the author in:

  Sebastian H. Greshake and Robert Bronsart. Using subdivision surfaces to address the limitations of B-spline surfaces in ship hull form modeling. In *Proceedings of the 13th International Symposium on Practical Design of Ships and Other Floating Structures (PRADS 2016)*, 2016

- The subdivision rules presented in Section 5.2 are based on the popular subdivision algorithm of Catmull and Clark [23] which generalizes knot refinement for uniform bicubic tensor-product B-splines. The features are realized based on the weights proposed by Biermann et al. [5].

This page intentionally left blank.

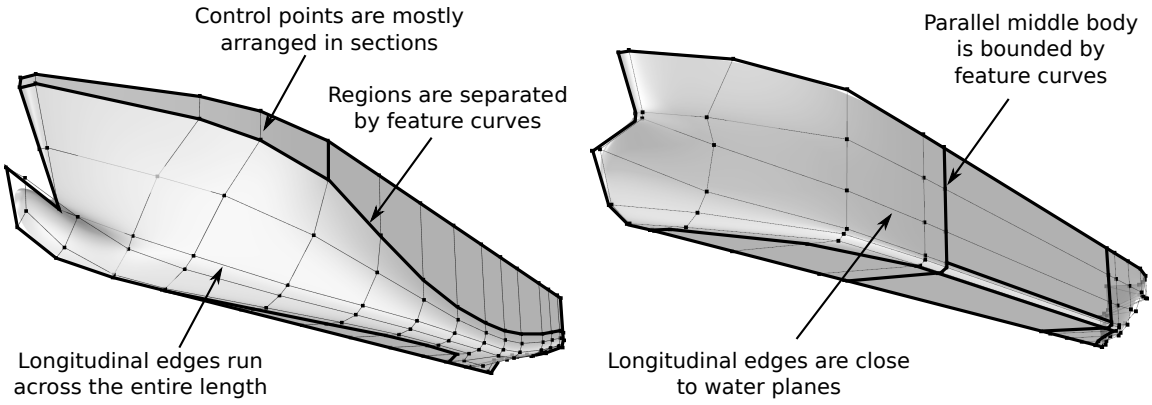
6 Ship hull form modeling

Conventional ship hull form modeling is based on *interpolation* as described in Chapter 2. The hull designer specifies a curve network to describe the hull form and the curves are smoothly interpolated by a set of surface patches to obtain a surface representation. It is well-known that interpolation tends to oscillation and this makes modeling, and especially fairing, of hull forms a time-consuming process. In contrast, the definition of a B-spline surface in terms of a control mesh is based on *approximation*. As pointed out in Chapter 3, a B-spline surface is a smooth approximation of its control mesh. The surface does not oscillate more often than its control mesh what makes surface fairing a simple and fast process. These favorable characteristics of B-splines cannot be utilized with tensor-product surfaces. Tensor-product B-splines are limited to four-sided surfaces and as a consequence complex surfaces such as hull forms are composed of several patches which are continuously fitted to each other. This results in a high number of control points because each patch requires a minimum number of control points based on its degree. A sufficient number of control points is required to formulate the continuity constraints between neighboring patches. Naturally, it is difficult to use a control mesh for modeling and fairing when the number of control points is high and many points are constrained by continuity requirements. Generalized B-splines, as introduced in Chapter 4 are not affected by this problem. They are capable to represent hull forms with a single surface. The control mesh requires only a small number of control points and is therefore a reasonable option for modeling.

This chapter serves the explanation of how hull form modeling based on a control mesh works. The cubic generalized B-spline surface described in Chapter 5 is used as the underlying surface representation. It is discussed that the structure of a control mesh and the workflow to define the control points are essentially the same as for curve networks. A number of examples is presented to show that the proposed surface representation and modeling methodology is capable to successfully define different vessel types commonly found in practice.

6.1. Structure of the control mesh

A typical control mesh for the hull form of a fast vessel is shown in Figure 6.1a. Features are used to separate different regions of the hull form. They are defined with crease edges. Moving the control points on one side of a crease does not affect the surface's shape on the other side. Therefore the hull surface is not only subdivided into regions, also their definition and shape is made independent of each other. The control points are mostly arranged in



(a) Fast vessel (b) Slow vessel

Figure 6.1.: Typical structure of a control mesh to define hull forms. Feature curves separating different regions of the hull form are shown in bold. The control points are mostly arranged in sections in order to support the naval architect's view on a hull form. The vertical distribution of the control points corresponds to the required control on the surface shape. The longitudinal edges run across the entire length in order to keep the control mesh as regular as possible.

sections. This arrangement is not a necessity, but the emphasis on sections matches the naval architect's view on a hull surface. The vertical distribution of the control points of a section corresponds to the required control. More points are placed where more control on the surface shape is needed. It is recommended to keep the control mesh as regular as possible. The faces of the control mesh should be quadrilateral and irregular points should be avoided if possible. Therefore the number of control points of a section should be constant for most parts of the hull. This results in longitudinal edges that are running all onto the bilge radius of the main section. The control mesh of the curved region between the flat side and flat bottom becomes mostly regular.

A control mesh for the hull form of a slow vessel is shown in Figure 6.1b. In contrast to a fast vessel, the hull form features a long parallel middle body. The extent of the flat side is limited to the parallel middle body, but the flat bottom covers almost the entire length of the vessel. The forebody is rather simple with similar waterlines and a less accentuated bulbous bow. In general, the same principles apply to control meshes for fast and slow vessels: features separate the hull form into regions, control points are arranged in sections, and quadrilateral faces are preferred for the control mesh. The longitudinal edges do not uniformly run onto the bilge radius, but may also run into the flat of side. All forward sections have the same number of control points and the mesh is almost regular. The rather simple shape of the forward sections allows to keep the longitudinal edges close to water planes.

The structure of the control meshes is very similar to the structure of curve networks which are employed for hull surface definition. The same principles apply to curve networks and

control meshes. Therefore any hull designer experienced in curve-based modeling should be able to generate control meshes in this context. The similarity of control meshes and curve networks and the long tradition of curve networks in hull design gives rise to the expectation that control meshes would blend well into hull form modeling.

6.2. Feature curves

Crease edges are an essential part of the control mesh. They are used to define a set of curves that is interpolated by the surface. A chain of crease edges represents the control polygon of a B-spline curve. For the generalized B-spline surface that is employed in this study, this is a uniform cubic B-spline curve. In the presence of irregular control points, the same concepts as described in Chapter 4 apply in order to obtain a closed-form solution for the curve's representation.

Referring the example from Figure 6.2, the top row shows the crease edges of the control mesh and the corresponding curves. Each curve is solely defined by the control polygon formed by the associated crease edges and is not affected by the rest of the control mesh. The surface interpolates the curves defined by the crease edges. The latter characteristic provides a simple mechanism to handle feature curves in hull surface modeling. After an appropriate control mesh structure is specified, the hull designer places the control points of the crease edges. Thereby the feature curves of the hull surface are defined. The hull designer then works with the remaining control points as described in the next section, what neither changes the feature curves, nor affects the interpolation of the feature curves by the final surface.

6.3. Modeling with control points

In hull form design the control mesh is frequently considered as inappropriate for modeling because the surface only approximates the control points, but generally does not interpolate them. Interpolation is considered as direct control, whereas approximation seams to offer only indirect control over the surface. This section is devoted to resolve this popular myth of interpolation.

Consider a set of points that is going to be interpolated by a B-spline curve. An infinite number of curves passes through the points. Some of them are fair, but others just wiggle through the points. What kind of curve is obtained using interpolation depends on the number of points, their relative position to each other, and the parametrization. The combination of these factors lacks a clear link between the input and the result. This is certainly the nature of indirect control rather than direct control. It is difficult to estimate how the curve that interpolates a given set of points will look like. In addition, even small changes of the input parameters may result in very different results. Therefore it is not only a matter of

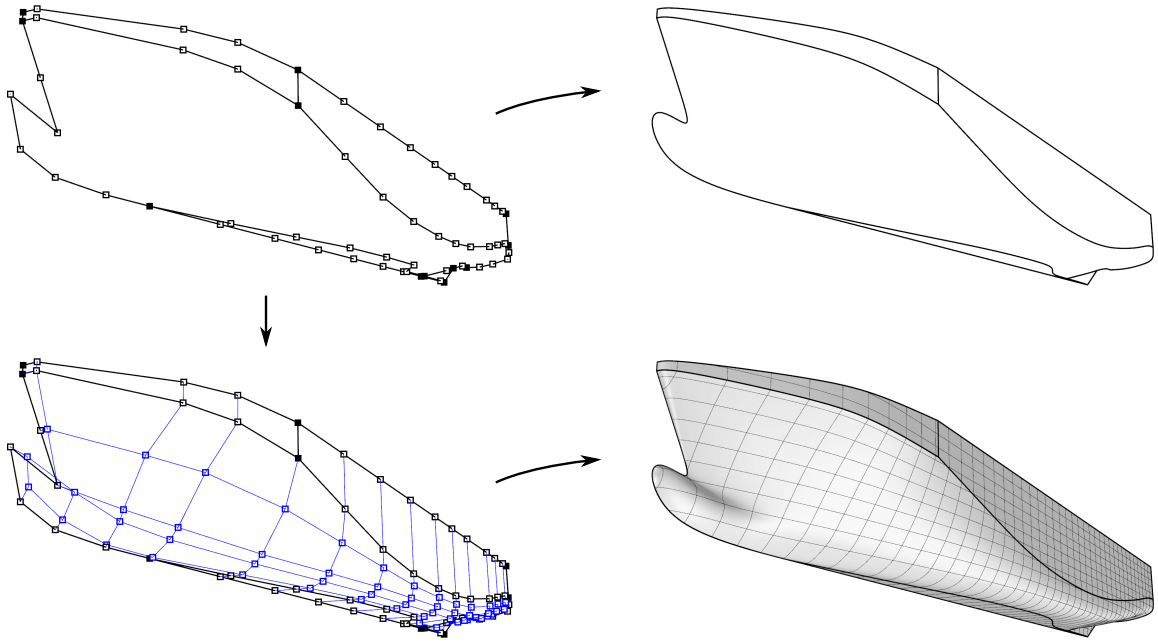


Figure 6.2.: Handling feature curves based on the control mesh. Top row: Only the crease edges of the control mesh are shown. They are further divided into chains, where each chain starts and ends at a corner point (filled squares). Each chain represents the control polygon of a single B-spline curve. Altogether, the crease edges define a network of linked B-spline curves as shown on the right side. Bottom row: In general, the hull designer will at first define the feature curves before designing the hull surface, leaving the feature curves interpolated by the surface unchanged. In terms of control mesh, the designer first places the control points (black squares) of the crease edges and then works with the remaining control points (blue squares). The surface interpolates the curve network defined by the crease edges.

trial and error to define a fair B-spline curve based on interpolation, but the result also lacks robustness.

Modeling with control points is the very reverse. There is only a single curve that belongs to a given control polygon. The curve closely approximates the control polygon. This is more technically known as the convex hull property of B-splines. In addition, the curve does not oscillate more often than its control polygon, usually referred to as the variation diminishing property. As a consequence, the control polygon supplies a clear link to the curve's shape. Even if the curve is not shown, it is obvious for a designer how the curve will look like. Moreover, small changes of the control polygon imply only small changes of the curve as required for robustness. In contrast to interpolation, each control point influences the curve only locally. This is an essential property as part of modeling.

The previous discussion covers B-spline curves, but all arguments apply equally well to

B-spline surfaces. Obviously, control points are most qualified for modeling, but they are still rarely used for hull surface modeling. Naturally, this begs the question for the reason. The representation of hull forms based on tensor-product B-spline requires several patches. As a consequence, the number of control points is high while many points are constrained by continuity requirements. In contrast, hull form representation based on generalized B-spline does not require patches. A single coarse control mesh is sufficient to define a hull surface what makes control meshes an option in hull design.

In order to start using control points for modeling, it is sufficient to know that the surface is a smooth approximation of the control mesh. Refer to Figure 6.3 for an example of modeling the forebody of a fast vessel. The left side of the figure shows the control mesh and a first impression of the surface. It can be easily seen that the surface is a close approximation of the control mesh, but it is rather difficult to capture the details which are relevant for the naval architect. Therefore the right side shows the same surface in conjunction with uniformly distributed sections and waterlines. They are cross-sections of the surface which are generated independently from the control mesh structure. The hull form features a bulbous bow and a slender forebody. The bulbous bow is smoothly integrated into the forebody, requiring bulb-shaped sections close to the forward perpendicular. Subsequently, V-shaped sections are intended which gradually change to U-shaped sections towards the middle body. The control points are placed accordingly. At the forward perpendicular, the control points reflect the bulb shape. They are concentrated in the lower part of the hull in order to provide a sufficient outline of the bulb shape. Halfway between the forward perpendicular and the main section, the control points are basically placed on a straight line as supposed for V-shaped sections. Towards the middle body, the control points are placed according to a U-shaped section.

The previous discussion shows that the control mesh is a simple method to define the shape of B-spline surfaces. The surface approximates the control mesh. This is sufficient for most parts of a hull form where the hull designer typically has a principle idea of the hull shape, but up to certain limits the precise shape is freely chosen by the designer. In contrast, other characteristics of the hull form have to be defined very precisely. Figure 6.4 illustrates this by taking the example of a vessel's main section. The left side shows a section of a control mesh for a hull form. For clarification the rest of the control mesh is omitted. The section consists of eight control points, where the first two points and the last two points belong to feature curves. The right side shows the corresponding section of the hull form.

The section interpolates the first two control points and the last two control points and the section takes the form of straight line between these points. The main dimensions of the section are shown and coincide with the control mesh. Therefore it is possible to define these dimensions precisely in terms of the control mesh.

In addition, the ability to represent circular bilges is illustrated. The control mesh features four additional control points between the FOS and FOB. This setup matches the structure of the control mesh of the RoRo vessel shown in Figure C.3. With the control points positioned

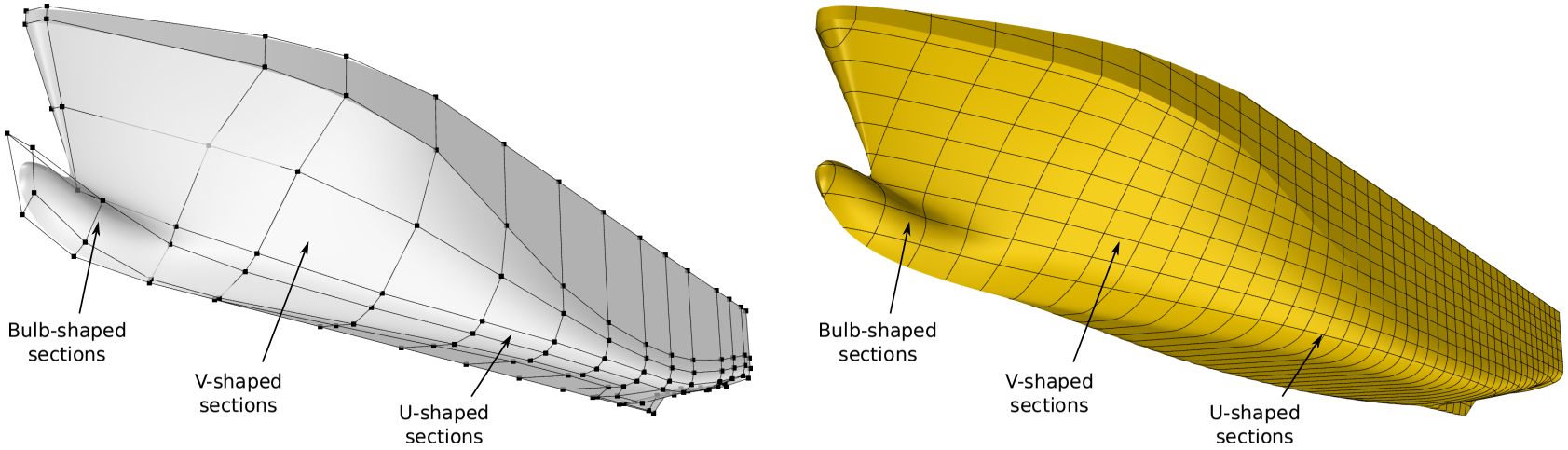


Figure 6.3.: Definition of the forebody in terms of the control mesh. The left side shows the control mesh and a first impression of the hull surface. The right side shows a second visualization of the hull surface which is extended by uniformly distributed cross-sections of the surface. The cross-sections help the naval architect to capture the relevant details of the hull form. The forebody features V-shaped sections which gradually change to U-shaped sections towards the middle body. Accordingly, the control points are placed on an imaginary V which gradually changes to a U. The rate at which the control point arrangement changes determines how fast the section-type changes.

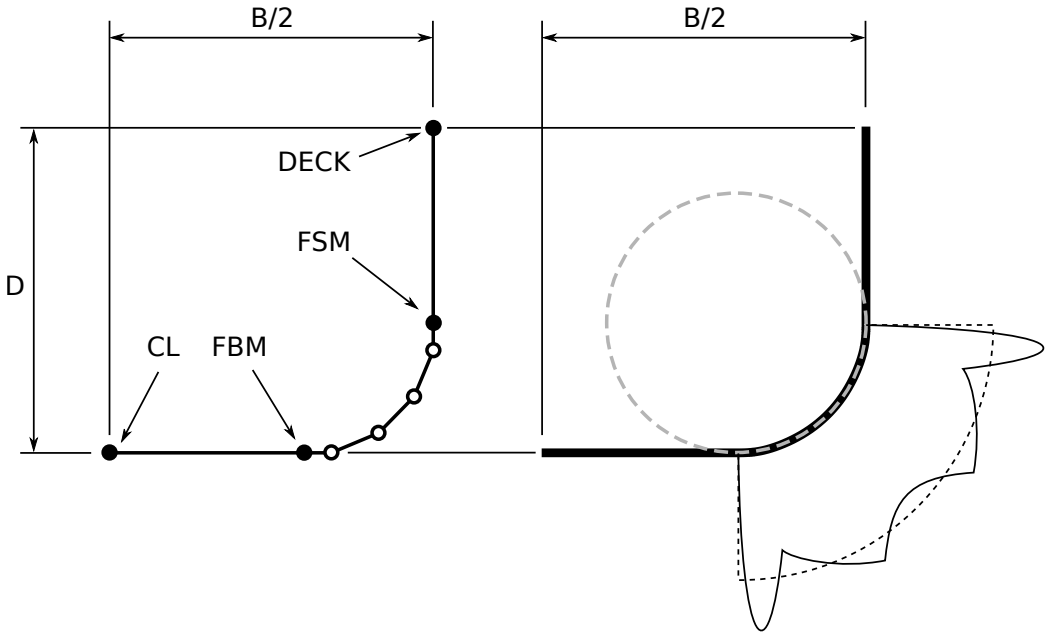


Figure 6.4.: Definition of the main section regarding the control mesh. Left: A section of the control mesh of the hull surface. The bold points denote control points of crease edges. Right: The corresponding cross-section of the hull surface. The section connects the bold control points on a straight line. This allows to define the main dimensions of the hull form precisely in terms of the control mesh. The section closely approximates a quarter-circle in the bilge area with a maximum radial deflection of 0.3%. The curvature plot (solid) oscillates around the constant curvature of the quarter-circle (dashed) and is zero at the ends.

as shown on the left side, the section approximates the circle with a maximum radial deflection of 0.3%. The curvature is plotted along the section. The solid line shows the curvature of the actual section and the dashed line shows the curvature of the intended circle. The curvature oscillates around the constant curvature value of the circle. Moreover, the curvature vanishes at the FOS and FOB whereas the ideal curvature plot takes the form of a step function. Unfortunately, the curvature does not only vanish close to surface features, but also introduces subsequent oscillations to the curvature flow. From the authors' investigations it seems that this can only be improved when the accuracy of the geometrical approximation is sacrificed.

6.4. Irregular control points

Much is said about the general structure of the control mesh and the use of control points for modeling. However, the possibility to define irregular control points is not discussed so far.

Irregular control points are a useful extension of B-spline surfaces in the context of hull form modeling. Usually two circumstances require the use of irregular control points in this

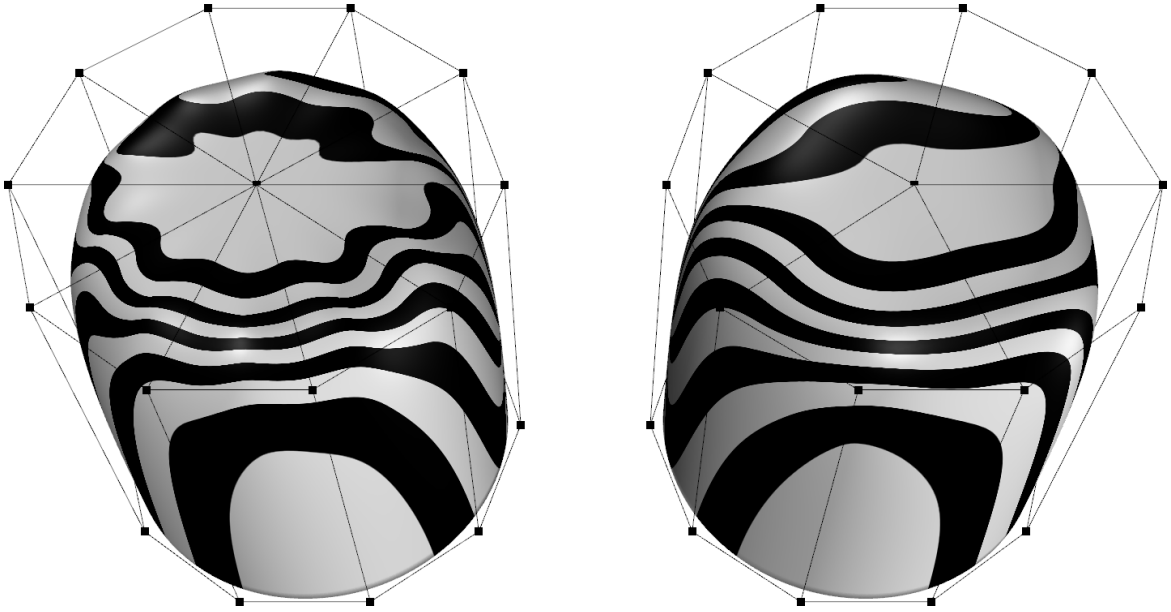


Figure 6.5.: Impact of the control mesh topology on the surface quality. Both surfaces are defined by the same set of control points, but the topology of the control meshes differs. Left: All outer control points are directly connected to the central control point. This yields a valence of $n = 10$ at the central control point and the surrounding faces are triangular. The reflection lines indicate a poor surface quality. Right: Only every second outer control point is connected to the central control point. This reduces the valence at the central control point to $n = 5$ and the surrounding faces are quadrilateral. The quality of the surface is improved.

context: the integration of typical hull form features and the local refinement of the control mesh. Examples for both scenarios are given below. In order to maintain a high surface quality in the presence of irregular control points, it is essential to keep the control mesh as regular as possible. A regular control mesh only consists of quadrilateral faces. The valence of interior control points is always $n = 4$. Crease edges divide the control mesh from a topological point of view. The regular valence of crease control points is $n = 3$ on both sides, the regular valence of corner control points is $n = 2$ for the respective sector. Whenever irregular control points are used, the valence should be as close as possible to the regular case. For clarification, the relation of the surface quality and the control mesh topology is illustrated in Figure 6.5. The figure shows two surfaces which are defined by the same set of control points, but the topology of the control mesh differs. Although both surfaces are geometrically similar, the reflection lines indicate a difference of the surface quality. The surface shown on the left side of Figure 6.5 suffers from a poor topology of the control mesh. The valence of $n = 10$ at the central control point is far off the regular value of $n = 4$. The second variant of the control mesh, shown on the right side of Figure 6.5, improves the quality of the surface. The valence of the central control points is reduced to $n = 5$ and therefore

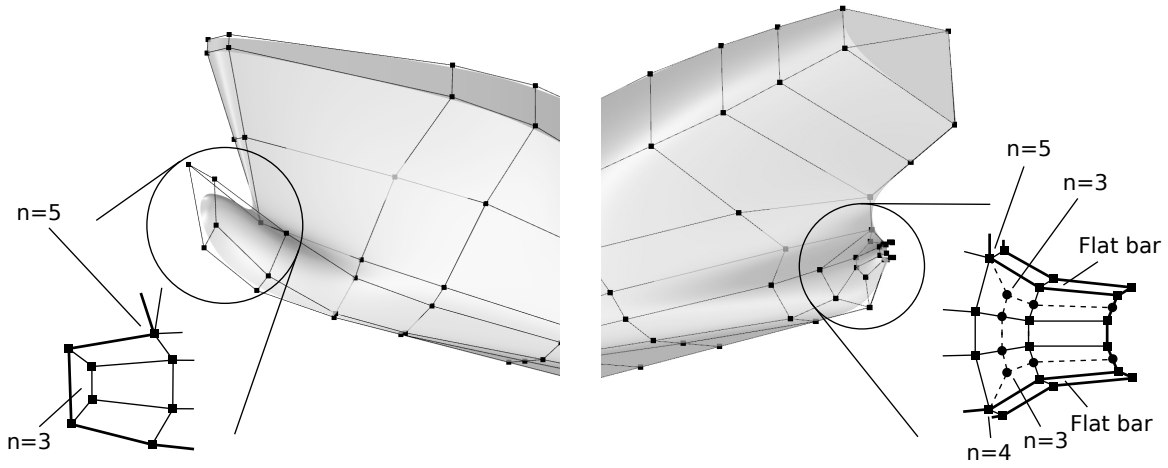


Figure 6.6.: Two examples where irregular control points are used to define typical hull form features. Left: The highlighted subset of the control mesh defines the bulbous bow. Bold lines denote crease edges and thin lines denote smooth edges. A crease control point of valence $n = 5$ is used to connect the bulbous bow to the forebody. Two interior control points of valence $n = 3$ are used to end the longitudinal edges. Right: The global control mesh is shown with solid lines. Again bold lines denote crease edges and thin lines denote smooth edges. The number of control points is not sufficient to model the stern tube properly therefore two interior control points of valence $n = 3$ are utilized to refine the control mesh. In addition the valence of the two crease control points is raised by one compared to the initial situation. Two parallel chains of crease edges are used at the boundaries of the control mesh to define a flat bar, see also Figure 6.8.

close to the regular value. Moreover, the faces incident to the central control point all are quadrilateral as opposed to the initial mesh.

An important example where irregular control points are a useful extension for hull form modeling is the bulbous bow. Its integration into the hull surface usually involves three irregular control points as shown on the left side of Figure 6.6. One irregular control point of valence $n = 5$ is used to connect the bulbous bow to the forebody. This control point belongs to a crease edge of the control mesh where regular control points have a valence of $n = 3$. The face above this control point is triangular. Two interior control points of valence $n = 3$ are used at the tip of the bulbous bow. The two longitudinal edges terminate at these points and triangular faces are avoided.

Irregular control points always affect the quality of the surface. In case the valence of the irregular control points is kept close to the regular case, however, the effect can be minimized. For the given control mesh of the bulbous bow, Figure 6.7 analyzes the impact of the irregular control points on the quality of the hull surface. The boundary control point of valence $n = 5$ causes a local distortion of the reflection lines. The two interior control points of valence $n = 3$

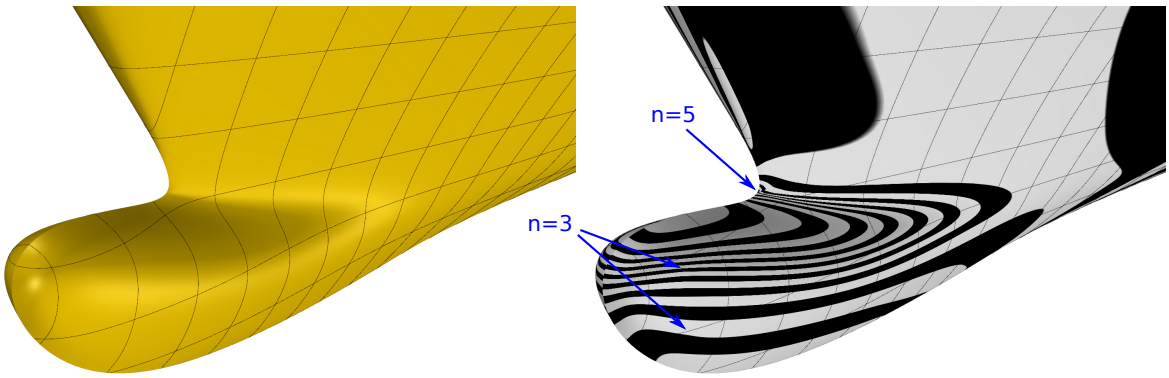


Figure 6.7.: Representation of a bulbous bow. Left: Detailed view of the bulbous bow defined by the irregular control mesh shown on the left side of Figure 6.6. Right: Impact of the irregular control points on the quality of the hull surface. The boundary control point with valence $n = 5$ causes a small distortion of the reflection lines as indicated by the arrow. The two control points of valence $n = 3$ at the tip of the bulbous bow do not have any visible effect at this scale.

which are used at the tip of the bulbous bow do not have any visible effect on the reflection lines at this scale.

Another application of irregular control points is the local refinement of the control mesh. This is important for hull form modeling because most parts of a hull surface are rather simple, but other parts are quite complex. Naturally, complex shapes require more control points to be defined. Therefore the goal is to keep the overall number of control points small, but having an increased appropriate number of control points where necessary. An example is the locally refined control mesh of the stern tube shown on the right side of Figure 6.6. The number of control points given by the global structure of the control mesh is not sufficient to model the transition of the stern bulb into the stern tube properly. Based on irregular control points the number of control points which are available in the region of the stern tube is raised from 12 points to 20 points. Two irregular control points of valence $n = 3$ are introduced in order to obtain a symmetrical refinement of the mesh. In addition the valence of the two crease control points is raised by one compared to the initial situation. Triangular faces are avoided and the control mesh remains purely quadrilateral.

Figure 6.8 shows a detailed view of the stern bulb. The reflection lines illustrate the impact of the irregular control points on the surface quality. The two interior control points of valence $n = 3$ do not have any visible effect. The boundary control point of valence $n = 4$ causes a local distortion of the reflection lines. The crease control point of valence $n = 5$ introduces a notable cusp to the reflection lines which is difficult to smooth out in practice.

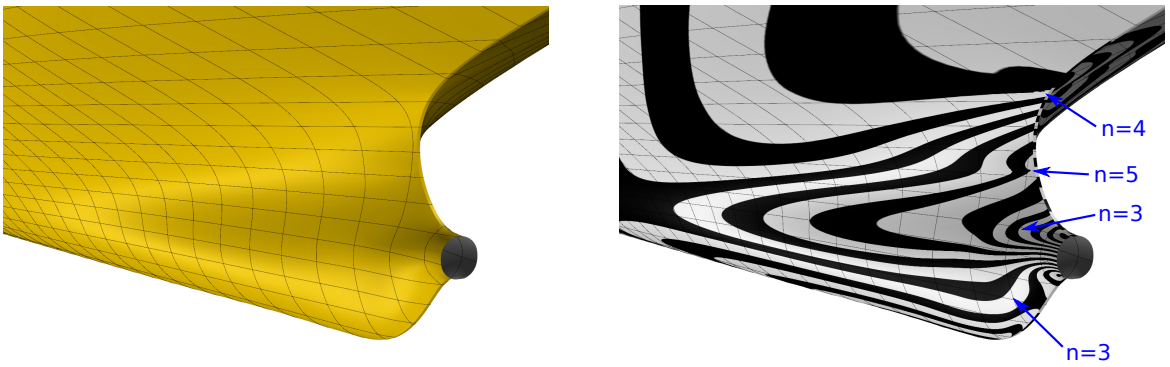


Figure 6.8.: Representation of a stern bulb. Left: Detailed view of the stern bulb defined by the control mesh shown on the right side of Figure 6.6. Note the detailed transition of the stern bulb into the stern tube and the flat bar on the center plane. Right: Impact of the irregular control points on the quality of the hull surface. The two interior control points of valence $n = 3$ do not have any visible effect on the surface quality. The crease control point of valence $n = 4$ causes a local distortion of the reflection lines. The crease control point of valence $n = 5$ introduces a notable cusp to the reflection lines which is difficult to smooth out.

6.5. Fairing

Fairing of B-spline surfaces based on interpolation is a time-consuming task. Interpolation tends to oscillation, but to eliminate oscillations is exactly what fairing is all about. In contrast, fairing is a simple task based on the control mesh. When the control mesh looks good, the surface looks good as well.

This easy rule of thumb is demonstrated in Figure 6.9. A side view and a top view of a control mesh for a hull form are shown. The control points are arranged in sections in order to support the naval architect's view on the surface. Therefore the longitudinal position of the control points is fixed, but not their lateral and vertical position. Working on these directions affects only the longitudinal edges in the given views. The goal is to remove as much oscillations as possible from the longitudinal edges while the section shape is maintained.

6.6. Design process

The starting point of the design process is the vessel type. It identifies the general structure of the control mesh as explained in Section 6.1. Important factors are the bow and stern type, the intended form of the flat of side and flat of bottom, and the length of the parallel middle body. Assuming an initial number of sections and control points for the control mesh, the hull designer is able to define the initial topology. Using ten sections in longitudinal direction and five control points between the flat of side and flat of bottom is usually sufficient to start modeling. The control mesh may, however, be refined locally or globally during the

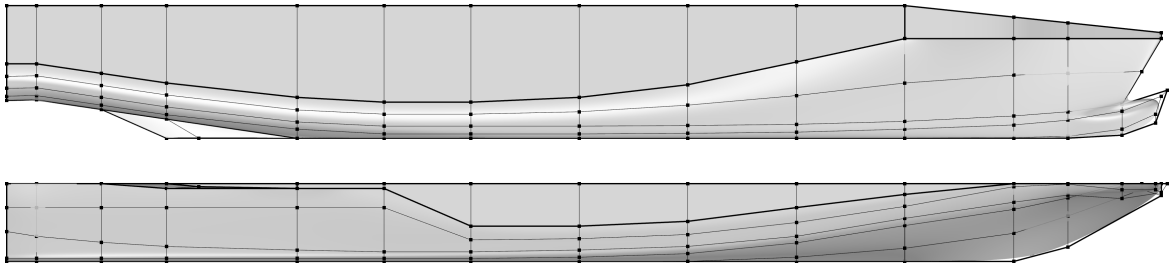


Figure 6.9.: Fairing of a hull form. A side view and a top view of a control mesh for a hull form of a fast vessel are shown. The longitudinal position of the control points is fixed, but not the lateral and vertical position. Working on these directions affects only the longitudinal edges in the given views. The goal of fairing is therefore to remove as much oscillations as possible from the longitudinal edges.

subsequent modeling steps.

The next modeling step is to define the feature curves of the hull form based on the main dimensions of the vessel such as the length L, the breadth B, the depth D. Using the example of the RoRo vessel shown in Figure C.1 and the corresponding control mesh shown in Figure C.3, this partially defines the position of the control points along the bold edges. While some of these control points are completely defined by the main dimensions such as the majority of the points along the deck, others are only partially defined such as the control points along the flat of side. The lateral position of these points is fixed to the breadth of the vessel, but the vertical position does not depend on the main dimensions, refer to Figure 6.4 for clarification. The hull designer chooses a preliminary position which might be exposed to changes in the subsequent modeling steps. The same holds for the non-feature control points which are placed according to an initial guess of the final hull shape and the intended section type, see Figure 6.3. The result of this modeling step is therefore a preliminary hull form that reflects the vessel's main dimensions.

The third modeling step addresses the volumetric parameters of the vessel such as the displacement ∇, the block coefficient c_B, the main section area coefficient c_M, and the longitudinal center of buoyancy LCB. The non-feature control points are mainly used to adjust these parameters. When the block coefficient is too high, the control points are placed closer to the center plane, when the block coefficient is too low, the control points are placed farther away. Similarly other volumetric properties such as the longitudinal center of buoyancy are adjusted. A continuously updated sectional area curve of the hull form is a useful tool in this context. This plot allows the hull designer to decide where the sections of the hull form have to be narrowed or widened. When the control points are additionally organized in sections as proposed in Section 6.1, the hull designer has a precise indication of the control points which are affected.

The fourth modeling step is the fairing of the hull surface. The fairing affects primarily

the non-feature control points, but might require to adjust the feature control points as well. An example are the control points of the flat of side or flat of bottom which may now be adjusted by the hull designer in order to support the fairing of the hull surface. Naturally, any change of control points affects the volumetric properties of the hull form, but typical design scenarios leave a margin to adjust most properties.

As a final remark, the described design process is similar to the design of hull forms based on curve networks. However, based on the authors' experience, the time needed to model a hull form based on a control mesh is less than the time needed to model a hull form based on a curve network. Most time is saved in the fourth step, the fairing, as this task is considerably simplified.

6.7. Example vessels

For a range of different vessel types, examples are presented in Appendix A to F. All example vessels are represented by only a single generalized B-spline surface as specified in Chapter 5 and the corresponding control mesh is shown for all vessels. In addition to a rendering of the hull surface, a reflection analysis is provided to illustrate the fairness of the hull forms as well as a lines plan and a sections plan which are both derived from the generalized B-spline surface. All renderings include a grid of sections and waterlines to improve the visualization of the hull forms. It is emphasized that these lines do not represent patch boundaries.

Container vessel

A hull form for a ultra-large container vessel is shown in Appendix A. The vessel has a bulbous bow, a stern bulb, a flat bottom and a flat side. The flat of side has the form of a knuckle close to the deck, but fades out quickly. The flat transom is included into the hull form and a flat bar runs along the centerline. The control mesh consists of 136 control points which are arranged in about 15 sections.

Bulk carrier

A hull form for a bulk carrier is shown in Appendix B. The vessel has a complex stern bulb, a large parallel middle body and a simple forebody with blunt waterlines and a displacement bulb. A flat bar and flat transom are included in the hull form. The control mesh consists of 125 control points which are arranged in about 10 sections. The parallel middle body is bounded by sections and requires no additional control points. Most control points are used in the afterbody to realize the complex stern bulb.

RoRo vessel

A hull form for a RoRo vessel is shown in Appendix C. This vessel is designed for high ship speeds and has therefore a low block coefficient and the longitudinal center of buoyancy is intentionally placed aft. The waterlines are straight and the forward shoulder is well rounded. The vessel features a bulbous bow, a skeg and a twin-propeller setup is realized based on a pram-type stern. A knuckle separates the forecastle and the flat of side also takes the form of a knuckle which only slowly fades out towards the design waterline. The control mesh consists of 117 control points which are arranged in about 15 sections.

Barge

A hull form of a simple barge as it is typically used for inland navigation is shown in Appendix D. The barge is designed for minimal production costs, the overall curvature is minimized with a hull form containing mostly flat regions. The control mesh consists of only 32 control points which are arranged in about five sections.

Research vessel

A hull form for a research vessel is shown in Appendix E. The vessel is designed for ice navigation what is reflected by the complex stem contour. The stern is tunneled and while the forebody has flare only, the remaining hull has also tumblehome. The forecastle is separated with a knuckle from the rest of the hull to minimize the overall curvature. The relatively large flat side is separated into three parts: the upper part has tumblehome, the middle part is vertical, and a small lower part has flare. Further the vessel has a flat bottom. The skeg and the transom are not included into the hull form. This vessel is the most complex vessel among the presented examples. The control mesh consists of 119 control points which are only loosely tied to sections. The control mesh is locally refined with the help of irregular control points to realize the complex stem contour.

Naval vessel

A hull form for a modern surface combatant is shown in Appendix F. With a block coefficient of only $C_B \approx 0.55$ the vessel is designed for high speeds. The vessel has a simple stern, but a complex forebody which includes a sonar dome at the bow. Above the waterline the vessel has a relatively large tumblehome to minimize the radar signature of the vessel. The control mesh consists of 153 control points which are arranged in about 15 sections. Four sections are concentrated around the forward shoulder to precisely control the radius of the waterlines. Compared to the bulbous bows of the other examples, a large number of control points is used to design the sonar dome and to realize smooth integration into the overall hull form.

6.8. Bibliographical notes

- This chapter is a minor revision of the material which was previously published by the author in:

  Sebastian H. Greshake and Robert Bronsart. Application of subdivision surfaces in ship hull form modeling. *Computer-Aided Design*, 100:79 – 92, 2018. ISSN 0010-4485. doi: https://doi.org/10.1016/j.cad.2018.03.004

- Section 6.7 is added to the previously published material and extends the analysis to a broad range of vessel types which includes standard vessels as well as one-off special purpose vessels.

This page intentionally left blank.

7 Integration into early ship design

Ship hull form modeling based on generalized B-splines has the potential to substantially improve hull design, but this is only of limited use when the support for this surface representation in ship design is missing. Providing the necessary support for generalized B-splines breaks down into two separate problems: on the one hand, how the support in hull design can be provided and one the other hand, how the results are effectively used in downstream applications. This chapter discusses the integration of generalized B-splines into the ship design process. First a general strategy is introduced and second an algorithm is described to convert a generalized B-spline surface into tensor-product B-splines.

7.1. Integration strategy

Making this surface representation available in hull design generally includes two options: either the support of generalized B-splines is added to existing design tools or completely new tools are developed based on this surface representation. Regarding the first alternative, to add the support of generalized B-splines to existing systems, this is generally possible as the standard architecture of geometric modeling kernels is compatible to generalized B-splines as shown by Antonelli et al. [3]. A geometric modeling kernel implements the means to represent and manipulate geometric objects. Such a kernel is the basis of a CAD software and nowadays only a few kernels exist which are conceptually quite similar to each other. A generalized B-spline surface is compatible to the commonly used boundary representation and can therefore be implemented on top of the standard kernels being used today. The advantage is that once implemented, all modeling tools provided by the kernel can be applied to generalized B-splines instantaneously. This is not only theoretically discussed by Antonelli et al. [3], but also implemented for a commercially available CAD system. The compatibility of generalized B-splines to standard kernels makes it relatively easy to develop new hull design tools based on this surface representation. Moreover, it is always a possibility to develop a dedicated software for hull form design on the basis of a custom implementation of generalized B-splines.

Regarding the integration in downstream design processes, the same arguments as above apply. It is, however, unlikely that a majority of the design systems used in downstream processes adopt generalized B-splines in the foreseeable future and in contrast to hull design, developing design systems from scratch for downstream processes is not an option. Therefore the only reasonable alternative is to convert a generalized B-spline to a surface representa-

tion which is widely supported in ship design. As pointed out in Chapter 2, the standard surface representation in this domain are tensor-product B-splines. The proposed strategy to integrate generalized B-splines into the ship design process is therefore to develop dedicated hull design tools based on this surface representation from scratch, or where possible to add the support for generalized B-splines to existing tools. Once the hull design is finished, this representation is converted back to tensor-product B-splines which is the industry standard for the exchange of surface data and commonly understood by other ship design systems.

7.2. Fitting algorithm

Two major algorithms are currently known to transform a generalized B-spline surface into tensor-product B-splines. The algorithm of Peters [85] generates a collection of patches which generally join with curvature continuity except for the vicinity of irregular control points where the patches join with normal continuity. The high-order continuity even at irregular control points comes at the price of a high number of patches being generated and the use of double interior knots which are known to be error-prone in some down-stream design tools. In contrast, the method of Loop and Schaefer [64] generates simple bicubic Bézier patches which are perfectly supported by all design tools. The patches generally join with curvature continuity, but in the vicinity of irregular control points the result is limited to position continuity only.

Both methods originate from the field of real-time rendering where the algorithm's time demand matters and the control points of the tensor-product B-spline patches are computed directly from the control points of the generalized B-spline surface in terms of simple conversion rules. While this is naturally a fast method, it is difficult to find conversion rules which result in smooth patches at irregularities. The algorithm's time demand is, however, not a primary requirement in hull design as this step is only required once each time a revision of the hull form is finalized. Therefore a different method for the conversion algorithm becomes a reasonable choice, namely fitting.

Tensor-product B-spline specification

A tensor-product B-spline is given by

$$\mathbf{x}_k(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} N_{i,p}(u)N_{j,q}(v)\mathbf{q}_{ij} \tag{7.1}$$

where $N_{i,p}$ and $N_{j,q}$ are the basis functions of degree p respectively degree q and $\mathbf{q}_{ij}$ are the control points.

For the scope of this study, the degree of the B-spline surface is chosen to be same in both directions, i.e. $p = q$. The same holds for the number of control points ($n = m$). For the knot

vectors

$$U = V = [\overbrace{0, \ldots, 0}^{p+1}, \underbrace{\frac{1}{n-p}, \frac{2}{n-p}, \ldots,}_{n-p-1} \overbrace{1, \ldots, 1}^{p+1}] \tag{7.2}$$

a standard open-uniform setup is chosen. Note that assuming the degree, the knot vectors, and the number of control points to be same for both parameter directions complies with the usual setup of generalized B-splines. Moreover this particular choice of the knot vectors let the domain of the tensor-product B-spline coincide with the unit cell, i.e. $(u, v) \in [0, 1]^2$.

Least-squares fitting

One tensor-product B-spline $\tilde{\mathbf{x}}_k$ is fitted to each patch of the generalized B-spline surface $\mathbf{x}$. A standard least-squares fitting, referred to as $\mathcal{LS}$, is utilized to compute the control points $\mathbf{q}_{ij}$ of the data points $\mathbf{x}_{ij}$ being generated on a uniform grid as shown in Figure 7.1a. More precisely,

$$\mathbf{x}_{ij} = \mathbf{x}(\frac{i}{m}, \frac{j}{m}, k) \tag{7.3}$$

with $i \in [0, m]$ and $j \in [0, m]$ are the data points defined for the kth patch of the generalized B-spline. The same grid is chosen as the parameterization of the least-squares problem. In order to guarantee watertightness of the generated tensor-product B-spline surfaces, a three-step fitting procedure is realized. The notation is clarified in Figure 7.1.

- **Step 1:** The patch corners are interpolated by the tensor-product B-spline:

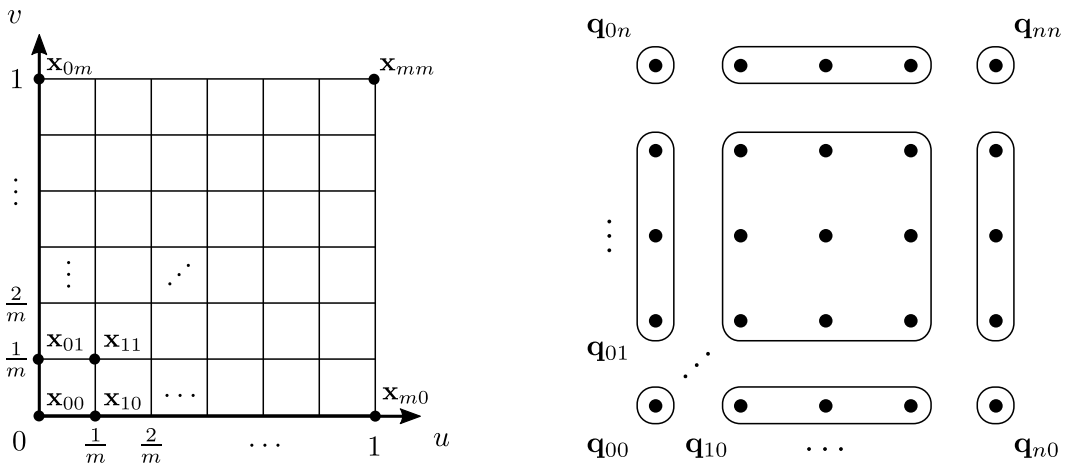
$$\begin{aligned} \tilde{\mathbf{x}}_k(0,0) &= \mathbf{x}_{00} \\ \tilde{\mathbf{x}}_k(0,1) &= \mathbf{x}_{0m} \\ \tilde{\mathbf{x}}_k(1,0) &= \mathbf{x}_{m0} \\ \tilde{\mathbf{x}}_k(1,1) &= \mathbf{x}_{mm} \end{aligned} \tag{7.4}$$

  this results in the corner points $\mathbf{q}_{00}$, $\mathbf{q}_{0n}$, $\mathbf{q}_{n0}$, and $\mathbf{q}_{nn}$.

- **Step 2:** The boundary curves of the tensor-product B-spline are fitted to the outermost data points:

$$\begin{aligned} \tilde{\mathbf{x}}_k(u,0) &= \mathcal{LS}(\mathbf{x}_{00}, \mathbf{x}_{10}, \ldots, \mathbf{x}_{m0}) \\ \tilde{\mathbf{x}}_k(u,1) &= \mathcal{LS}(\mathbf{x}_{0m}, \mathbf{x}_{1m}, \ldots, \mathbf{x}_{mm}) \\ \tilde{\mathbf{x}}_k(0,v) &= \mathcal{LS}(\mathbf{x}_{00}, \mathbf{x}_{01}, \ldots, \mathbf{x}_{0m}) \\ \tilde{\mathbf{x}}_k(1,v) &= \mathcal{LS}(\mathbf{x}_{m0}, \mathbf{x}_{m1}, \ldots, \mathbf{x}_{mm}) \end{aligned} \tag{7.5}$$

  whereby the already known corner control points $\mathbf{q}_{00}$, $\mathbf{q}_{0n}$, $\mathbf{q}_{n0}$, and $\mathbf{q}_{nn}$ are taken into



(a) (a) Data points

(b) (b) Control points

Figure 7.1.: Notation of the least-squares fitting procedure used to generate a tensor-product B-spline surface $\tilde{\mathbf{x}}_k$ for each patch of a generalized B-spline surface $\mathbf{x}$. Left: The uniform parameter grid that is utilized to generate the data points $\mathbf{x}_{ij}$ to which the generated tensor-product B-spline surface is fitted to. Right: The control points $\mathbf{q}_{ij}$ of the generated tensor-product B-spline surface. The boxes indicate the three steps of the fitting procedure. First the corners are calculated, next the control points inside the elongated boundary boxes are computed, and finally the interior control points are computed.

account. The result are the outermost control points of the tensor-product B-spline patch as illustrated in Figure 7.1b.

- **Step 3:** The surface is fitted to the entire data set what results in the interior control points

$$\tilde{\mathbf{x}}_k(u,v) = \mathcal{LS}(\mathbf{x}_{00}, \mathbf{x}_{01}, \ldots, \mathbf{x}_{mm}) \tag{7.6}$$

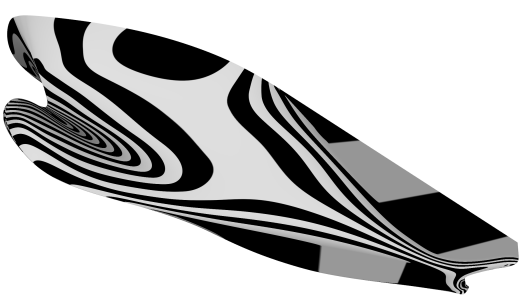
whereby all previously calculated control points are taken into account.

7.3. Results

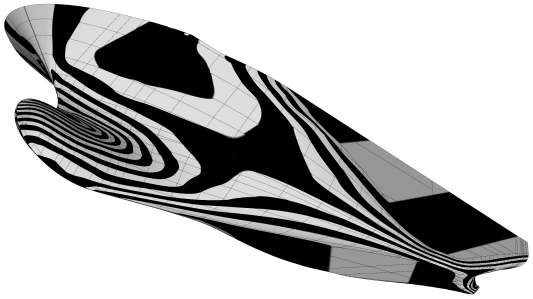
The algorithm is analyzed for the hull form of a modern container vessel shown in Appendix A. This hull form closely imitates the design of the Duisburg Test Case (DTC) [29] and therefore the original surface representation of this hull form is used as reference in the following analysis.

The conversion algorithm generates about 400 patches which are shown in Figure 7.2 for the degree $p = 3$ and the number of control points in ranging from 4×4 to 7×7.

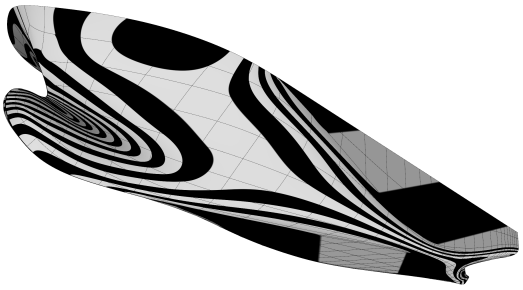
The patches are indicated by the patch boundaries which are rendered as dark gray curves.



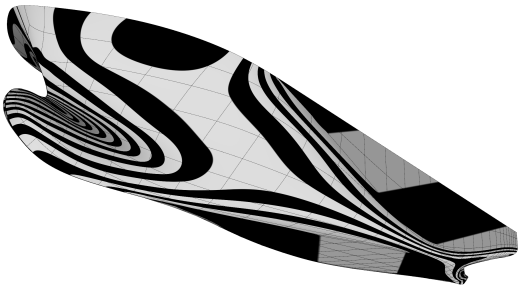
(a) Generalized B-spline surface



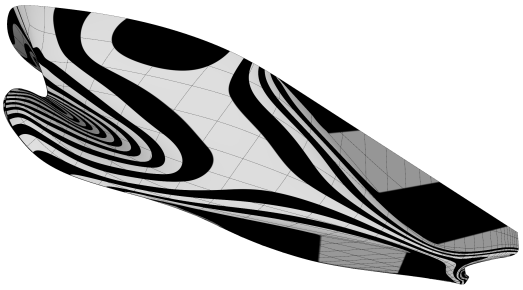
(b) Duisburg Test Case (DTC)



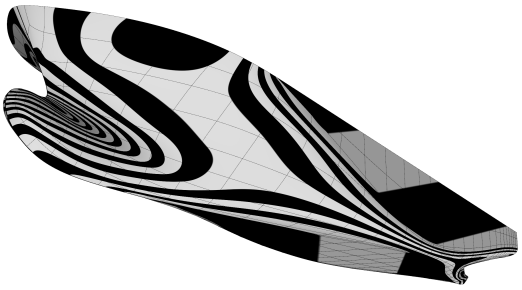
(c) Fitting with $p = 3$ and $n = 4$



(d) Fitting with $p = 3$ and $n = 5$



(e) Fitting with $p = 3$ and $n = 6$



(f) Fitting with $p = 3$ and $n = 7$

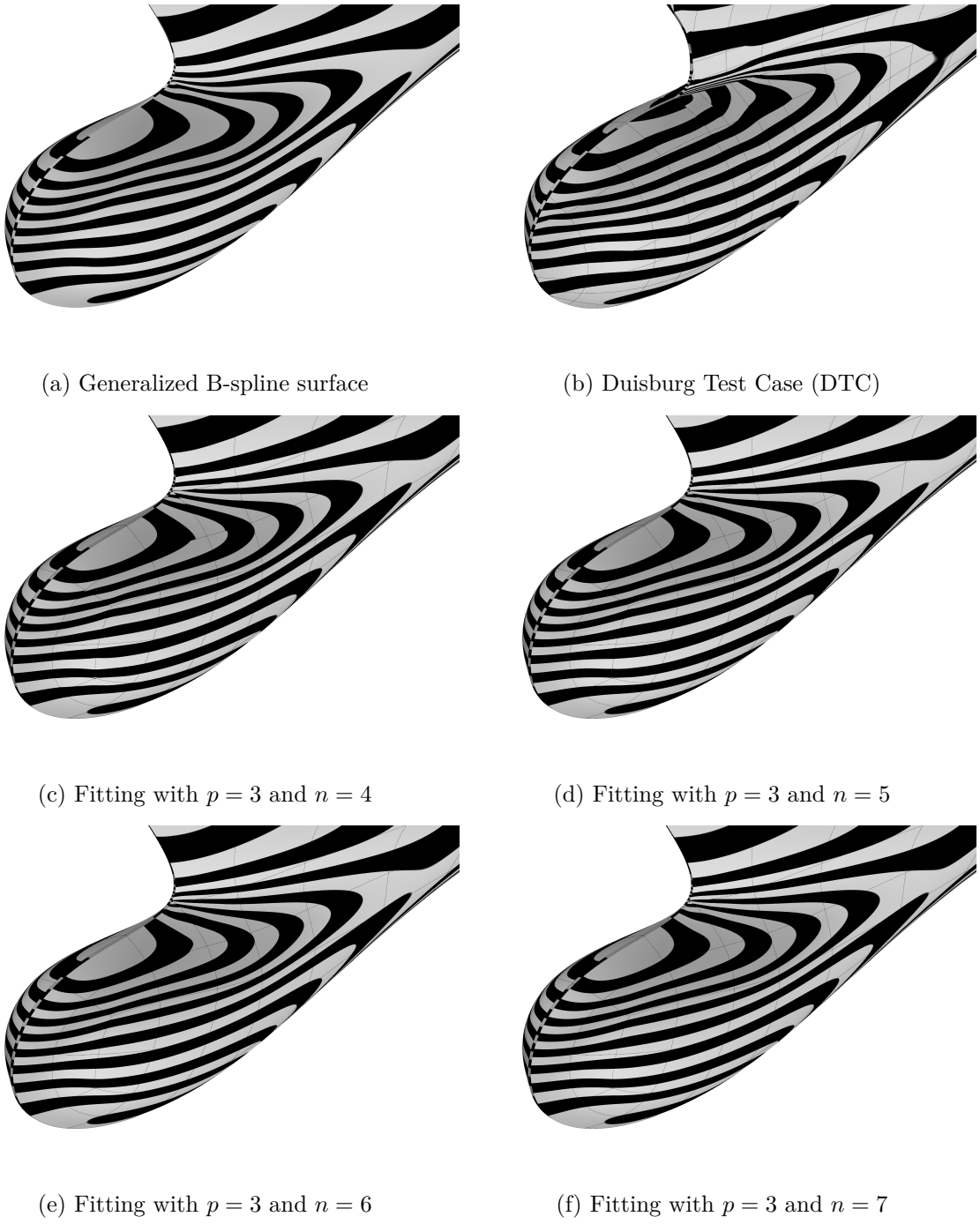
Figure 7.2.: The conversion algorithm generates about 400 patches which are shown for the degree $p = 3$ and the number of control points in each direction in ranging from $n = 4$ to $n = 7$. The reflection analysis shows that the good quality of the generalized B-spline surface is maintained. The generated patches appear curvature continuously at this scale, whereas the surface representation of the DTC as a reference for the curve-based design looks only normal continuous.

The DTC is shown for reference and is represented by almost 700 patches. Both are a rather large number of patches in this context and from the author's experience a well-designed curve network results in only about 200 patches for this particular case. This figure is only a very rough indication as the actual number of patches depends considerably on the designer's skills and the topology of the curve network. It is emphasized, however, that a few hundreds of patches usually represent no problem in practice. Only when the number of patches exceeds a few thousands of patches, a software which is processing the surfaces is potentially slowed down.

The reflection analysis shows that the high quality of the generalized B-spline surface is maintained by the patches. At the scale shown in Figure 7.2 which covers the entire hull form, it is hard to identify any differences between the generalized B-spline and the fitted patches, regardless of the number of control points chosen for the patches. In comparison, the DTC is clearly of lower quality. While the generalized B-spline surface and the generated patches are both almost curvature continuous as indicated by the smooth reflection lines, the DTC is only normal continuous. Therefore the reflection lines are less smooth and change their directions at the patch boundaries.

In Figure 7.3 a detailed analysis of the bulbous bow is shown. Again, the patch boundaries are shown by the dark gray curves. At this scale, differences are noticeable for the different parameters the patches are fitted with. For $n = 4$ the reflection lines do not exactly match up across patch boundaries incident of irregularities, i.e. locations where more or less than three patches intersect in a common point. This indicates a normal error and therefore the patches are visibly only position continuous. The normal distortion is, however, small as the mismatch of the reflection lines is rather small. For $n = 5$ the reflection lines are significantly better. There is only a minor improvement of the reflection lines for $n > 5$ so that n = 5 is identified as a reasonable trade-off between the number of control points and the surface quality continuity. The result is very close to the quality of the generalized B-spline surface.

In both cases, the generated patches clearly outperform the surface representation of the DTC. From the author's experience, it is a rather good representative of a patch-based surface representation of a hull form which is generated based on the curve-based design methodology. It comes as no surprise that the generalized B-spline surface outperforms the curve-based design as it is basically curvature continuous and the curve-based design methodology is limited to normal continuity as pointed out in Chapter 2. That a least-squares fitting suffices, however, to widely maintain the continuity characteristics of a generalized B-spline surface surprises. This is a step forward in hull form modeling, as now there is not only a method to quickly create high quality hull surfaces, but also a method to convert them to the industry standard for the exchange of surface data which maintains the good overall quality upon conversion.



(a) Generalized B-spline surface

(b) Duisburg Test Case (DTC)

(c) Fitting with $p = 3$ and $n = 4$

(d) Fitting with $p = 3$ and $n = 5$

(e) Fitting with $p = 3$ and $n = 6$

(f) Fitting with $p = 3$ and $n = 7$

Figure 7.3.: Reflection analysis of the generated surfaces. Whenever reflection lines do not match across patch boundaries, the normals are not the same and patches are only position continuous. For $n = 4$ the reflection lines have a small mismatch along some of the patch boundaries, but for $n = 5$ the reflection lines appear smooth. For $n > 5$ only a minor improvement is noticeable. For all n, the result clearly outperforms the surface representation of the DTC which is generated based on the curve-based design methodology.

7.4. Bibliographical notes

- The conversion of a generalized B-spline surface into cubic Bézier patches based on the algorithm of Loop and Schaefer [64] was previously analyzed by the author in

  Sebastian H. Greshake and Robert Bronsart. Integration of ship hull form modeling based on subdivision surfaces with other ship design tools. In *International Conference on Computer Applications in Shipbuilding (ICCAS 2017)*, volume 3, pages 61–68, 2017

  It was found, however, that unwanted knuckles occur in the presence of irregularities. The fitting method introduced in this thesis performs better and avoid unwanted knuckles. Thus the initial conversion algorithm presented by the author is not included into the material, but the reader may refer to publication for more information.

8 Conclusion

Ship hull form modeling is currently based on the curve-based design method. The designer specifies a network of curves which is continuously *interpolated* by tensor-product B-spline patches in order to obtain a surface representation for the hull form. In practice, this method is limited to tangent continuity which is not considered smooth in the context of hull form design. Moreover, the fairing of the hull form is a time-consuming process since small changes of the curve network may result in large differences of the generated surface and interpolation is generally well-known for unwanted oscillations. In contrast, tensor-product B-splines are designed as a smooth *approximation* of the control mesh they are defined on. The simple link between the control mesh and the surface makes it an easy to use, but yet powerful and precise tool for modeling. A small change of the control mesh implies only a small change of the surface and unwanted oscillations are minimized in terms of the control mesh. Modeling based on the approximation of the control mesh instead of the interpolation of curve networks is therefore considered to be the better alternative.

Tensor-product B-splines are limited to four-sided surfaces only. Therefore hull forms and other complex surfaces are decomposed into patches what increases the number of control points and introduces complex continuity constraints to many control points in order to maintain smooth transitions between neighboring patches. Naturally, it is difficult to use a control mesh for modeling when the number of control points is high and many of them are constrained. Thus, the control mesh is usually not used for hull modeling. Instead the curve-based design, though being affected by important disadvantages such as unwanted oscillations and unpredictable behavior when curves are modified, is preferred in practice. However, to identify the limitation of tensor-product B-splines to four-sided surfaces as the reason for the dominance of curve-based design represents a controversial point of view in the community and is an essential result of this thesis.

Having this in mind, generalized B-splines are introduced. They behave like tensor-product B-splines, but are capable to represent surfaces of any complexity on a single control mesh with a significant smaller number of control points. This makes the control mesh a suitable option for hull form modeling. While many authors believe that the curve network reflects the naval architect's view on a hull surface, and is therefore the natural choice for hull form design, it is shown that the structure and the principles to define control meshes and curve networks are basically the same in both cases. On the basis of generalized B-splines it is therefore concluded that the control mesh integrates equally well into ship hull form modeling.

In contrast to the curve-based design, better results are obtained with generalized B-splines

as the surface is almost everywhere G^{p-1} continuous with p being the degree and not being limited to normal continuity (G^1) as usually the case for the curve network interpolation. Since curvature continuity (G^2) is required a degree $p = 3$ is recommended for this application. A fair hull form, however, is not only smooth, but also has a minimal number of inflection points. While interpolation is well-known for unwanted oscillations, they are easily kept to minimum on the basis of a control mesh. The simplified fairing is the main reason why the modeling performance is increased on the basis of the control mesh.

Smooth surfaces and an improved modeling performance are, however, only of limited use when the support for generalized B-splines in ship design is missing. It is unlikely that this surface representation will be supported by many ship design systems in the foreseeable future and therefore the result is converted to tensor-product B-splines, once another revision of the hull form is released. The conversion methods proposed in the literature are based on an explicit conversion which is either limited to position continuity in the vicinity of irregularities or complex non-uniform patches are required. In this thesis, a simple least squares fitting of tensor-product B-splines is utilized. The result maintains the high quality of the original surface and clearly outperforms the curve-based methods.

The generalized B-spline surface which is used in this thesis offers a reasonable trade-off between a low-degree, curvature continuity, and additional modeling features. It is successfully utilized for the representation of knuckles and, with limitations, tangent curves on the surface. The limitations of the surface are the zero curvature across features and the inability to represent conic sections exactly. It is proposed that future work focuses on the representation of tangent curves and the representation of circular arcs as the most important application of conic sections in hull form modeling. Another potential improvement is the behavior of the surface in the vicinity of irregularities. The surface is not only limited to normal continuity at those spots, but the surface is also visibly distorted when the control points are not carefully chosen. Future work may also address this issue.

Bibliography

[1] A. Abbas. Interpolating arbitrary networks of curves by Catmull-Clark subdivision surfaces. *Computer-Aided Design and Applications*, 3(1-4):505–512, 2006.

[2] Abdulwahed Abbas and Ahmad Nasri. Interpolating multiple intersecting curves using Catmull-Clark subdivision surfaces. *Computer-Aided Design and Applications*, 1(1-4): 25–32, 2004.

[3] Michele Antonelli, Carolina Vittoria Beccari, Giulio Casciola, Roberto Ciarloni, and Serena Morigi. Subdivision surfaces integrated in a CAD system. *Computer-Aided Design*, 45(11):1294–1305, 2013. ISSN 0010-4485. doi: 10.1016/j.cad.2013.06.007.

[4] B. A. Barsky and T. D. DeRose. Geometric continuity of parametric curves: three equivalent characterizations. *IEEE Computer Graphics and Applications*, 9(6):60–69, November 1989. ISSN 0272-1716. doi: 10.1109/38.41470.

[5] Henning Biermann, Adi Levin, and Denis Zorin. Piecewise Smooth Subdivision Surfaces with Normal Control. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 113–120, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co. ISBN 1-58113-208-5. doi: 10.1145/344779.344841.

[6] Henning Biermann, Ioana M. Martin, Denis Zorin, and Fausto Bernardini. Sharp Features on Multiresolution Subdivision Surfaces. *Graphical Models*, 64(2):61 – 77, 2002. ISSN 1524-0703. doi: 10.1006/gmod.2002.0570.

[7] Lothar Birk and T. Luke McCulloch. Robust generation of constrained B-spline curves based on automatic differentiation and fairness optimization. *Computer Aided Geometric Design*, 59:49 – 67, 2018. ISSN 0167-8396. doi: https://doi.org/10.1016/j.cagd.2017.11.005.

[8] Ioana Boier-Martin and Denis Zorin. Differentiable Parameterization of Catmull-Clark Subdivision Surfaces. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '04, pages 155–164, New York, NY, USA, 2004. ACM. ISBN 3-905673-13-4. doi: 10.1145/1057432.1057453.

[9] Marcus Bole. *A Hull Surface Generation Technique Based on a Form Topology and Geometric Constraint Approach*. PhD thesis, University of Strathclyde, 2002.

[10] Marcus Bole. Interactive Hull Form Transformations using Curve Network Deformation. *Ship Technology Research*, 58(1):46–64, 2011. doi: 10.1179/str.2011.58.1.004.

[11] Marcus Bole. Revisiting Traditional Curve Lofting to Improve the Hull Surface Design Process. *Ship Technology Research*, 59(2):18–33, 2012. doi: 10.1179/str.2012.59.2.002.

[12] Marcus Bole. Regenerating Hull Design Definition from Poor Surface Definitions and other Geometric Representations. In *13th International Conference on Computer and IT Applic ations in the Maritime Industries*, 2014.

[13] Marcus Bole. A Strategy for Closely Integrating Parametric Generation and Interactive Manipulation in Hull Surface Design. In *18th International Conference on Computer and IT Applic ations in the Maritime Industries*, 2019.

[14] Marcus Bole and BS Lee. An hierarchical approach to hull form design. In *NAV 2003 : International Conference on Ship and Shipping Research*, 2003.

[15] Marcus Bole, Graphics Research Corporation Ltd, Byung-Suk Lee, and University of Strathclyde-Glasgow. Integrating Parametric Hull Generation into Early Stage Design. *Ship Technology Research*, 53(3):115–137, 2006. doi: 10.1179/str.2006.53.3.003.

[16] Jeffrey Bolz and Peter Schröder. Rapid evaluation of Catmull-Clark subdivision surfaces. In *Proceedings of the seventh international conference on 3D Web technology*, pages 11–17. ACM, 2002.

[17] Robert Bronsart, Desta M. Edessa, and Lutz Kleinsorge. Automatic Pre-Mesh CAD Data Repairing. *International Journal of Mechanical Engineering and Applications*, 1 (1):1–9, 2013.

[18] Robert Bronsart, Desta M. Edessa, and Lutz Kleinsorge. A Contribution to Automatic Ship Hull Form CAD Data Repairing. In *12th International Symposium on Practical Design of Ships and Other Floating Structures*, pages 437–443, 2013.

[19] Robert Bronsart, Desta M. Edessa, and Lutz Kleinsorge. Assuring Quality Ship Hull Form Representations for Downstream Applications. In *Proceedings of 2nd International Conference on Maritime Technology and Engineering*, pages 317–324, 2014.

[20] Thomas J. Cashman, Ursula H. Augsdörfer, Neil A. Dodgson, and Malcolm A. Sabin. NURBS with Extraordinary Points: High-degree, Non-uniform, Rational Subdivision Schemes. *ACM Trans. Graph.*, 28(3):46:1–46:9, July 2009. ISSN 0730-0301. doi: 10.1145/1531326.1531352.

[21] Thomas J. Cashman, Neil A. Dodgson, and Malcolm A. Sabin. Selective knot insertion for symmetric, non-uniform refine and smooth B-spline subdivision. *Computer Aided*

Geometric Design, 26(4):472 – 479, 2009. ISSN 0167-8396. doi: http://dx.doi.org/10.1016/j.cagd.2008.11.002.

[22] Thomas J. Cashman, Neil A. Dodgson, and Malcolm 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. ISSN 0167-8396. doi: 10.1016/j.cagd.2007.12.001.

[23] Edwin Catmull and James Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10(6):350–355, 1978. ISSN 0010-4485. doi: 10.1016/0010-4485(78)90110-0.

[24] Doo-Yeoun Cho, Kyu-Yeul Lee, and Tae-Wan Kim. Interpolating {G1} Bézier surfaces over irregular curve networks for ship hull design. *Computer-Aided Design*, 38(6):641 – 660, 2006. ISSN 0010-4485. doi: http://dx.doi.org/10.1016/j.cad.2006.02.005.

[25] Steven A Coons. Surfaces for computer-aided design of space forms. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE PROJECT MAC, 1967.

[26] Tony DeRose, Michael Kass, and Tien Truong. Subdivision Surfaces in Character Animation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 85–94, New York, NY, USA, 1998. ACM. ISBN 0-89791-999-8. doi: 10.1145/280814.280826.

[27] Daniel Doo and Malcolm Sabin. Behaviour of recursive division surfaces near extraordinary points. *Computer-Aided Design*, 10(6):356–360, 1978. ISSN 0010-4485. doi: 10.1016/0010-4485(78)90111-2.

[28] Donald Doo and Malcolm Sabin. A subdivision algorithm for smoothing down irregularly shaped polyhedrons. In *Proceedings on interactive techniques in computer aided design*, volume 157, page 165, 1978.

[29] Ould el Moctar, Vladimir Shigunov, and Tobias Zorn. Duisburg Test Case: Post-panamax container ship for benchmarking. *Ship Technology Research-Schiffstechnik*, 59 (3):50–64, 2012.

[30] Jianhua Fan and Jörg Peters. Smooth Bi-3 spline surfaces with fewest knots. *Computer-Aided Design*, 43(2):180 – 187, 2011. ISSN 0010-4485. doi: http://dx.doi.org/10.1016/j.cad.2010.11.002.

[31] Gerald E Farin. *Curves and surfaces for CAGD: a practical guide*. Morgan Kaufmann, 2002.

[32] Gerald E Farin, Josef Hoschek, and Myung-Soo Kim. *Handbook of computer aided geometric design*. Elsevier, 2002.

[33] Carlotta Giannelli, Bert Jüttler, and Hendrik Speleers. Thb-splines: The truncated basis for hierarchical splines. *Computer Aided Geometric Design*, 29(7):485 – 498, 2012. ISSN 0167-8396. doi: https://doi.org/10.1016/j.cagd.2012.03.025.

[34] William J Gordon. Spline-blended surface interpolation through curve networks. *Journal of Mathematics and Mechanics*, pages 931–952, 1969.

[35] Sebastian H. Greshake and Robert Bronsart. Using subdivision surfaces to address the limitations of B-spline surfaces in ship hull form modeling. In *Proceedings of the 13th International Symposium on Practical Design of Ships and Other Floating Structures (PRADS 2016)*, 2016.

[36] Sebastian H. Greshake and Robert Bronsart. Integration of ship hull form modeling based on subdivision surfaces with other ship design tools. In *International Conference on Computer Applications in Shipbuilding (ICCAS 2017)*, volume 3, pages 61–68, 2017.

[37] Sebastian H. Greshake and Robert Bronsart. Application of subdivision surfaces in ship hull form modeling. *Computer-Aided Design*, 100:79 – 92, 2018. ISSN 0010-4485. doi: https://doi.org/10.1016/j.cad.2018.03.004.

[38] Stefan Harries. *Parametric design and hydrodynamic optimization of ship hull forms*. PhD thesis, Mensch-und-Buch-Verlag, 1998.

[39] Hugues Hoppe, Tony DeRose, Tom Duchamp, Mark Halstead, Hubert Jin, John McDonald, Jean Schweitzer, and Werner Stuetzle. Piecewise Smooth Surface Reconstruction. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, pages 295–302, New York, NY, USA, 1994. ACM. ISBN 0-89791-667-0. doi: 10.1145/192161.192233.

[40] Zhangjin Huang and Guoping Wang. Non-uniform recursive Doo–Sabin surfaces. *Computer-Aided Design*, 43(11):1527 – 1533, 2011. ISSN 0010-4485. doi: http://dx.doi.org/10.1016/j.cad.2011.08.016.

[41] Kęstutis Karčiauskas and Jörg Peters. Guided spline surfaces. *Computer Aided Geometric Design*, 26(1):105 – 116, 2009. ISSN 0167-8396. doi: https://doi.org/10.1016/j.cagd.2007.12.002.

[42] Kęstutis Karčiauskas and Jörg Peters. Biquintic G2 surfaces. *The Mathematics of Surfaces*, 14:213–236, 2013.

[43] Kęstutis Karčiauskas and Jörg Peters. Biquintic G2 surfaces via functionals. *Computer Aided Geometric Design*, 33:17 – 29, 2015. ISSN 0167-8396. doi: https://doi.org/10.1016/j.cagd.2014.11.003.

[44] Kęstutis Karčiauskas and Jörg Peters. Can Bi-cubic Surfaces Be Class A? *Comput. Graph. Forum*, 34(5):229–238, 2015. ISSN 0167-7055. doi: 10.1111/cgf.12711.

[45] Kęstutis Karčiauskas and Jörg Peters. Minimal bi-6 G2 completion of bicubic spline surfaces. *Computer Aided Geometric Design*, 41:10 – 22, 2016. ISSN 0167-8396. doi: https://doi.org/10.1016/j.cagd.2015.10.005.

[46] Kęstutis Karčiauskas and Jörg Peters. Improved shape for refinable surfaces with singularly parameterized irregularities. *Computer-Aided Design*, 90:191 – 198, 2017. ISSN 0010-4485. doi: https://doi.org/10.1016/j.cad.2017.05.004.

[47] Hyun Cheol Kim. *Parametric design of ship hull forms with a complex multiple domain surface topology*. PhD thesis, Mensch-und-Buch-Verlag, 2004.

[48] Herbert J. Koelman. Application of the H-rep Ship Hull Modelling Concept. *Ship Technology Research*, 50(4):172–181, 2003. doi: 10.1179/str.2003.50.4.005.

[49] Herbert J. Koelman and Bastiaan N. Veelo. A technical note on the geometric representation of a ship hull form. *Computer-Aided Design*, 45(11):1378 – 1381, 2013. ISSN 0010-4485. doi: http://dx.doi.org/10.1016/j.cad.2013.06.013.

[50] Herbert Jan Koelman. *Computer support for design, engineering and prototyping of the shape of ship hulls*. PhD thesis, Delft University of Technology, 1999.

[51] H.J. Koelman, I. Horváth, and A. Aalbers. Hybrid representation of the shape of ship hulls. *International Shipbuilding Progress*, 48(3):247–269, 2001.

[52] J. Kosinka, M. A. Sabin, and N. A. Dodgson. Semi-sharp Creases on Subdivision Curves and Surfaces. *Computer Graphics Forum*, 33(5):217–226, 2014. ISSN 1467-8659. doi: 10.1111/cgf.12447.

[53] J. Kosinka, M. A. Sabin, and N. A. Dodgson. Subdivision Surfaces with Creases and Truncated Multiple Knot Lines. *Computer Graphics Forum*, 33(1):118–128, 2014. ISSN 1467-8659. doi: 10.1111/cgf.12258.

[54] Jiří Kosinka, Malcolm Sabin, and Neil Dodgson. Cubic subdivision schemes with double knots. *Computer Aided Geometric Design*, 30(1):45 – 57, 2013. ISSN 0167-8396. doi: http://dx.doi.org/10.1016/j.cagd.2012.06.004.

[55] Jiří Kosinka, Malcolm Sabin, and Neil Dodgson. Creases and boundary conditions for subdivision curves. *Graphical Models*, 76(5):240 – 251, 2014. ISSN 1524-0703.

[56] Dylan Lacewell and Brent Burley. Exact evaluation of Catmull-Clark subdivision surfaces near B-spline boundaries. *Journal of Graphics, GPU, and Game Tools*, 12(3): 7–15, 2007.

[57] Shuhua Lai and Fuhua Cheng. Parametrization of general Catmull-Clark subdivision surfaces and its applications. *Computer-Aided Design and Applications*, 3(1-4):513–522, 2006.

[58] Kyu-Yeul Lee, Doo-Yeoun Cho, and Tan-Wan Kim. Interpolation of the irregular curve network of ship hull form using subdivision surfaces. *Computer-Aided Design and Applications*, 1(1-4):17–23, 2004.

[59] Adi Levin. Combined subdivision schemes for the design of surfaces satisfying boundary conditions. *Computer Aided Geometric Design*, 16(5):345–354, 1999.

[60] Adi Levin. Interpolating nets of curves by smooth subdivision surfaces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 57–64. ACM Press/Addison-Wesley Publishing Co., 1999.

[61] Adi Levin. Modified Subdivision Surfaces with Continuous Curvature. *ACM Trans. Graph.*, 25(3):1035–1040, July 2006. ISSN 0730-0301. doi: 10.1145/1141911.1141990.

[62] Qijin Liu and T. C. Sun. G1 interpolation of mesh curves. *Computer-Aided Design*, 26 (4):259 – 267, 1994. ISSN 0010-4485.

[63] Charles Loop and T. D. DeRose. Generalized B-spline Surfaces of Arbitrary Topology. *SIGGRAPH Comput. Graph.*, 24(4):347–356, 1990. ISSN 0097-8930. doi: 10.1145/97880.97917.

[64] Charles Loop and Scott Schaefer. Approximating Catmull-Clark subdivision surfaces with bicubic patches. *ACM Transactions on Graphics (TOG)*, 27(1):8, 2008.

[65] Charles Loop and Scott Schaefer. G2 tensor product splines over extraordinary vertices. In *Computer Graphics Forum*, volume 27, pages 1373–1382. Wiley Online Library, 2008.

[66] Charles T. Loop and Tony D. DeRose. A Multisided Generalization of Bézier Surfaces. *ACM Trans. Graph.*, 8(3):204–234, 1989. ISSN 0730-0301. doi: 10.1145/77055.77059.

[67] T. Luke McCulloch. *Feasible Form Parameter Design of Complex Ship Hull Form Geometry*. PhD thesis, University of New Orleans, 2018.

[68] Kerstin Müller, Lars Reusche, and Dieter Fellner. Extended Subdivision Surfaces: Building a Bridge Between NURBS and Catmull-Clark Surfaces. *ACM Trans. Graph.*, 25(2):268–292, 2006. ISSN 0730-0301. doi: 10.1145/1138450.1138455.

[69] Kerstin Müller, Christoph Fünfzig, Lars Reusche, Dianne Hansford, Gerald Farin, and Hans Hagen. Dinus: Double Insertion, Nonuniform, Stationary Subdivision Surfaces. *ACM Trans. Graph.*, 29(3):25:1–25:21, 2010. ISSN 0730-0301. doi: 10.1145/1805964.1805969.

[70] A Nasri, A Abbas, and I Hasbini. Skinning Catmull-Clark subdivision surfaces with incompatible cross-sectional curves. In *Computer Graphics and Applications, 2003. Proceedings. 11th Pacific Conference on*, pages 102–111. IEEE, 2003.

[71] Ahmad H. Nasri. Recursive subdivision of polygonal complexes and its applications in computer-aided geometric design. *Computer Aided Geometric Design*, 17(7):595 – 619, 2000. ISSN 0167-8396. doi: http://dx.doi.org/10.1016/S0167-8396(00)00015-7.

[72] Ahmad H. Nasri and A. Abbas. Designing Catmull–Clark subdivision surfaces with curve interpolation constraints. *Computers & Graphics*, 26(3):393 – 400, 2002. ISSN 0097-8493. doi: http://dx.doi.org/10.1016/S0097-8493(02)00082-1.

[73] Ahmad H Nasri and A Abbas. Lofted catmull-clark subdivision surfaces. In *Geometric Modeling and Processing, 2002. Proceedings*, pages 83–93. IEEE, 2002.

[74] Ahmed H Nasri. Interpolating an Unlimited Number of Curves Meeting at Extraordinary Points on Subdivision Surfaces. In *Computer Graphics Forum*, volume 22, pages 87–97. Wiley Online Library, 2003.

[75] Horst Nowacki. Five decades of computer-aided ship design. *Computer-Aided Design*, 42(11):956–969, 2010.

[76] Horst Nowacki, Malcolm Bloor, and Boguslaw Oleksiewicz. *Computational geometry for ships*. World scientific, 1995.

[77] Min-jae Oh, Kittichai Suthunyatanakit, Sung Ha Park, and Tae-wan Kim. Constructing G1 Bézier surfaces over a boundary curve network with T-junctions. *Computer-Aided Design*, 44(7):671 – 686, 2012. ISSN 0010-4485. doi: https://doi.org/10.1016/j.cad.2012.02.008.

[78] Min-Jae Oh, Myung-Il Roh, and Tae-wan Kim. G1 Bézier surface interpolation with T-junctions at a 3-valent singular vertex. *Computer Aided Geometric Design*, 67:79 – 95, 2018. ISSN 0167-8396. doi: https://doi.org/10.1016/j.cagd.2018.10.001.

[79] Min-Jae Oh, Myung-Il Roh, and Tae-wan Kim. Local T-spline surface skinning with shape preservation. *Computer-Aided Design*, 2018. ISSN 0010-4485. doi: https://doi.org/10.1016/j.cad.2018.04.006.

[80] Jörg Peters. Smooth interpolation of a mesh of curves. *Constructive Approximation*, 7 (1):221–246, 1991. ISSN 1432-0940. doi: 10.1007/BF01888155.

[81] Jörg Peters. Smooth free-form surfaces over irregular meshes generalizing quadratic splines. *Computer Aided Geometric Design*, 10(3):347 – 361, 1993. ISSN 0167-8396. doi: http://dx.doi.org/10.1016/0167-8396(93)90046-6.

[82] Jörg Peters. Biquartic C1-surface splines over irregular meshes. *Computer-Aided Design*, 27(12):895 – 903, 1995. ISSN 0010-4485. doi: http://dx.doi.org/10.1016/0010-4485(95)00010-0.

[83] Jörg Peters. C1-Surface Splines. *SIAM Journal on Numerical Analysis*, 32(2):645–666, 1995. ISSN 00361429.

[84] Jörg Peters. Curvature continuous spline surfaces over irregular meshes. *Computer Aided Geometric Design*, 13(2):101 – 131, 1996. ISSN 0167-8396. doi: http://dx.doi.org/10.1016/0167-8396(95)00017-8.

[85] Jörg Peters. Patching Catmull-Clark Meshes. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 255–258, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co. ISBN 1-58113-208-5. doi: 10.1145/344779.344908.

[86] Jörg Peters. C2 free-form surfaces of degree (3,5). *Computer Aided Geometric Design*, 19(2):113 – 126, 2002. ISSN 0167-8396. doi: http://dx.doi.org/10.1016/S0167-8396(01)00081-4.

[87] Jörg Peters and Jianhua Fan. On the complexity of smooth spline surfaces from quad meshes. *Computer Aided Geometric Design*, 27(1):96 – 105, 2010. ISSN 0167-8396. doi: http://dx.doi.org/10.1016/j.cagd.2009.09.003.

[88] Jörg Peters and Ulrich Reif. *Subdivision surfaces*. Springer, 2008.

[89] Les Piegl and Wayne Tiller. *The NURBS book*. Monographs in Visual Communication. Springer, 1997.

[90] Hartmut Prautzsch and Ulrich Reif. Degree estimates for Ck-piecewise polynomial subdivision surfaces. *Advances in Computational Mathematics*, 10(2):209–217, February 1999. ISSN 1572-9044. doi: 10.1023/A:1018922530826.

[91] F. Pérez and J.A. Clemente. Constrained design of simple ship hulls with B-spline surfaces. *CAD Computer Aided Design*, 43(12):1829–1840, 2011.

[92] F. L. Pérez, J. A. Clemente, J. A. Suárez, and J. M. González. Parametric Generation, Modeling, and Fairing of Simple Hull Lines With the Use of Nonuniform Rational B-Spline Surfaces. *Journal of Ship Research*, 52(1):1–15, 2008. ISSN 0022-4502.

[93] F.L. Pérez-Arribas and E. Péter-Cosma. Parametric generation of planing hulls with NURBS surfaces. *Journal of Ship Research*, 57(4):241–261, 2013.

[94] Ulrich Reif. A degree estimate for subdivision surfaces of higher regularity. *Proceedings of the American Mathematical Society*, 124(7):2167–2174, 1996.

[95] J.-H. Rhim, D.-Y. Cho, K.-Y. Lee, and T.-W. Kim. Generation of discrete bicubic G 1 B-spline ship hullform surfaces from a given curve network using virtual iso-parametric curves. *Journal of Computer Science and Technology*, 21(2):265–271, 2006.

[96] David F Rogers. *An introduction to NURBS: with historical perspective*. Elsevier, 2000.

[97] 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. ISSN 0010-4485. doi: https://doi.org/10.1016/S0010-4485(02)00113-6.

[98] Ramon F. Sarraga. G1 interpolation of generally unrestricted cubic Bézier curves. *Computer Aided Geometric Design*, 4(1):23 – 39, 1987. ISSN 0167-8396. doi: https://doi.org/10.1016/0167-8396(87)90022-7.

[99] Scott Schaefer, Joe Warren, and Denis Zorin. Lofting curve networks using subdivision surfaces. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 103–114. ACM, 2004.

[100] Thomas W. Sederberg, Jianmin Zheng, David Sewell, and Malcolm Sabin. Non-uniform Recursive Subdivision Surfaces. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 387–394, New York, NY, USA, 1998. ACM. ISBN 0-89791-999-8. doi: 10.1145/280814.280942.

[101] Thomas W. Sederberg, Jianmin Zheng, Almaz Bakenov, and Ahmad Nasri. T-splines and T-NURCCs. *ACM Trans. Graph.*, 22(3):477–484, 2003. ISSN 0730-0301. doi: 10.1145/882262.882295.

[102] R Sharma, Tae-wan Kim, Richard Lee Storch, Hans JJ Hopman, and Stein Ove Erikstad. Challenges in computer applications for ship and floating structure design and analysis. *Computer-Aided Design*, 44(3):166–185, 2012.

[103] Jos Stam. Exact Evaluation of Catmull-Clark Subdivision Surfaces at Arbitrary Parameter Values. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 395–404, New York, NY, USA, 1998. ACM. ISBN 0-89791-999-8. doi: 10.1145/280814.280945.

[104] Jos Stam. On subdivision schemes generalizing uniform B-spline surfaces of arbitrary degree. *Computer Aided Geometric Design*, 18(5):383–396, 2001. ISSN 0167-8396. doi: 10.1016/S0167-8396(01)00038-3.

[105] Ian F. Stewart and André R. Foisy. Arbitrary-Degree Subdivision with Creases and Attributes. *Journal of Graphics Tools*, 9(4):3–17, 2004. doi: 10.1080/10867651.2004.10504898.

[106] Huawei Wang, Kaihuai Qin, and Ron Kikinis. Exact evaluation of NURSS at arbitrary parameter values. In *Proceedings of the IASTED International Conference on Computer Graphics and Imaging*, pages 169–174. Las Vegas, USA, 2000.

[107] Geir Westgaard and Horst Nowacki. Construction of Fair Surfaces over Irregular Meshes. In *Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications*, SMA '01, pages 88–98, New York, NY, USA, 2001. ACM. ISBN 1-58113-366-9. doi: 10.1145/376957.376969.

[108] Jarke J. Van Wijk. Bicubic patches for approximating non-rectangular control-point meshes. *Computer Aided Geometric Design*, 3(1):1 – 13, 1986. ISSN 0167-8396. doi: http://dx.doi.org/10.1016/0167-8396(86)90021-X.

[109] Yasushi Yamaguchi. A basic evaluation method of subdivision surfaces. *Journal for Geometry and Graphics*, 5(2):145–155, 2001.

[110] Xiuzi Ye. Curvature continuous interpolation of curve meshes. *Computer Aided Geometric Design*, 14(2):169 – 190, 1997. ISSN 0167-8396. doi: https://doi.org/10.1016/S0167-8396(96)00027-1.

[111] Xiuzi Ye and Horst Nowacki. Ensuring compatibility of G2-continuous surface patches around a nodepoint. *Computer Aided Geometric Design*, 13(9):931 – 949, 1996. ISSN 0167-8396.

[112] Denis Zorin. Constructing Curvature-continuous Surfaces by Blending. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pages 31–40, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association. ISBN 3-905673-36-3.

[113] Denis Zorin and Daniel Kristjansson. Evaluation of piecewise smooth subdivision surfaces. *The Visual Computer*, 18(5):299–315, 2002.

[114] Denis Zorin and Peter Schröder. A unified framework for primal/dual quadrilateral subdivision schemes. *Computer Aided Geometric Design*, 18(5):429 – 454, 2001. ISSN 0167-8396. doi: 10.1016/S0167-8396(01)00040-1.

A Container vessel

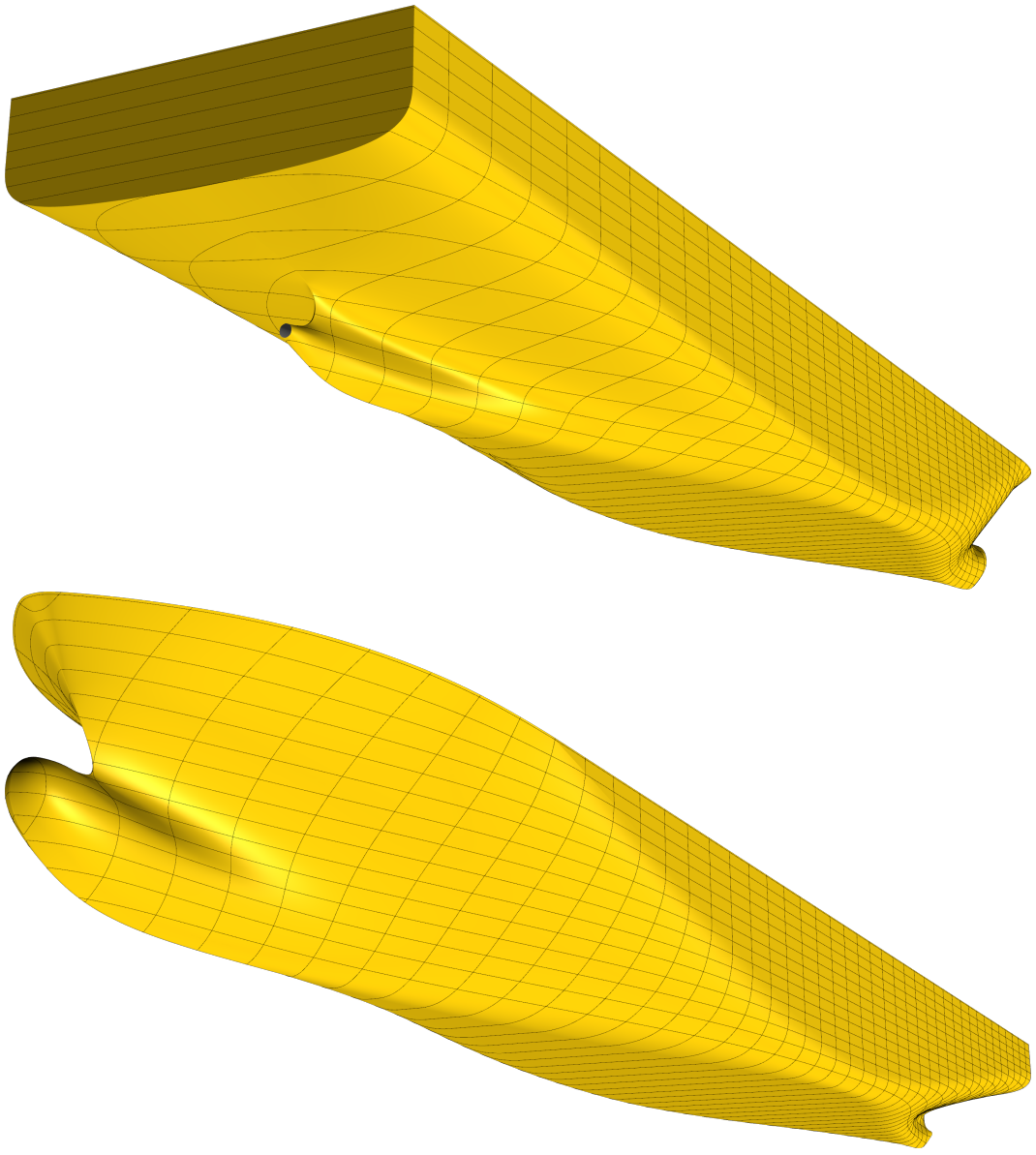


Figure A.1.: Generalized B-spline surface rendered with sections and waterlines.

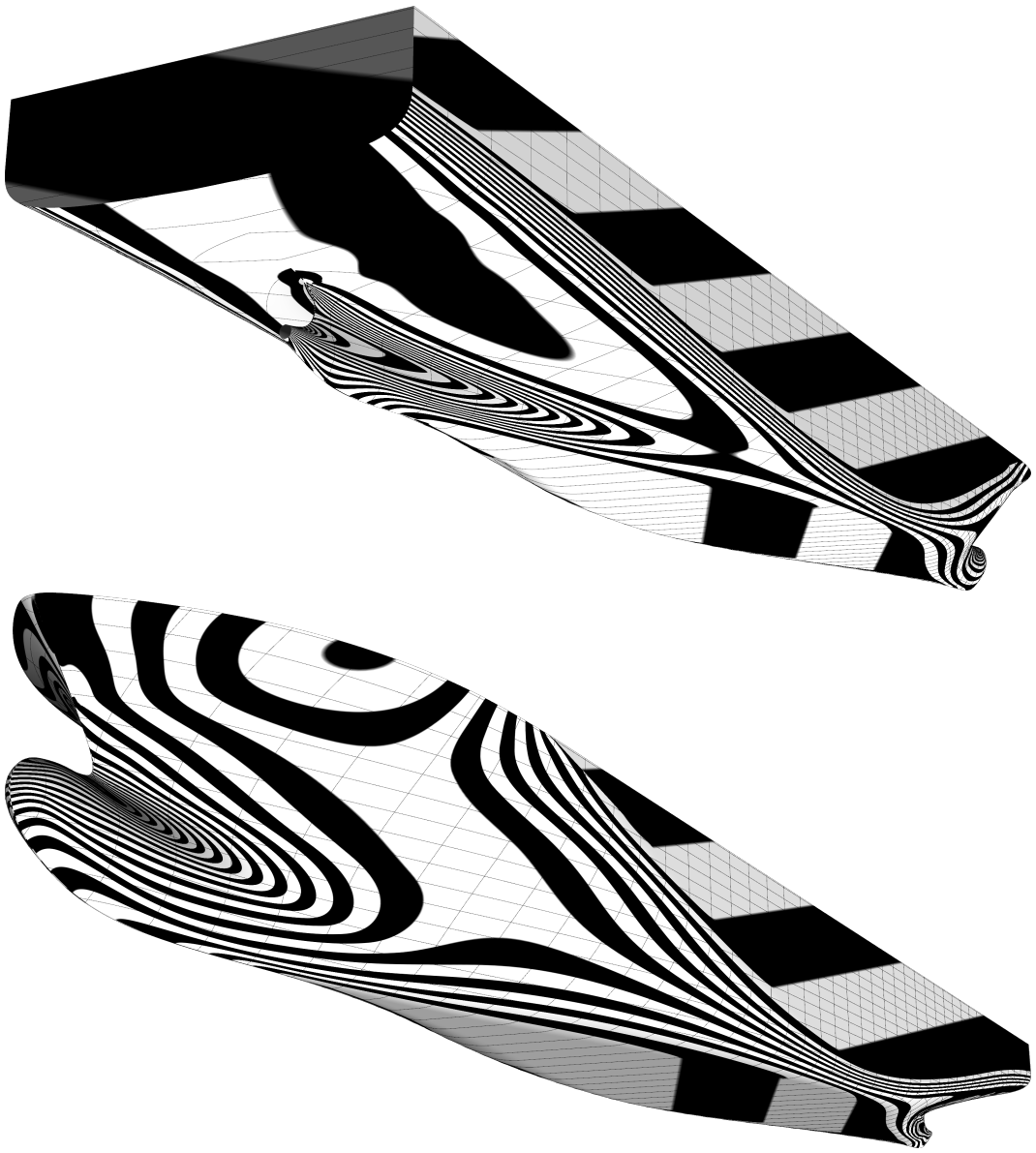


Figure A.2.: Reflection analysis of the generalized B-spline surface.

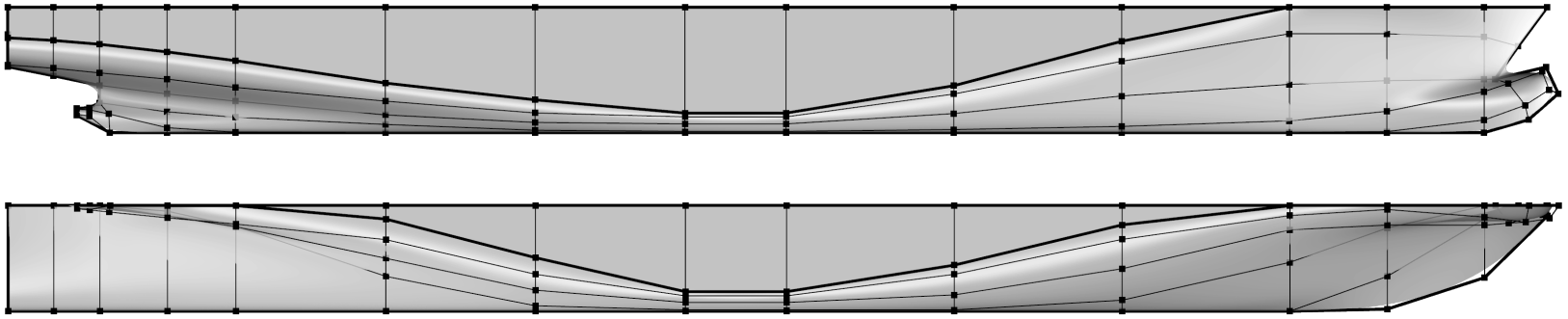


Figure A.3.: Control mesh $\mathbf{Q}^0$ of the generalized B-spline surface.

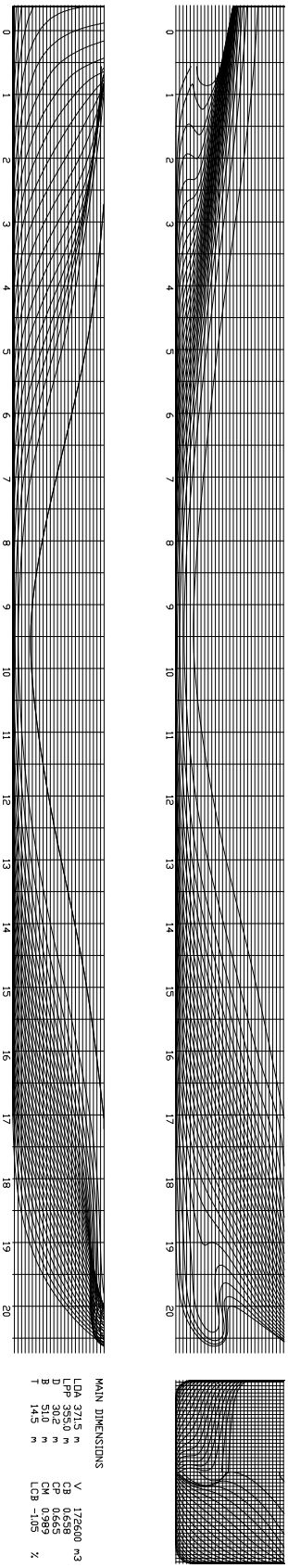


Figure A.4.: Lines plan derived from the generalized B-spline surface.

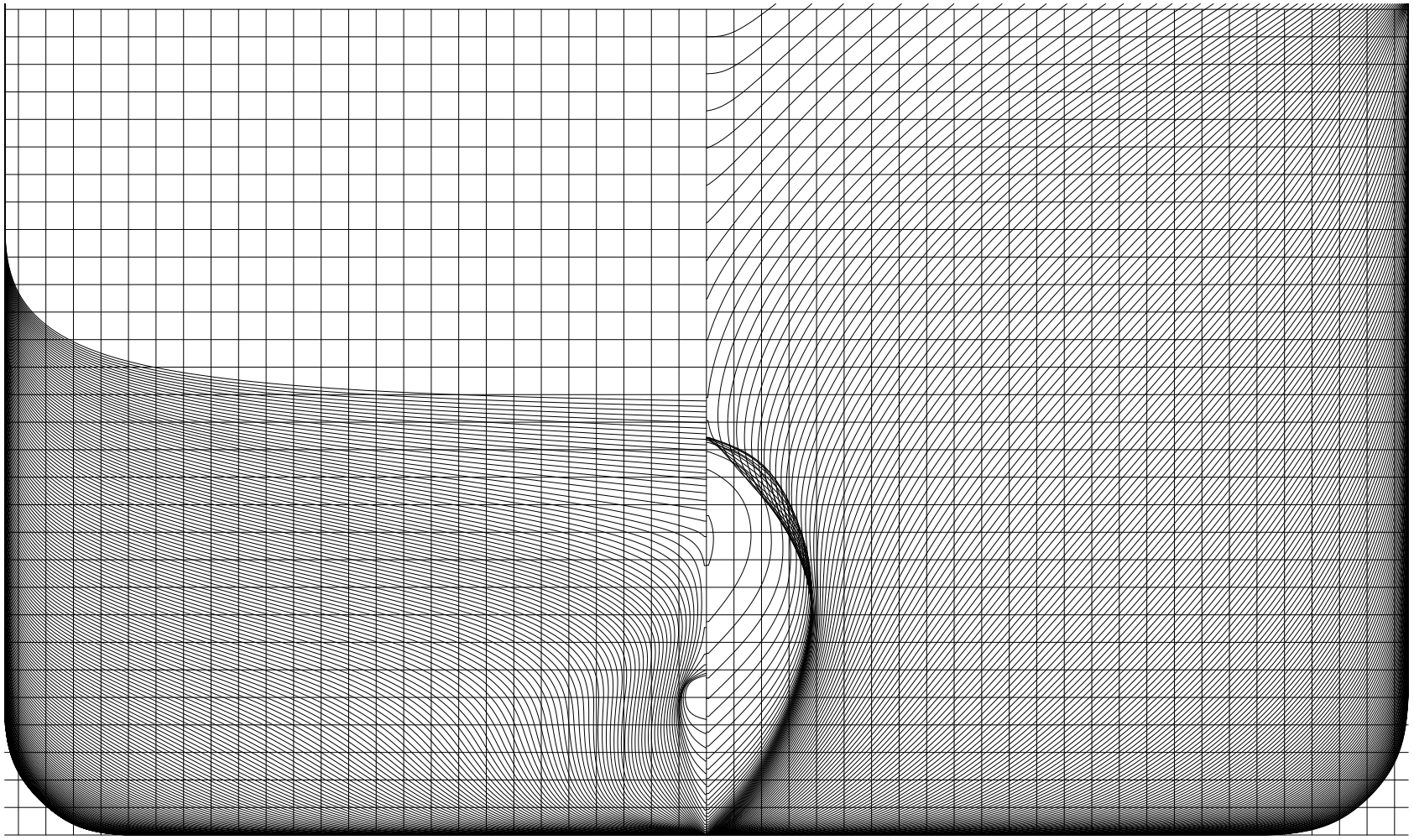


Figure A.5.: Sections plan derived from the generalized B-spline surface.

This page intentionally left blank.

B Bulk carrier

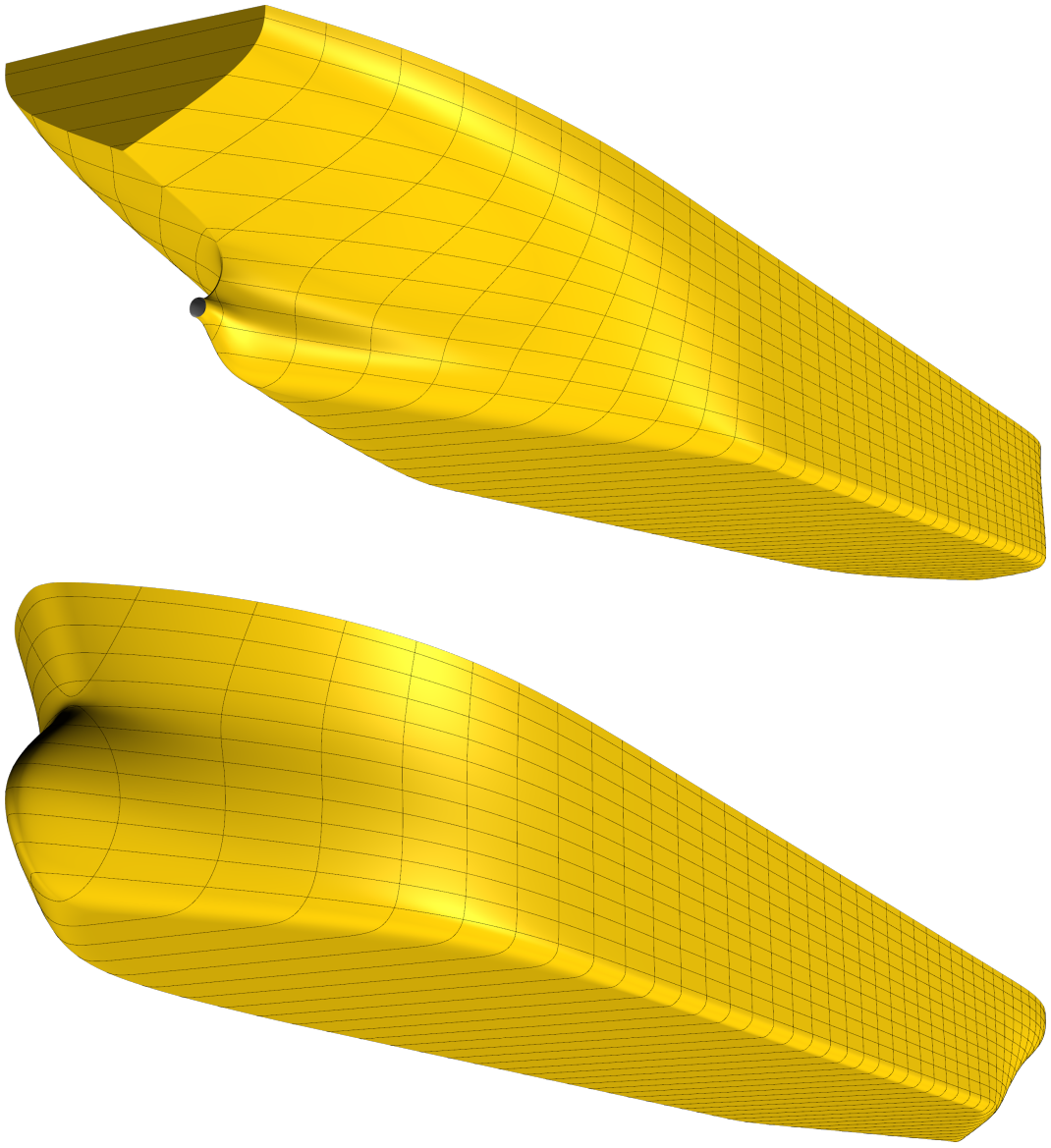


Figure B.1.: Generalized B-spline surface rendered with sections and waterlines.

Figure B.2.: Reflection analysis of the generalized B-spline surface.

Figure B.3.: Control mesh $\mathbf{Q}^0$ of the generalized B-spline surface.

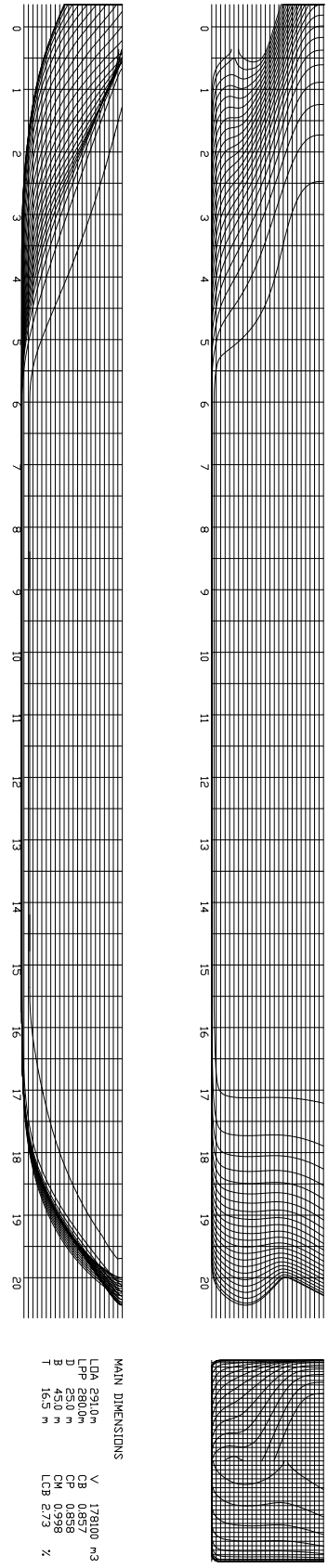


Figure B.4.: Lines plan derived from the generalized B-spline surface.

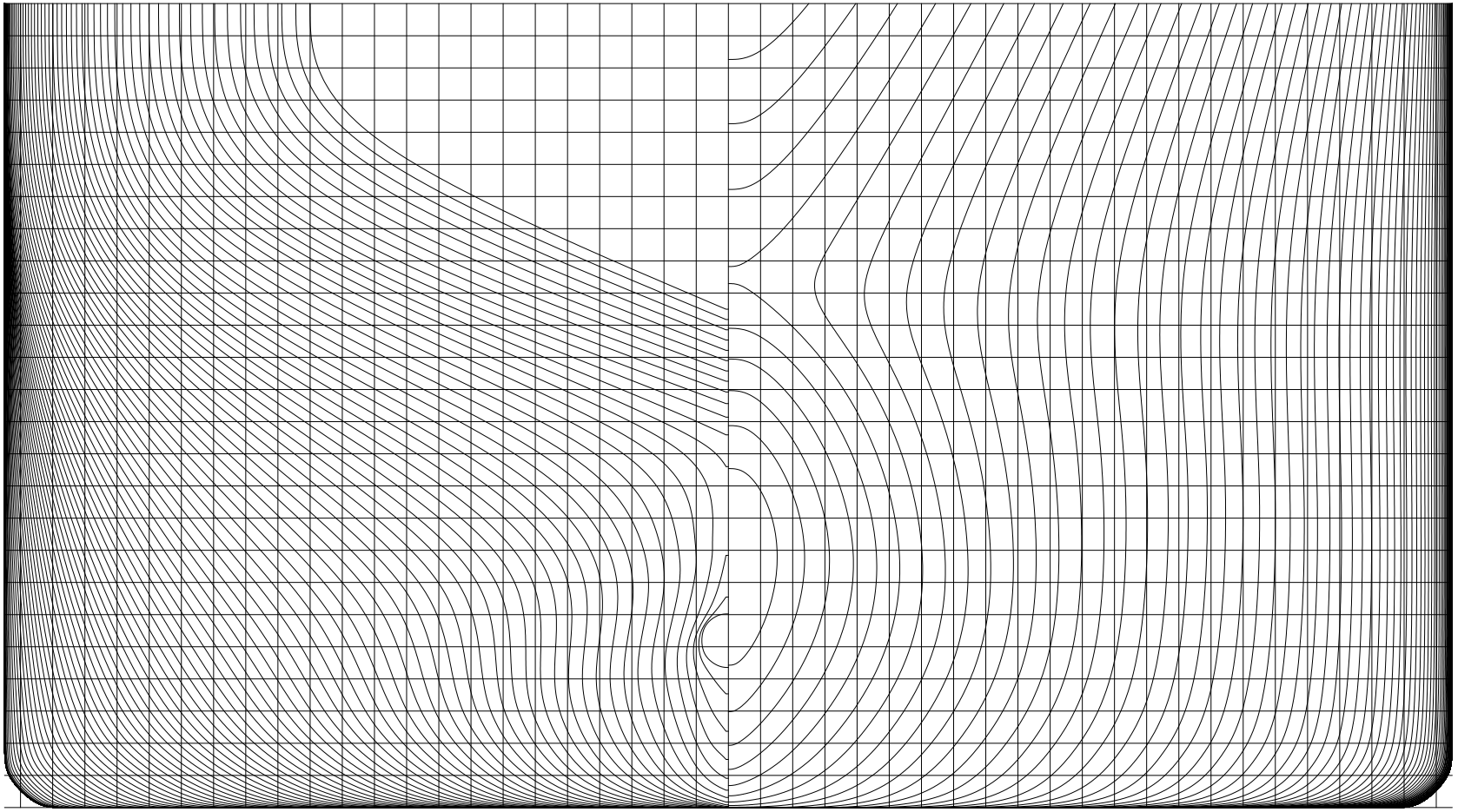


Figure B.5.: Sections plan derived from the generalized B-spline surface.

This page intentionally left blank.

C RoRo vessel

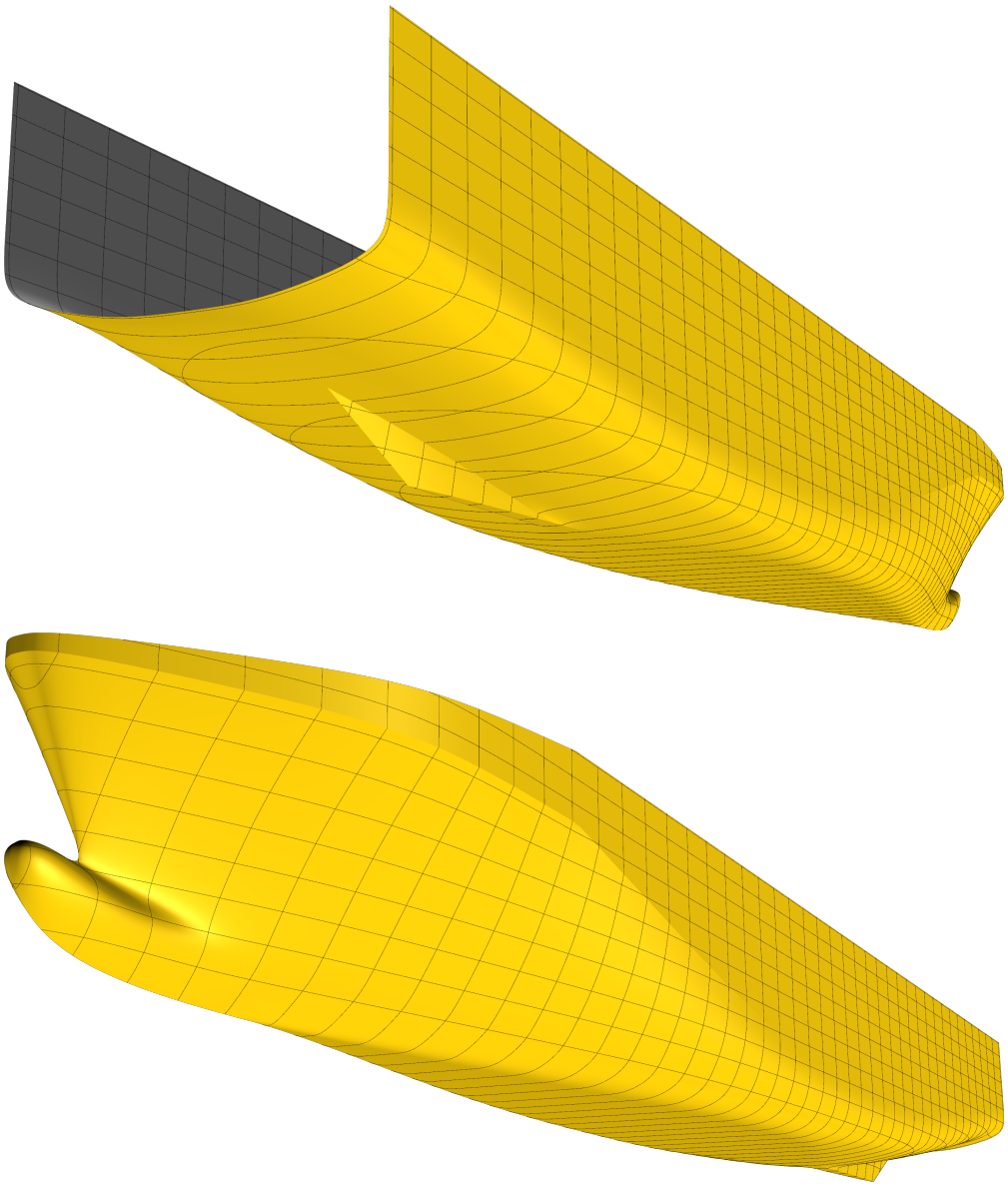


Figure C.1.: Generalized B-spline surface rendered with sections and waterlines.

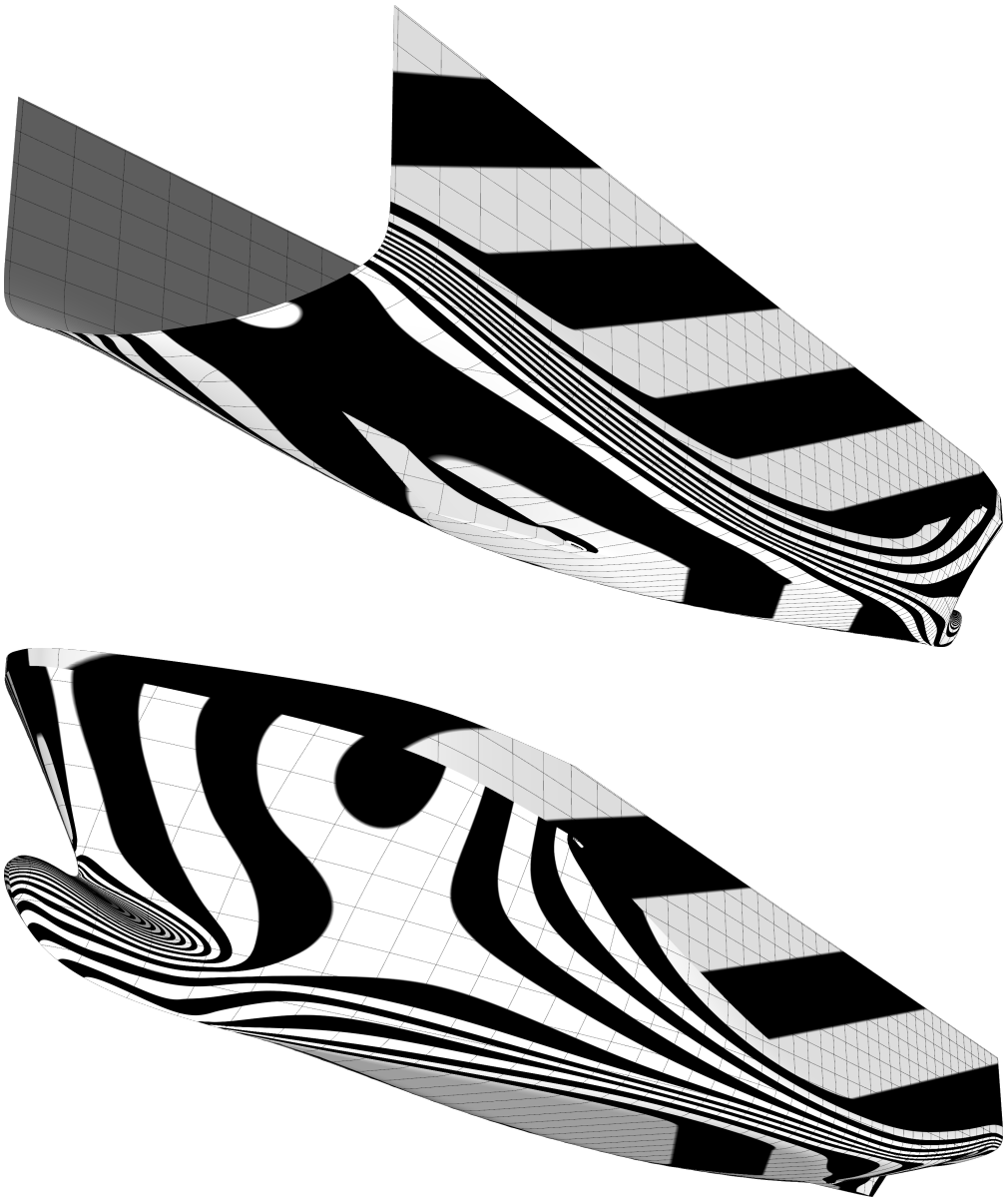


Figure C.2.: Reflection analysis of the generalized B-spline surface.

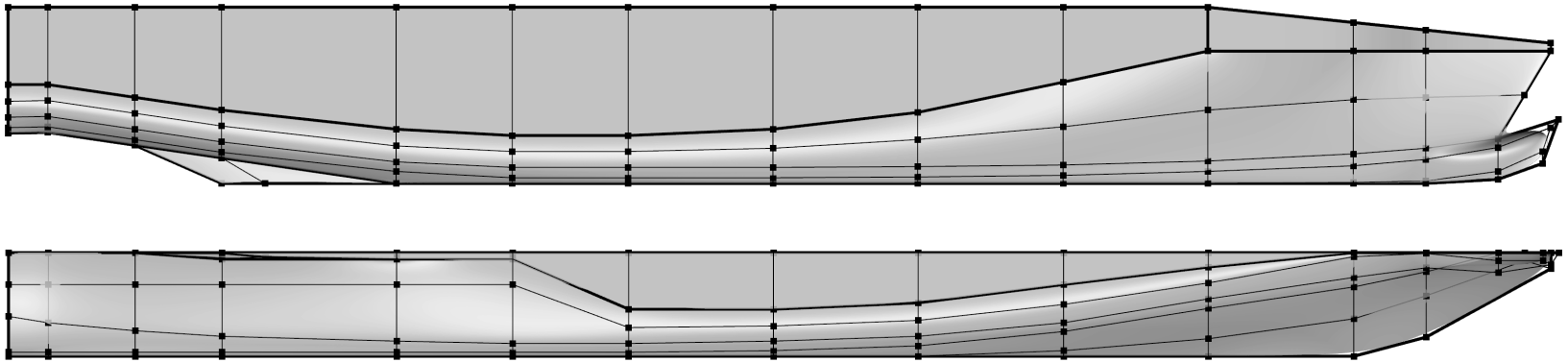


Figure C.3.: Control mesh $\mathbf{Q}^0$ of the generalized B-spline surface.

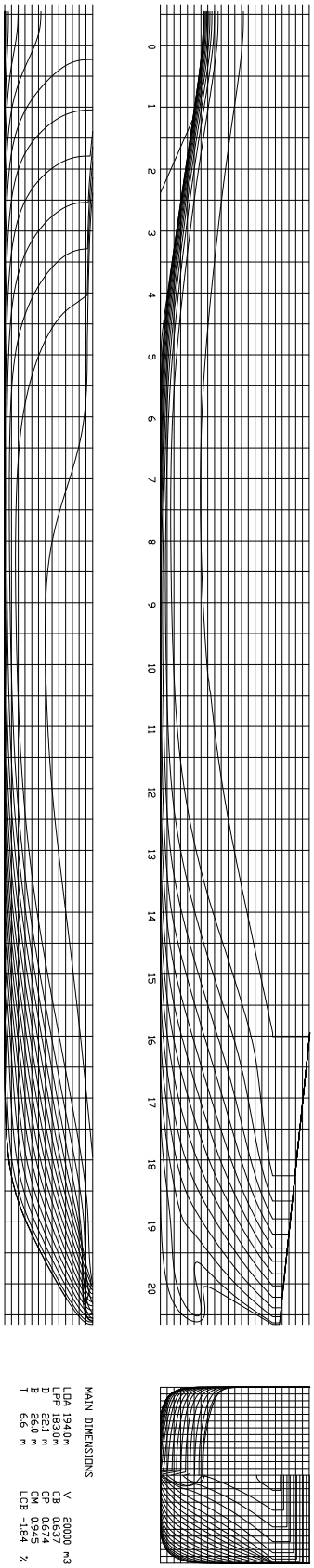


Figure C.4.: Lines plan derived from the generalized B-spline surface.

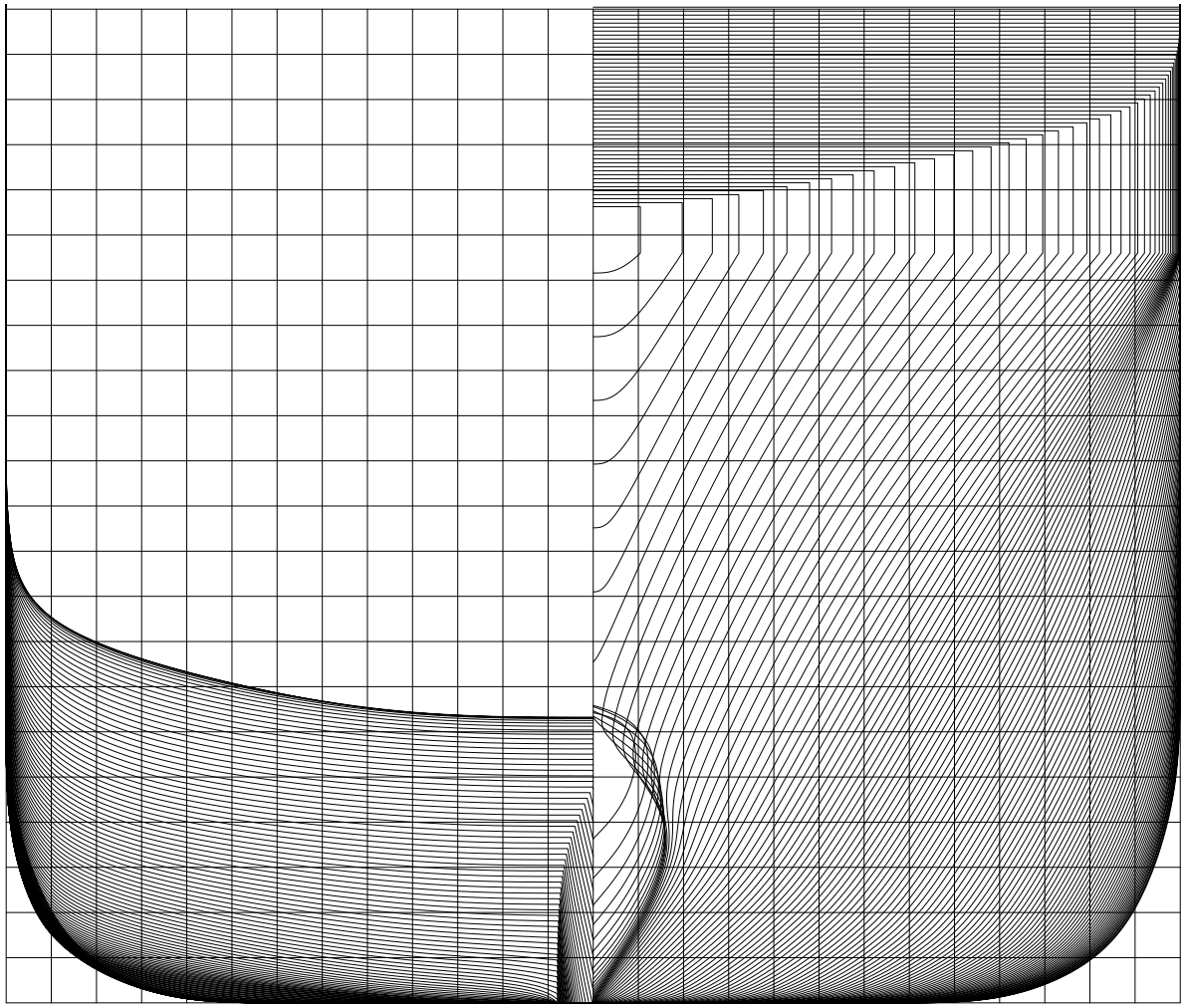


Figure C.5.: Sections plan derived from the generalized B-spline surface.

This page intentionally left blank.

D Barge

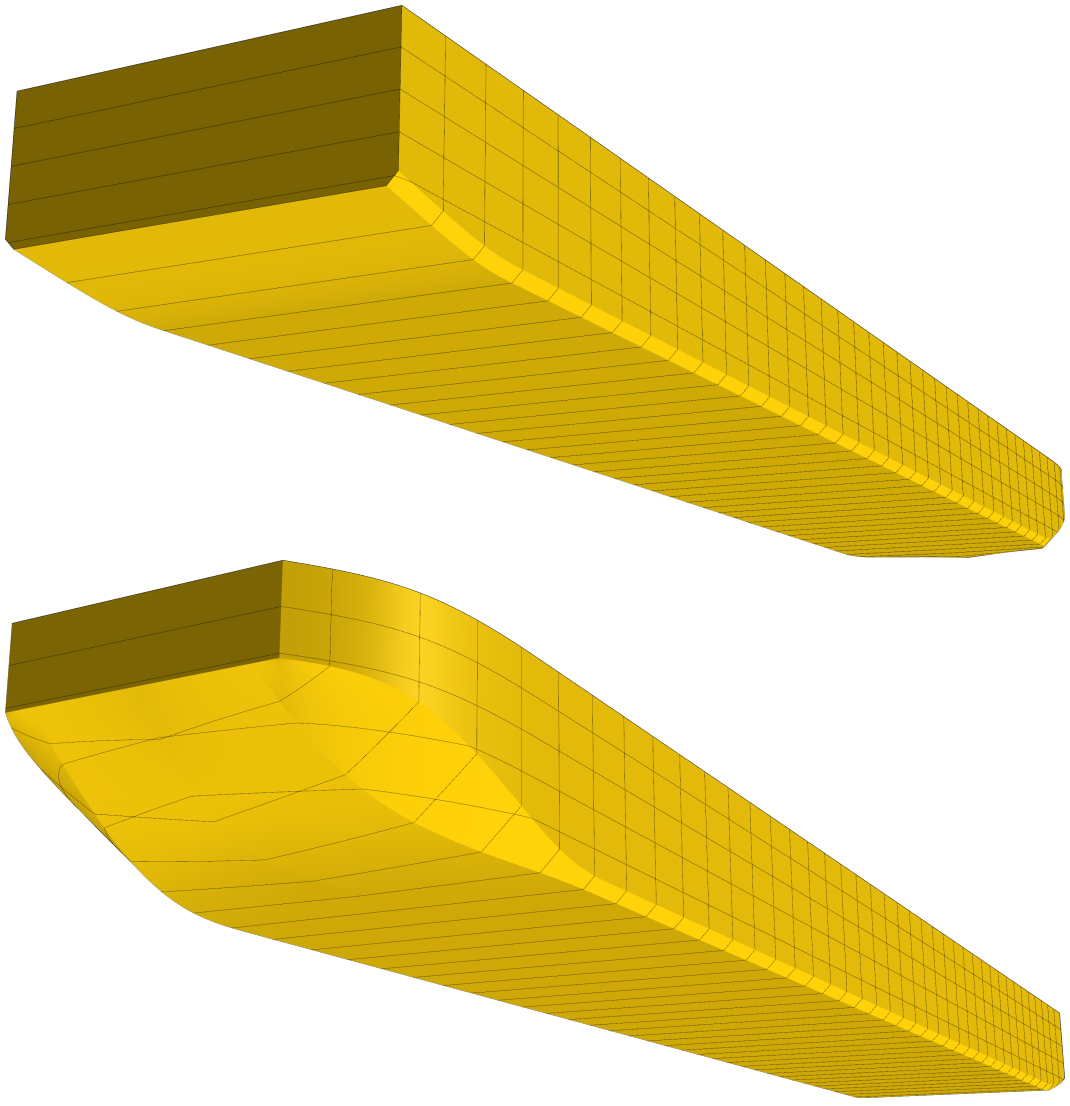


Figure D.1.: Generalized B-spline surface rendered with sections and waterlines.

Figure D.2.: Reflection analysis of the generalized B-spline surface.

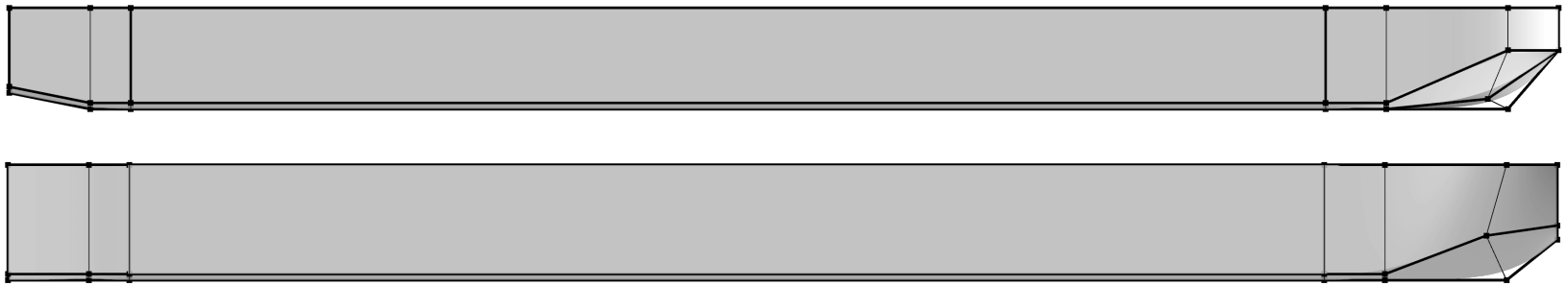


Figure D.3.: Control mesh $\mathbf{Q}^0$ of the generalized B-spline surface.

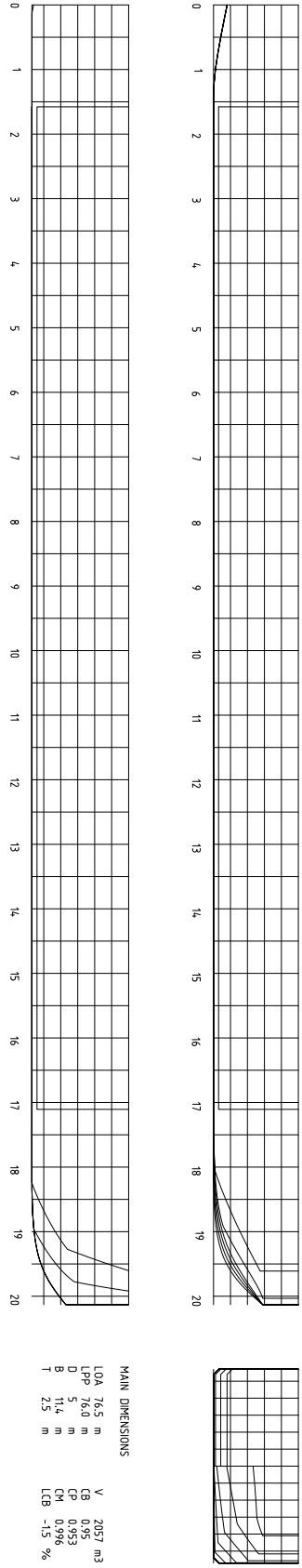


Figure D.4.: Lines plan derived from the generalized B-spline surface.

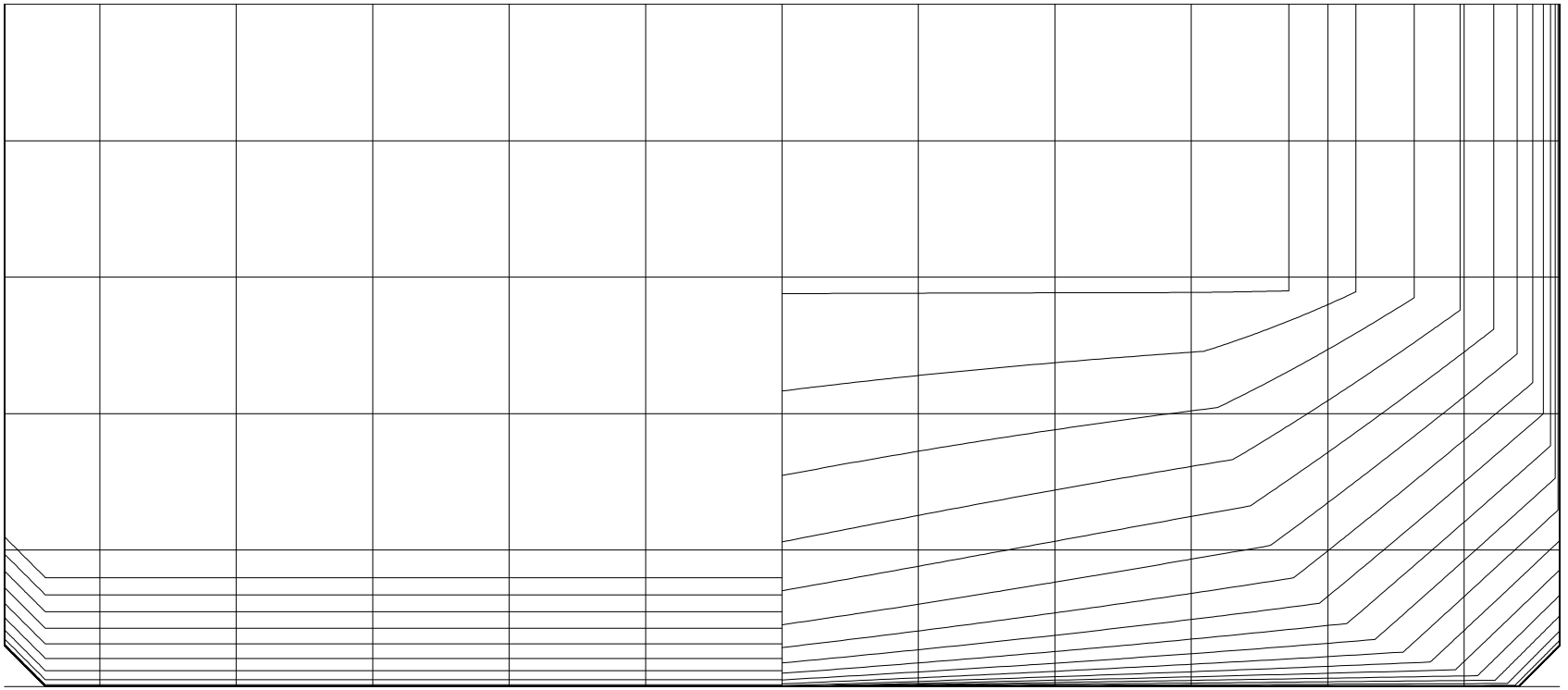


Figure D.5.: Sections plan derived from the generalized B-spline surface.

This page intentionally left blank.

E Research vessel

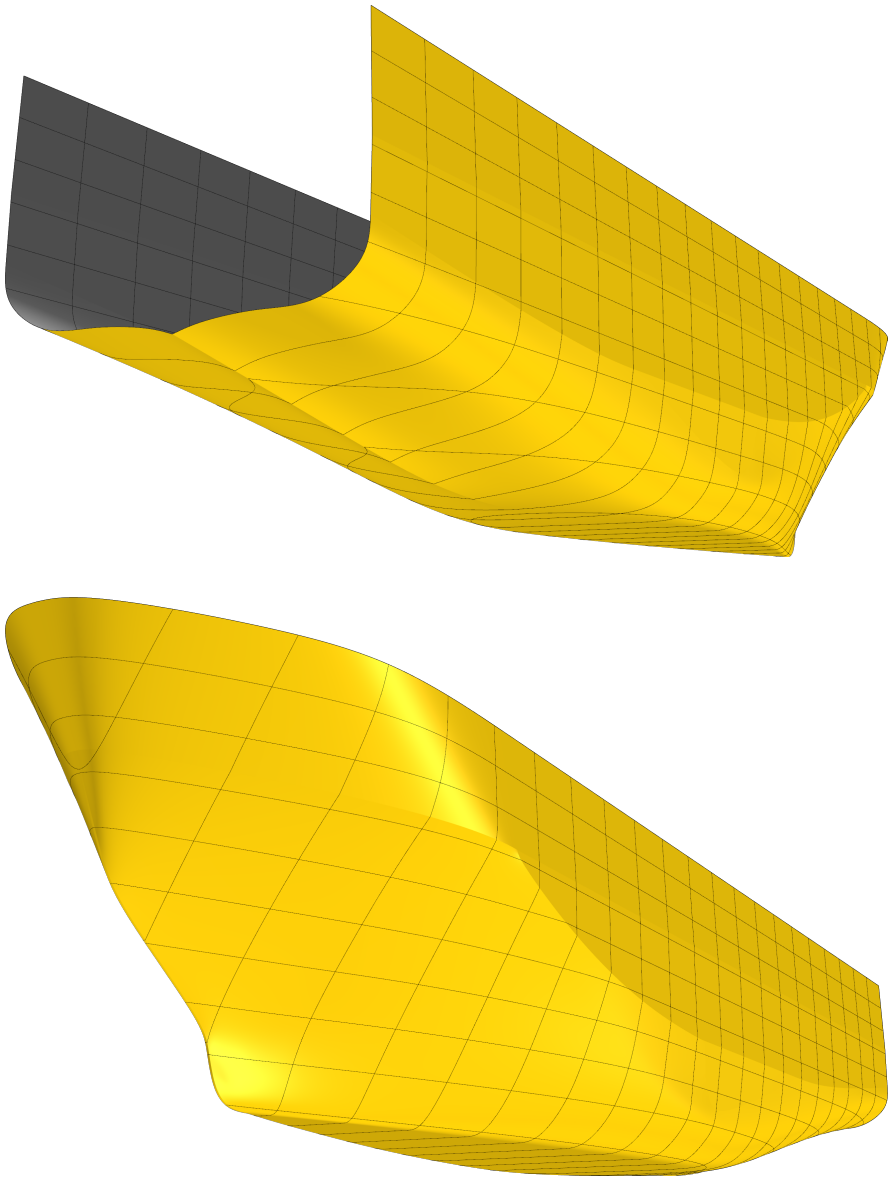


Figure E.1.: Generalized B-spline surface rendered with sections and waterlines.

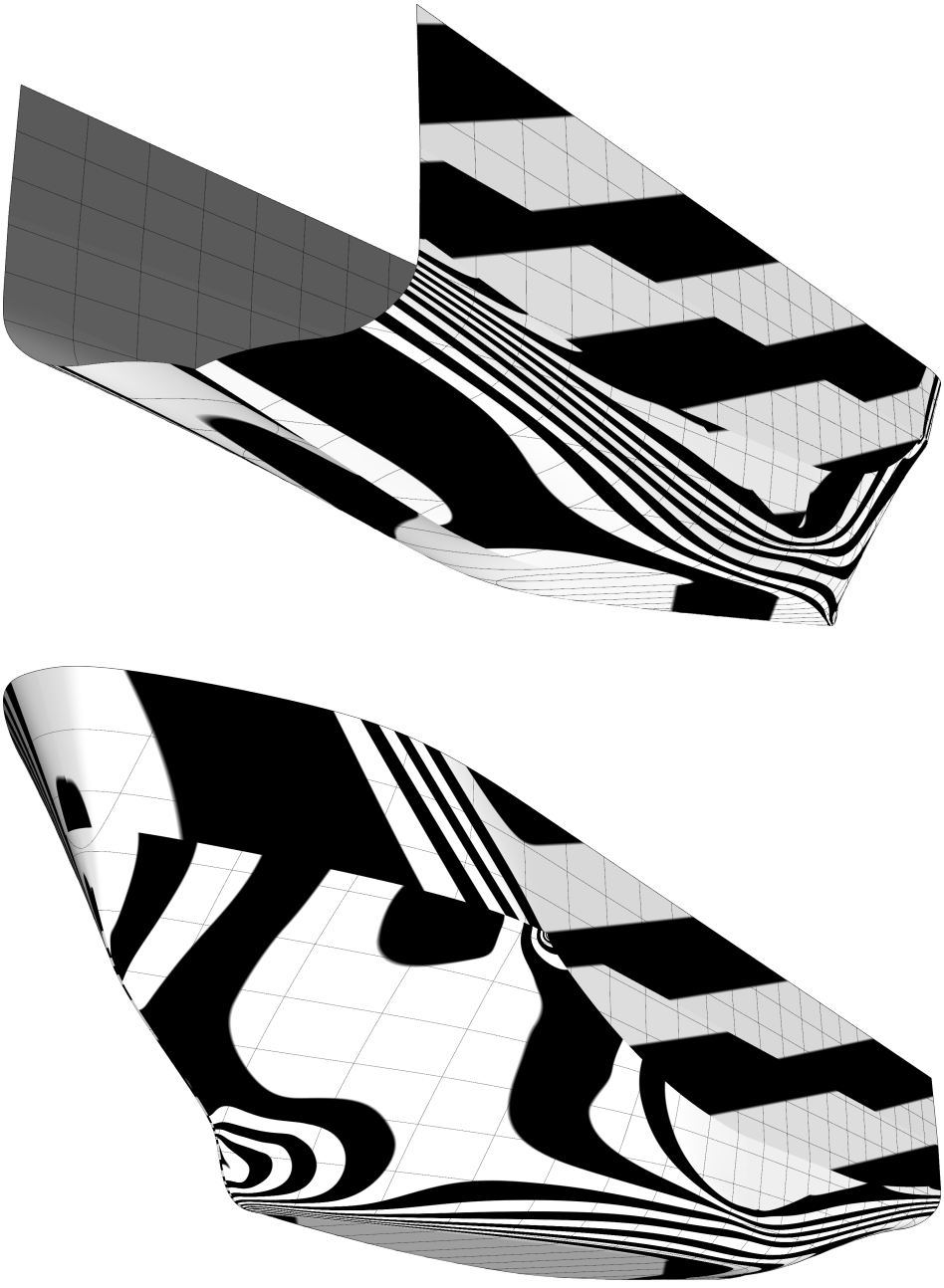


Figure E.2.: Reflection analysis of the generalized B-spline surface.

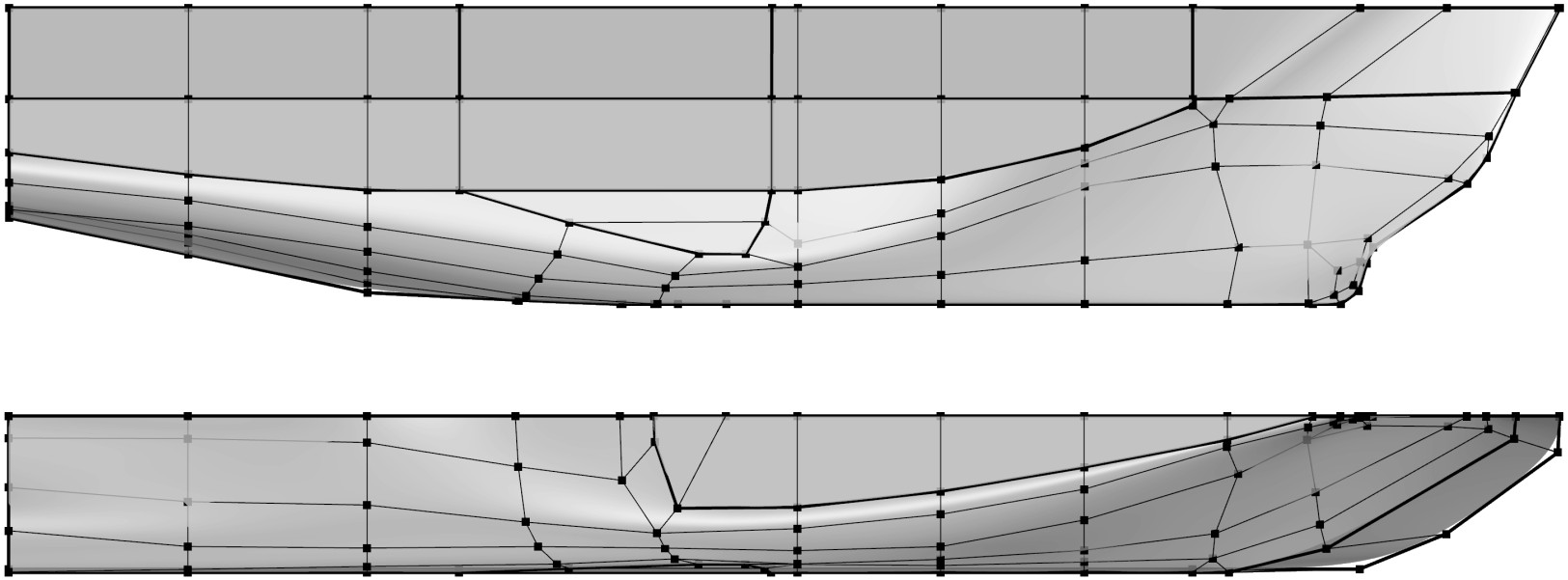


Figure E.3.: Control mesh $\mathbf{Q}^0$ of the generalized B-spline surface.

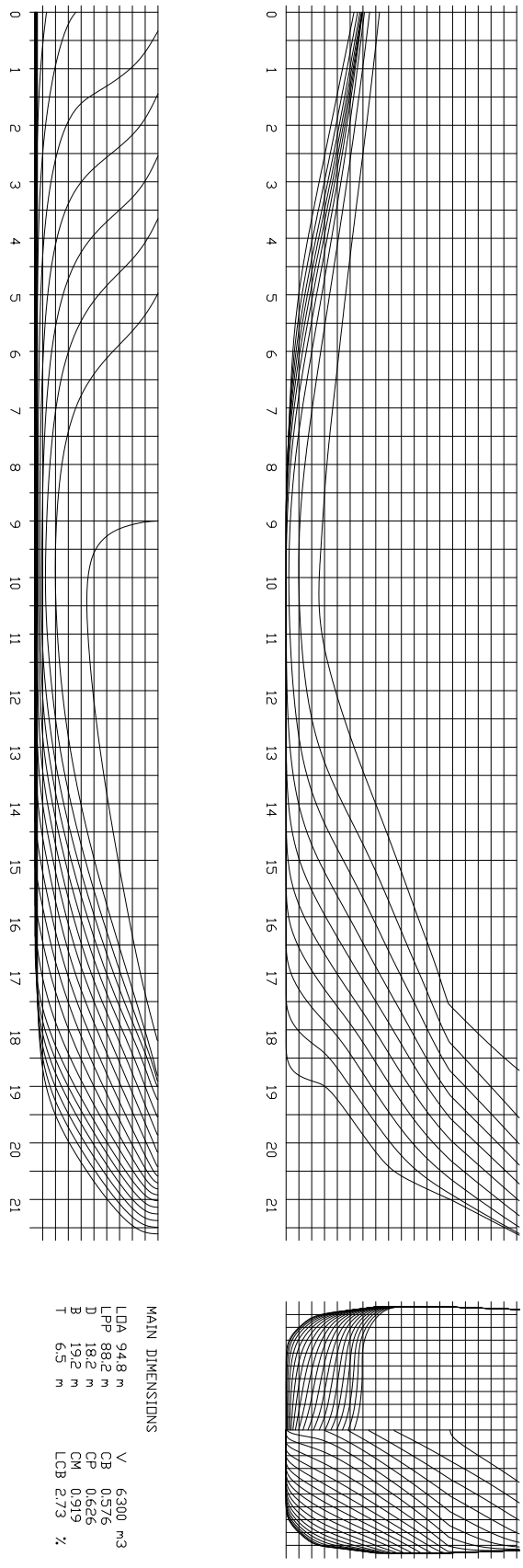


Figure E.4.: Lines plan derived from the generalized B-spline surface.

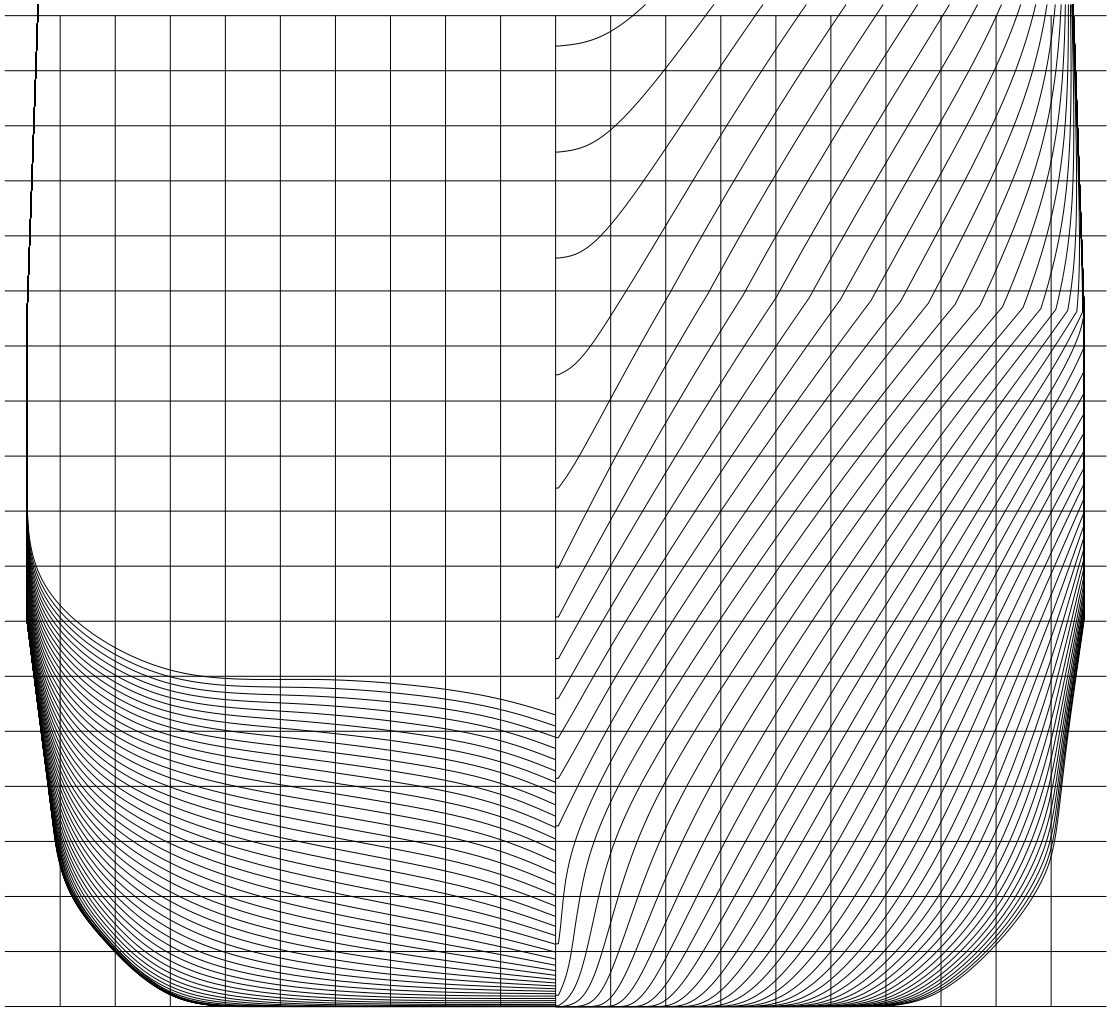


Figure E.5.: Sections plan derived from the generalized B-spline surface.

This page intentionally left blank.

F Naval vessel

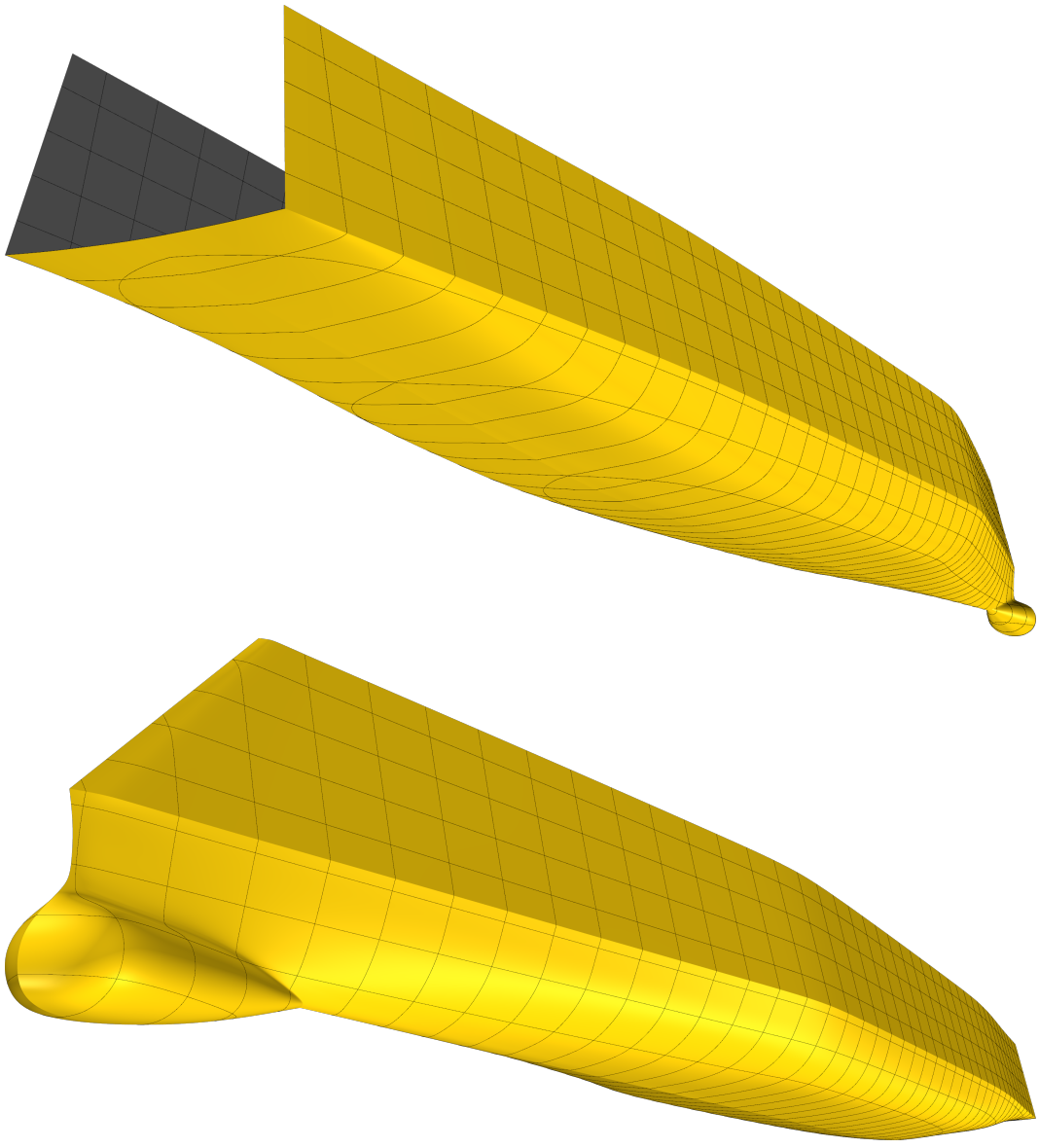


Figure F.1.: Generalized B-spline surface rendered with sections and waterlines.

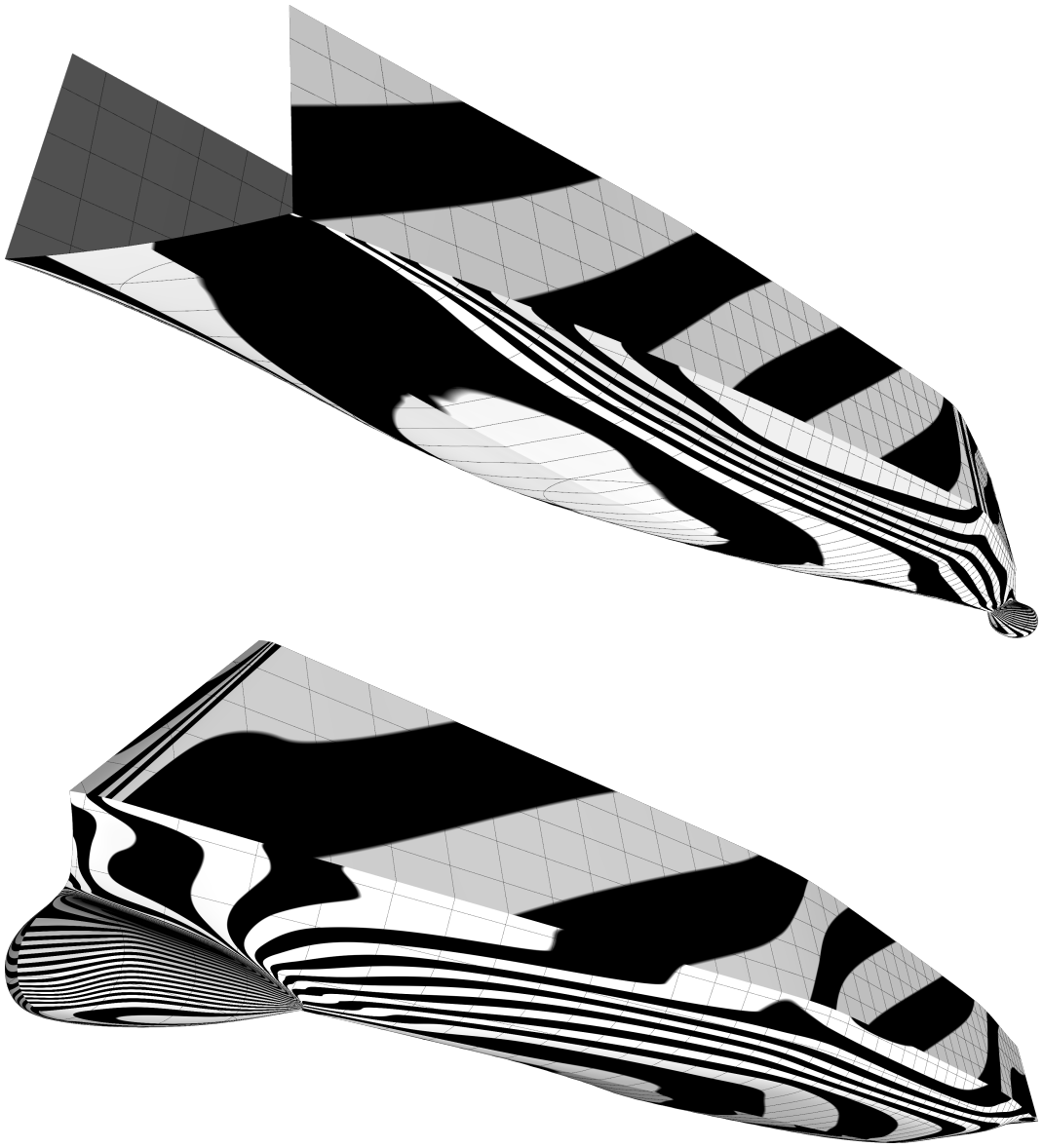


Figure F.2.: Reflection analysis of the generalized B-spline surface.

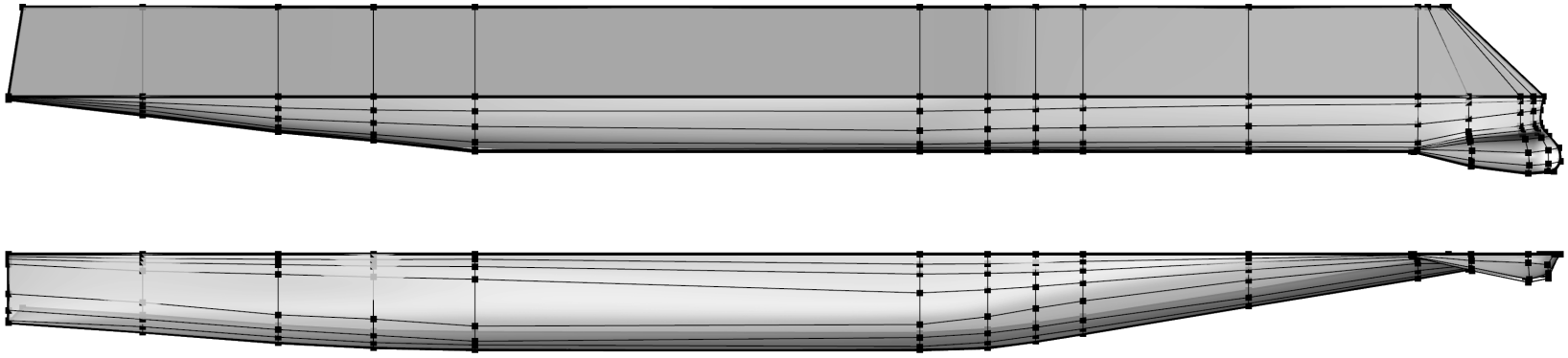


Figure F.3.: Control mesh $\mathbf{Q}^0$ of the generalized B-spline surface.

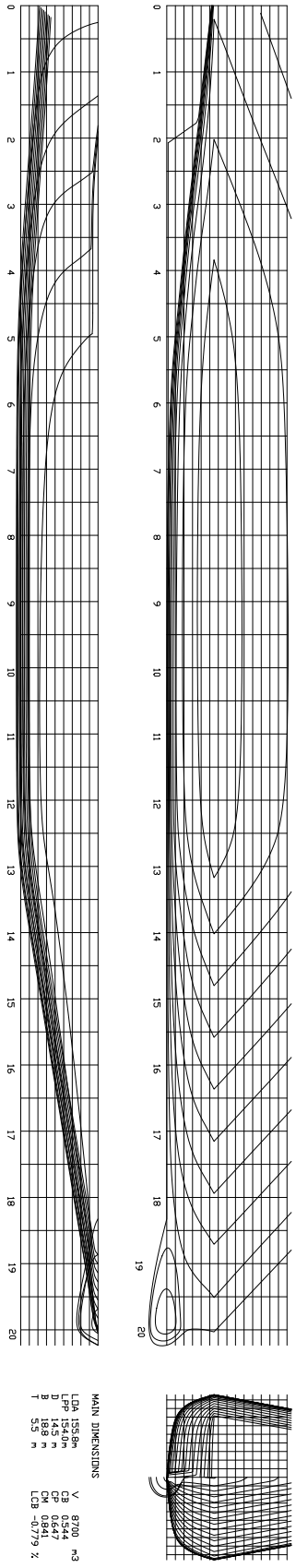


Figure F.4.: Lines plan derived from the generalized B-spline surface.

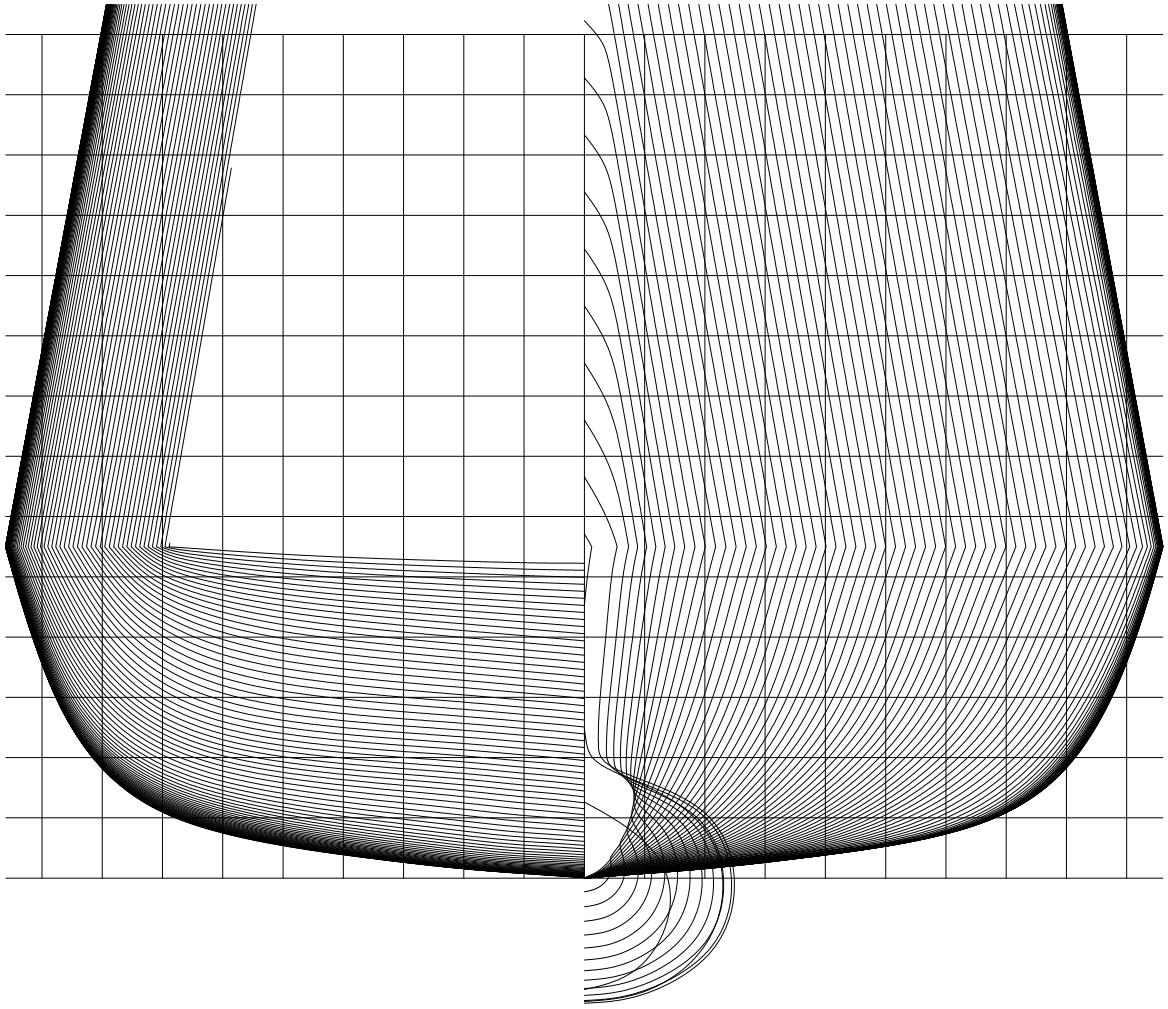


Figure F.5.: Sections plan derived from the generalized B-spline surface.

This page intentionally left blank.

G Remarks on smoothness

The terms *smoothness* and *continuity* are occasionally used imprecisely and as a consequence may lead to confusion. For the scope of this thesis, the usage of these terms is defined precisely in this chapter. The terms are discussed on the basis of curves, but the definitions apply equally well to surfaces.

Consider an indexed series of parametric curve segments

$$\mathbf{x}_i : U_i \ni u \mapsto \mathbf{x}_i(u) \in \mathbb{R}^d \tag{G.1}$$

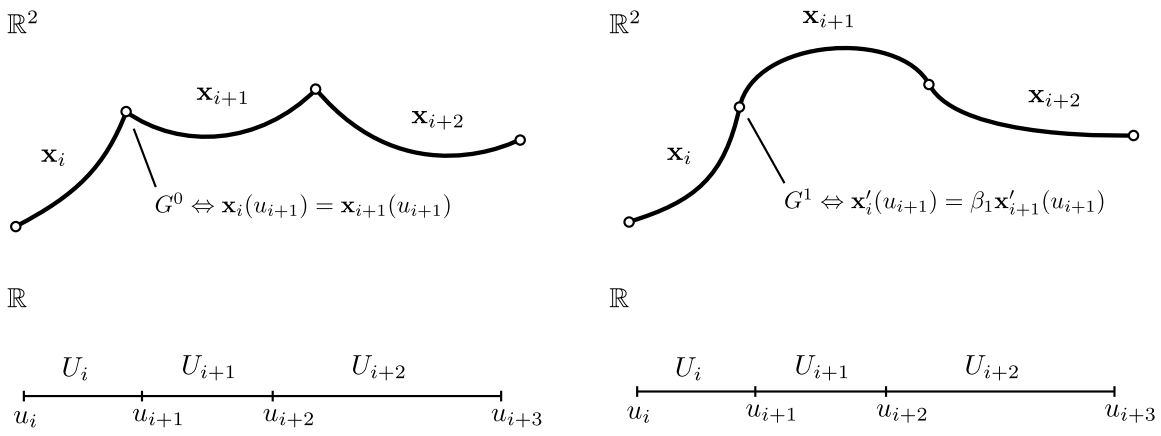
defined over the interval $U_i = [u_i, u_{i+1}]$ with $u_{i+1} \geq u_i$. The segments are supposed to be connected at their endpoints with

$$\mathbf{x}_i(u_{i+1}) = \mathbf{x}_{i+1}(u_{i+1}) \tag{G.2}$$

what is referred to as position continuity because the segments altogether represent at least a continuous curve, see Figure G.1(a) for clarification. As illustrated this does not imply smoothness. The given example has cusps at the connections points what intuitively prevents the overall curve to be considered as smooth. Figure G.1(b) shows another curve where all segments are smoothly connected to each other. In contrast to the previous example, the tangents are continuous at the connection points what is for certain applications sufficient to be considered as an overall smooth curve. The term *smoothness* is apparently used in a geometrical sense and as discussed below this is the most appropriate usage in the context of computer-aided design.

In Figure G.2(a) two quadratic Bézier segments are used to construct an example to introduce the concepts of *parametric continuity* and *geometric continuity*. The advantage of the Bézier constructions is that the segments $\mathbf{x}_i$ can be expressed by a polynomial for each coordinate direction over the entire domain U_i. With reference to the control points specified in Figure G.2, the polynomial representation of the first segment is

$$\mathbf{x}_1 = \begin{pmatrix} 2u \\ 2u^2 - 2u + 1 \end{pmatrix} \tag{G.3}$$



(a) Position continuity

(b) Tangent continuity

Figure G.1.: An indexed series of parametric curve segments $\mathbf{x}_i$ defined over the domain U_i is stitched together to form a single curve. In a generalized sense this is known as a spline curve. The segments are pairwise connected at their endpoints with $\mathbf{x}_i(u_{i+1}) = \mathbf{x}_{i+1}(u_{i+1})$ what is called position continuity (G^0). However, as illustrated on the left side this does not constitute a smooth curve because the curve features cusps at the connection points. On the right side, the tangents are continuous (G^1) at the connection points and for certain applications this already suffices to consider the curve as smooth.

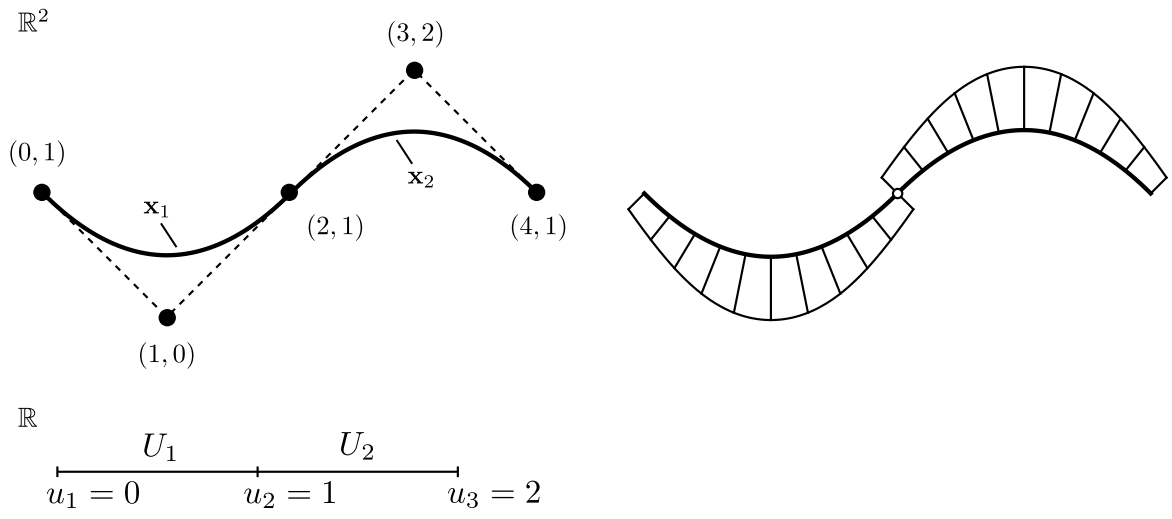
with $U_1 = [0, 1]$ and the second segment is represented by

$$\mathbf{x}_2 = \begin{pmatrix} 2u \\ -2u^2 + 6u - 3 \end{pmatrix} \tag{G.4}$$

with $U_2 = [1, 2]$. C^n denotes that a function is n times continuously differentiable what is referred to as *parametric continuity*. The segment's polynomial representations are infinitely differentiable and therefore said to be C^∞. While the parametric differentiability of the segments is usually given, for instance the segments of the popular B-spline curves are always C^∞, the same is generally not true at the connection points. For the running example, it is by inspection shown that with

$$\begin{aligned} \mathbf{x}_1(1) &= \mathbf{x}_2(1) \\ \mathbf{x}_1'(1) &= \mathbf{x}_2'(1) \\ \mathbf{x}_1''(1) &\neq \mathbf{x}_2''(1) \\ &\vdots \end{aligned} \tag{G.5}$$

both curves meet in a common point as well as the first derivative agrees at the connection



(a) Bézier segments (b) Curvature graph

Figure G.2.: Two cubic Bézier segments $\mathbf{x}_1$ and $\mathbf{x}_2$ are connected to each other. Left: The control points and the domains are specified of the segments are specified. Both segments seem to be smoothly connected to each other. Right: At the connection point of both segments the curvature graph is discontinuous and the curve is therefore only G^1. Whether a lack of high-order continuity at the connection point can be considered as smooth depends on the application context.

point, but higher derivatives generally disagree. The overall curve is therefore said to be C^1 only. This seems to verify what is intuitively seen in Figure G.2, namely that the curve is smooth, but apparently lacks high-order smoothness as depicted by the discontinuous curvature graph. At the connection point of both curves, the curvature graph takes the form a step function what represents a discontinuity of the curvature and consequently a lack of high-order smoothness. At the first glance, the analysis of parametric continuity thus seems to be reasonable measure of smoothness. However, a simple reparameterization of $\mathbf{x}_2$ with $u = \tilde{u}^2$ changes the picture and shows that parametric continuity is in fact a weak notion of the smoothness.

Applying $u = \tilde{u}^2$, the second segment's parametrization reads

$$\tilde{\mathbf{x}}_2(\tilde{u}) = \begin{pmatrix} 2\tilde{u}^2 + 2 \\ -2\tilde{u}^4 + 2\tilde{u}^2 + 1 \end{pmatrix} \tag{G.6}$$

with $\tilde{U}_2 = [0, 1]$. Reparametrization does not affect the curve's geometry, so plotting $\mathbf{x}_2$ and $\tilde{\mathbf{x}}_2$ yields the same graph. What changes, though, are the differential properties at the

connection point of both segments. By inspection it is easily found that

$$\begin{aligned} \mathbf{x}_1(1) &= \tilde{\mathbf{x}}_2(1) \\ \mathbf{x}_1'(1) &\neq \tilde{\mathbf{x}}_2'(1) \\ \mathbf{x}_1''(1) &\neq \tilde{\mathbf{x}}_2''(1) \\ &\vdots \end{aligned} \tag{G.7}$$

and both segments still meet in a common point, but generally all derivatives are different at this point and consequently the curves is said to be C^0 only. This is not satisfying because different conclusions are drawn for geometrically one and the same curve based on the analysis of parametric continuity.

This insight motivates to introduce another continuity measure which is independent of the actual parametrization called *geometric continuity* and shortly referred to as G^n. A descriptive characterization of low-order geometric continuity is: two segments are *position continuous* (G^0) when they have common end points, two segments are *tangent continuous* (G^1) when they additionally have common tangents, and two segments are *curvature continuous* (G^2) when they moreover have common curvature vectors. This is, however, only an informal description of geometric continuity to clarify how geometric continuity is formulated in terms of geometric properties invariant to reparametrization. The following definition makes geometric continuity precise: two regular parametrized segments $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$ meet with nth order geometric continuity G^n if a reparameterization $\tilde{\mathbf{x}}_i$ of $\mathbf{x}_i$ exists such that $\tilde{\mathbf{x}}_i$ and $\mathbf{x}_{i+1}$ meet with C^n continuity [4].

For the running example it is already known that a parametrization exists which is C^1 and therefore it can be concluded that both segments are G^1. The question is, however, whether there exists another reparametrization which is C^n for $n > 1$. This illustrates that the above definition is a good theoretical characterization of geometric continuity, but not suitable for practical analysis. Using this definition requires to consider all possible parametrizations what constitutes a quite laborious process. With the Beta-constraints [4] an equivalent characterization of geometric continuity exists which is better suited for analysis purposes. The constraints for G^2 continuity are for example:

$$\begin{aligned} \mathbf{x}_{i+1}(u_{i+1}) &= \mathbf{x}_i(u_{i+1}) \\ \mathbf{x}_{i+1}'(u_{i+1}) &= \beta_1 \mathbf{x}_i'(u_{i+1}) \\ \mathbf{x}_{i+1}''(u_{i+1}) &= \beta_1^2 \mathbf{x}_i''(u_{i+1}) + \beta_2 \mathbf{x}_i'(u_{i+1}) \end{aligned} \tag{G.8}$$

with the constraints for G^1 take the same form, but omit the last equation. The goal is, however, not to provide a complete formulary for the Beta-constraints, but to clarify that the analysis of geometric continuity relies on the differential properties.

The term *continuity* is now defined precisely, but the same still does not hold for the term

smoothness. From a mathematical point of view a curve, or function, is smooth when it is C^∞. This is, however, not a meaningful notion of smoothness in computer-aided design, since a curve can be C^0 and yet being geometrically smooth as shown in Figure G.2. Based on the discussion on continuity, a natural conclusion is to consider a curve as smooth when it is G^∞, but this would be of little use in practice as many popular curve representations, such as B-spline curves, are not G^∞ and thus would be always considered as non-smooth by definition. The term *smoothness* is defined as follows: a curve is considered as smooth when the segments and their connections are G^n continuous with the minimum required n depends on the application. For some of applications G^1 suffices to be considered as smooth, but in the context of hull form modeling and other engineering applications usually G^2 is required. The term *smoothness* builds on top of *continuity* as the latter is analyzed to conclude smoothness based on the application's requirements.

This page intentionally left blank.

H Knot refinement

Popular subdivision schemes such as Catmull and Clark [23] or Doo and Sabin [28] are a generalization of knot refinement for tensor-product B-splines in the presence of irregular control meshes. The following material provides the necessary background on knot refinement to clarify the derivation of subdivision schemes and to show their role in the context of generalized B-splines. A running example of a uniform biquadratic B-spline surface is used and the employed knot insertion strategy complies with the domain subdivision for generalized B-splines. This example is a reasonable choice because the resulting control mesh refinement is particularly simple and can easily be adopted to an irregular control mesh in order to define a subdivision scheme.

Let the initial setup be a tensor-product B-spline of degree $p = q = 2$ and the uniform knot vectors are

$$U^0 = V^0 = [-2, -1, 0, 1, 2, 3]$$

and the domain is therefore $\Sigma^0 = [0, 1]^2$. The surface is defined on a regular control mesh $\mathbf{Q}^0$ which consists of 3×3 ($m = n = 3$) control points. The superscript of the symbols denotes the refinement level k, with $k = 0$ refers to the initial setup, $k = 1$ to the first refinement, and so on. The initial setup is shown in Figure H.2 on the left side.

A knot refinement results in

$$U^1 = V^1 = [-1, -\frac{1}{2}, 0, \frac{1}{2}, 1, \frac{3}{2}, 2]$$

by introducing a new knot halfway between each existing knot and narrowing down the domain to $\Sigma^1 = \Sigma^0 = [0, 1]^2$. Other choices for the refined knot vector are possible, for instance the knot vector could be turned into the standard form $U = V = [0, 0, 0, 1, 1, 1]$ or any other form as long as the knots are increasing and the minimum length is taken into account.

The particular choice for the knot refinement being used, however, complies with the domain subdivision for generalized B-splines and naturally yields the refinement rules for the control mesh which are the basis to derive subdivision schemes. This first step of knot refinement increases the length of the knot vectors by one and consequently the control mesh $\mathbf{Q}^1$ grows to a size of 4×4 ($n = m = 4$) control points. Note that subsequent refinements will exponentially grow the size of the control mesh because the knot vector's length is generally increased by 2^{k-1} on the basis of this refinement strategy, a characteristic which is well-known from subdivision surfaces.

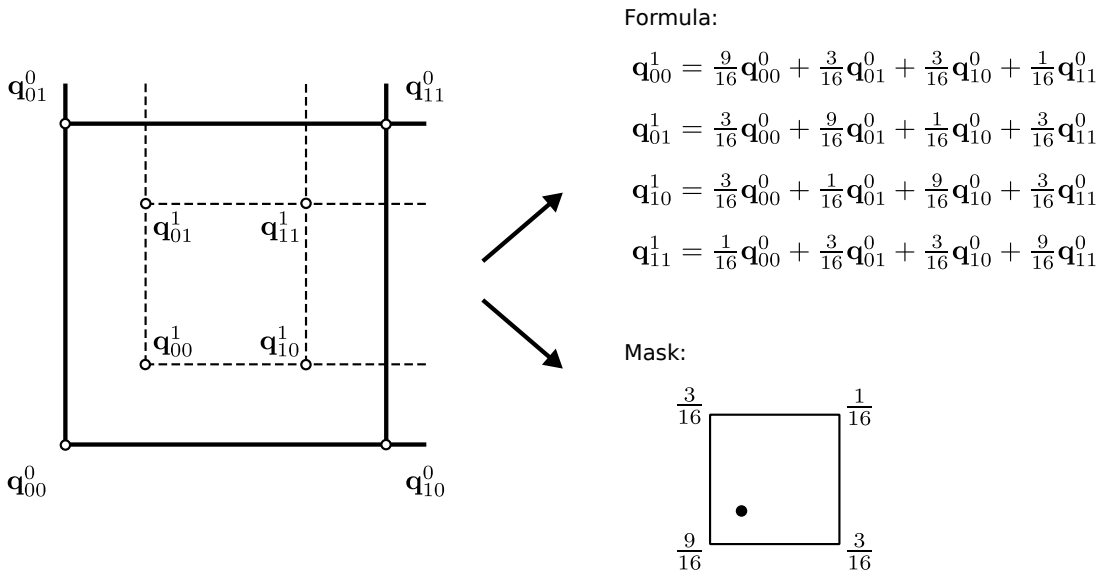


Figure H.1.: The original control mesh $\mathbf{Q}^0$ is shown in bold and the dashed lines illustrate the new control mesh $\mathbf{Q}^1$ For the sake of simplicity, only one face of $\mathbf{Q}^0$ is shown. A closed-form expression for the new points is possible, but depends on the indexing and all points have a similar pattern. An elegant notation are refinement masks, where the particular example requires only a single mask to compute all new control points.

The goal of the knot refinement operation is to compute an equivalent control mesh $\mathbf{Q}^1$ from the original control mesh $\mathbf{Q}^0$ so that the same surface is represented by both control meshes. Spoken mathematically, the goal is to find $\mathbf{Q}^1$ so that

$$\mathbf{x}(u,v) = \sum_{i=0}^{2}\sum_{j=0}^{2} N^0_{i,2}(u)N^0_{j,2}(v)\mathbf{q}^0_{ij} = \sum_{i=0}^{3}\sum_{j=0}^{3} N^1_{i,2}(u)N^1_{j,2}(v)\mathbf{q}^1_{ij}$$

for $(u,v) \in \Sigma$ and $N^0_{i,2}$ and $N^0_{j,2}$ being the basis functions defined on U^0 and V^0, as well as $N^1_{i,2}$ and $N^1_{j,2}$ are the basis functions defined on U^1 and V^1 respectively. A general solution of this problem is given by the Oslo algorithm, see for instance the corresponding material in Rogers [96] or any other reference source on tensor-product B-splines.

For this particular example the solution is illustrated in Figure H.1. The original mesh $\mathbf{Q}^0$ is shown in bold and the new mesh $\mathbf{Q}^1$ is illustrated by the dashed lines. For each control mesh, four new control points $\mathbf{q}^1_{ij}$ are computed based on the original ones $\mathbf{q}^0_{ij}$ defining the face. For the sake of simplicity, only one face of $\mathbf{Q}^0$ is shown, but the other faces are defined similarly.

The Oslo algorithm results in a closed-form expression, one for each new control point, which defines the new points as an affine combination of the old points. The expressions

defining the four control points calculated for the illustrated face are given in Figure H.1. Apparently, all expressions have the same structure and use the same weights to calculate the new control points. This is also the case for the other points which are, for clarity, omitted in Figure H.1. The common pattern of the expression could be described as follow: for each existing face of the control mesh $\mathbf{Q}^0$, compute four new control points, one for each corner of the face, with the nearest control point weighted with $\frac{9}{16}$, the next two control points weighted with $\frac{3}{16}$, and the opposite control point is weighted with $\frac{1}{16}$. A graphical illustration is a refinement mask which is shown in Figure H.1. Since the refinement mask does not depend on the actual indexing due to the symmetry, for the running a example a single refinement mask is sufficient to compute the entire control mesh $\mathbf{Q}^1$. Due to the elegant notation of the rules to compute a refined control mesh, subdivision schemes are usually defined on the basis of refinement masks, referred to as subdivision mask in this context.

The result of the first refinement step is

$$\mathbf{Q}^1 = \begin{bmatrix} \frac{9}{16} & \frac{3}{16} & 0 & \frac{3}{16} & \frac{1}{16} & 0 & 0 & 0 & 0 \\ \frac{3}{16} & \frac{9}{16} & 0 & \frac{1}{16} & \frac{3}{16} & 0 & 0 & 0 & 0 \\ 0 & \frac{9}{16} & \frac{3}{16} & 0 & \frac{3}{16} & \frac{1}{16} & 0 & 0 & 0 \\ 0 & \frac{3}{16} & \frac{9}{16} & 0 & \frac{1}{16} & \frac{3}{16} & 0 & 0 & 0 \\ & & & & \vdots & & & & \end{bmatrix} \begin{bmatrix} \mathbf{q}^0_{00} \\ \mathbf{q}^0_{01} \\ \mathbf{q}^0_{02} \\ \mathbf{q}^0_{10} \\ \vdots \end{bmatrix} = A\mathbf{Q}^0$$

with matrix A is referred to as refinement matrix or, in the context of subdivision surfaces as subdivision matrix. Both, the original control mesh $\mathbf{Q}^0$ and the refined control mesh $\mathbf{Q}^1$, are shown in Figure H.2.

The additional knot in U^1 and V^1 represents an internal subdivision of the domain. For the particular choice of the knot vector, the domain is subdivided into four equal quarters. Successively quartering the domain is the setup from which subdivision surfaces are derived. Accordingly to the domain subdivision, the surface can now be understood as being subdivided into four segments. Each segment is defined by one quarter Σ^1_i and depends only on a subset

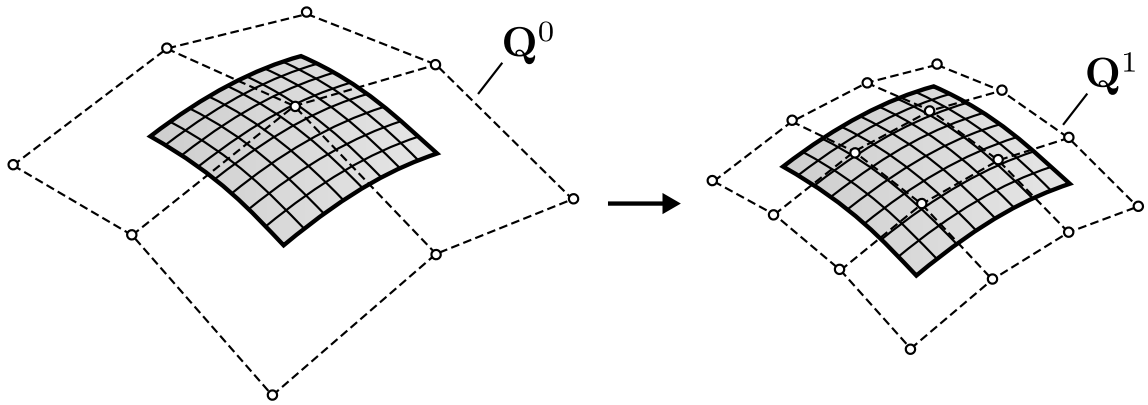


Figure H.2.: Example for the control mesh of a uniform biquadratic tensor-product B-spline surface which is defined on the knot vectors $U^0 = V^0 = [-2, -1, 0, 1, 2, 3]$ and the control mesh $\mathbf{Q}^0$. The knot vectors are refined to $U^1 = V^1 = [-1, -\frac{1}{2}, 0, \frac{1}{2}, 1, \frac{3}{2}, 2]$ and $\mathbf{Q}^1$ is the equivalent control mesh to represent exactly the same surface on the refined knots.

$\mathbf{Q}_i^1$ of the entire control mesh, see

$$\mathbf{x}_0(u,v) = \sum_{i=1}^{3}\sum_{j=0}^{2} N_{i,2}^1(u) N_{j,2}^1(v) \mathbf{q}_{ij}^1 \qquad \text{for } (u,v) \in \Sigma_0^1 = [\frac{1}{2}, 1] \times [0, \frac{1}{2}]$$

$$\mathbf{x}_1(u,v) = \sum_{i=1}^{3}\sum_{j=1}^{3} N_{i,2}^1(u) N_{j,2}^1(v) \mathbf{q}_{ij}^1 \qquad \text{for } (u,v) \in \Sigma_1^1 = [\frac{1}{2}, 1] \times [\frac{1}{2}, 1]$$

$$\mathbf{x}_2(u,v) = \sum_{i=0}^{2}\sum_{j=1}^{3} N_{i,2}^1(u) N_{j,2}^1(v) \mathbf{q}_{ij}^1 \qquad \text{for } (u,v) \in \Sigma_2^1 = [0, \frac{1}{2}] \times [\frac{1}{2}, 1]$$

$$\mathbf{x}_3(u,v) = \sum_{i=0}^{2}\sum_{j=0}^{2} N_{i,2}^1(u) N_{j,2}^1(v) \mathbf{q}_{ij}^1 \qquad \text{for } (u,v) \in \Sigma_3^1 = [0, \frac{1}{2}] \times [0, \frac{1}{2}]$$

with the upper and lower bounds of the sums being adjusted to contain only the relevant subsets of the control mesh and the basis functions are defined on the knot vectors U^1 and V^1. A graphical illustration is given in Figure H.3 which shows the domain subdivision, the segments and the corresponding subset of the control mesh. As the surface remains exactly the same, the four segments are smoothly connected to each other. The segments are defined on overlapping subsets of the control mesh.

Knot refinement is a well-known operation from the tensor-product B-spline theory which does not affect the surface, but changes the size of the control mesh the surface is defined on. Popular subdivision algorithms generalize the particular refinement strategy introduced in this chapter in the presence of irregular control meshes. This is called *subdivision* and the corresponding rules to refine the control mesh are referred to as *subdivision rules* which are usually defined in terms of *subdivision masks*. Subdivision can be encoded as a matrix oper-

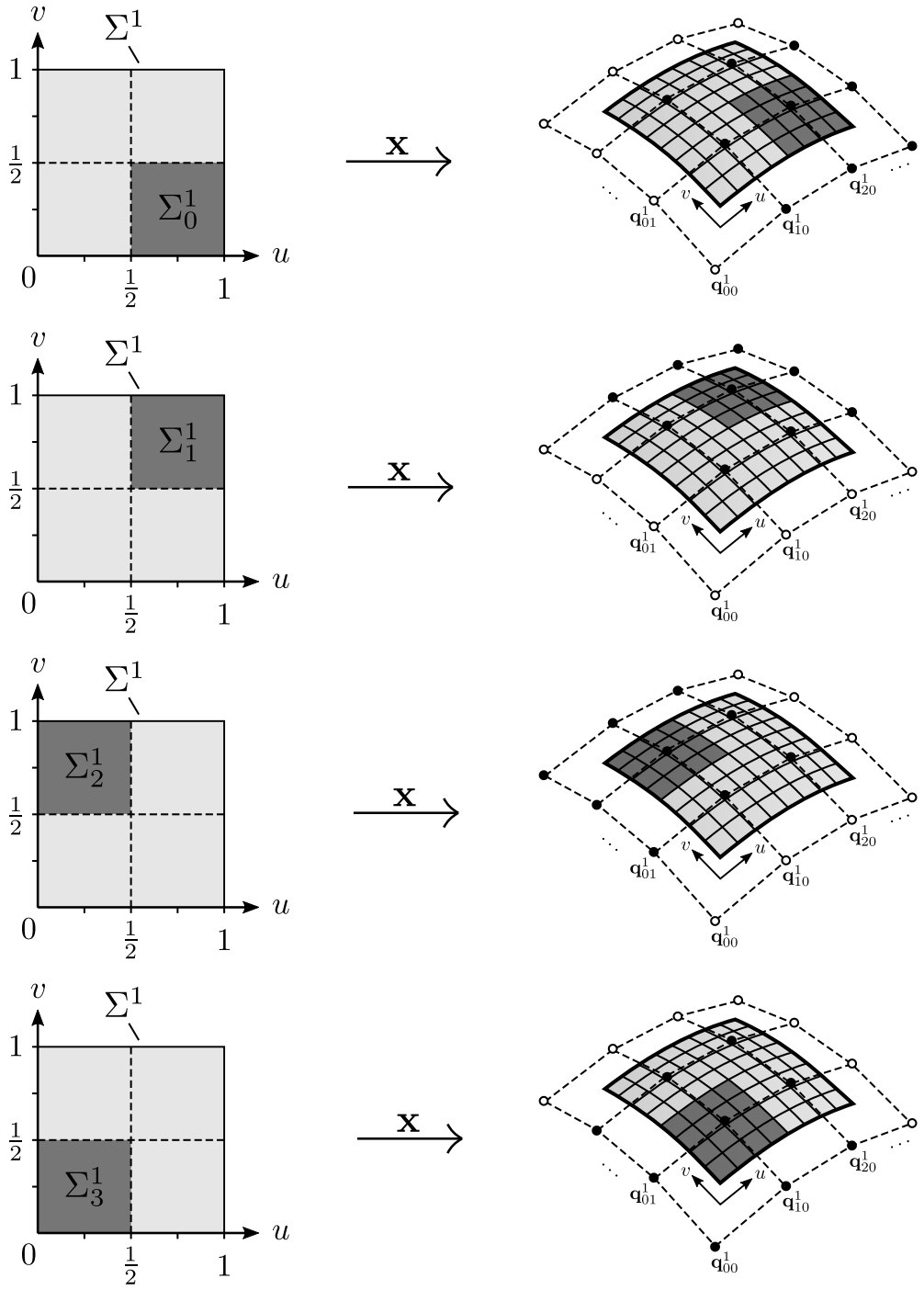


Figure H.3.: The knot refinement represents a subdivision of the domain. For the particular choice of the knot vector, the domain is subdivided into four equal quarters. The surface now can be understood as being composed of four segments each defined over one quarter Σ_i^1 and defined on a subset $\mathbf{Q}_i^1$ of the control mesh which is depicted by the black points.

ation, similar to the refinement matrix given above, and properly formalized this eventually leads to generalized B-spline surfaces.

I Diagonalization

The goal of diagonalization in the context of generalized B-splines is the calculation of arbitrary powers A^k of a matrix $A \in \mathbb{R}^{n \times n}$. In particular, this should allow to calculate the limit of this operation for $k \to \infty$.

Eigenvalues and eigenvectors

A vector $\mathbf{x} \in \mathbb{C}^n$ is an eigenvector to the eigenvalue $\lambda \in \mathbb{C}$ of a matrix $A \in \mathbb{R}^{n \times n}$ if the equation

$$A\mathbf{x} = \lambda\mathbf{x} \tag{I.1}$$

is fulfilled. The trivial solution $\mathbf{x} = \mathbf{0}$ is excluded, but in contrast $\lambda = 0$ is allowed. Usually several solutions exist for Equation I.1

Eigenvalues

The eigenvalues $\lambda \in \mathbb{C}$ of a matrix $A \in \mathbb{R}^{n \times n}$ are the solutions of the characteristic equation

$$\det(A - \lambda I) = 0 \tag{I.2}$$

where the left side of the equation

$$p(\lambda) = \det(A - \lambda I) \tag{I.3}$$

is a polynomial of degree n. The eigenvalues are the roots of $p(\lambda)$. According to the fundamental theorem of algebra, this polynomial can be decomposed in n complex linear factors

$$p(\lambda) = \prod_{i=1}^{i=n} (\lambda - \lambda_i) \tag{I.4}$$

where $\lambda_i \in \mathbb{C}$ are the roots which may not all have distinct values. For a real matrix $A \in \mathbb{R}^{n \times n}$ there are therefore n eigenvalues, where some or even all may be complex. In addition, several eigenvalues can be the same.

Eigenvectors

An eigenvector $\mathbf{x}_i \in \mathbb{C}^n$ to the eigenvalue $\lambda_i \in \mathbb{C}$ of the matrix $A \in \mathbb{R}^{n\times n}$ is the solution of the homogeneous linear equation system

$$(A - \lambda_i I)\mathbf{x}_i = \mathbf{0}. \tag{I.5}$$

Every multiple of $\mathbf{x}_i$ is also an eigenvector to the eigenvalue λ_i. Eigenvectors are usually given in normalized form with $|\mathbf{x}_i| = 1$.

Now several statements about the number of eigenvectors of a real matrix $A \in \mathbb{R}^{n\times n}$ are made. Let $\lambda_i \in \mathbb{C}$ be an eigenvalue of A which has an algebraic multiplicity of k_i. If $k_i = 1$ it is referred to as simple eigenvalue of A, and for $k_i > 1$ it is called a multiple eigenvalue. The number of linearly independent eigenvectors to an eigenvalue λ_i is the defect d_i of the characteristic matrix $A - \lambda_i I$. This is

$$1 \leq d_i = n - r_i \leq k_i \tag{I.6}$$

where n is the dimension and $r_i > 0$ is the rank of the characteristic matrix. The dimension is the same for all eigenvalues, whereas the rank varies for each eigenvalue. Each eigenvalue λ_i has at least one, but not more than k_i linearly independent eigenvectors. In addition, the eigenvectors of different eigenvalues are always linearly independent.

A real matrix $A \in \mathbb{R}^{n\times n}$ therefore has at most n linearly independent eigenvectors, i.e. when exactly k eigenvectors belong to an eigenvalue of multiplicity k. Since this is not always the case, a matrix $A \in \mathbb{R}^{n\times n}$ can also have less than n linearly independent eigenvectors.

Generalized eigenvectors

A vector $\mathbf{v}_i^k$ with $k \in \mathbb{N}$ is referred to as a generalized eigenvector to the eigenvalue $\lambda_i \in \mathbb{C}$ of the matrix $A \in \mathbb{R}^{n\times n}$ if the equation

$$(A - \lambda_i I)^k \mathbf{v}_i^k = \mathbf{0} \tag{I.7}$$

is fulfilled and at the same time

$$(A - \lambda_i I)^{k-1} \mathbf{v}_i^k \neq \mathbf{0}. \tag{I.8}$$

holds. In this context k is called the rank of the generalized eigenvector. As in the case of ordinary eigenvectors $\mathbf{v}_i^k \neq \mathbf{0}$ is required. From $\mathbf{v}_i^k$ are for $1 < l \leq k$ further generalized eigenvectors of lower rank calculated by

$$\mathbf{v}_i^{l-1} = (A - \lambda_i I)\mathbf{v}_i^l. \tag{I.9}$$

This results in a chain of k generalized eigenvectors with the important property that all vectors are linearly independent. In contrast to ordinary eigenvalues, therefore for each eigenvalue λ_i of a matrix A that has a multiplicity of k_i, exactly k_i linearly independent generalized eigenvectors exist. Furthermore, generalized eigenvectors of different eigenvalues are linearly independent. In other words: a real matrix $A \in \mathbb{R}^{n\times n}$ always has exactly n linearly independent generalized eigenvectors vectors.

Eigendecomposition

The eigenvalue problem for a matrix $A \in \mathbb{C}^{n\times n}$ as shown in Equation I.1 is represented in matrix notation to cover all solutions at once as follows. Let

$$X = [\mathbf{x}_0, ..., \mathbf{x}_m] \in \mathbb{C}^{n\times m} \tag{I.10}$$

be a matrix containing the eigenvectors of the matrix A column by column. Then

$$AX = [A\mathbf{x}_0, ..., A\mathbf{x}_m] = [\lambda_0\mathbf{x}_0, ..., \lambda_m\mathbf{x}_m] = X\Lambda \tag{I.11}$$

where the matrix Λ has the following form:

$$\Lambda = \begin{bmatrix} \lambda_0 & & 0 \\ & \ddots & \\ 0 & & \lambda_m \end{bmatrix} \in \mathbb{C}^{m\times m}. \tag{I.12}$$

Suppose the matrix $A \in \mathbb{R}^{n\times n}$ now has exactly n linear independent eigenvectors. Then $X \in \mathbb{C}^{n\times n}$ is an invertible matrix, so that matrix A can be decomposed in

$$A = X\Lambda X^{-1} \tag{I.13}$$

what is referred to as the *eigendecomposition* of A. The advantage of this form is that the calculation of arbitrary powers of A is simplified. From

$$A^k = X\Lambda^k X^{-1} \tag{I.14}$$

follows, that for the eigendecomposition only the potentiation of the diagonal matrix Λ is required. This is particularly simple as only the powers of the elements on the main diagonal

are required, i.e.

$$\Lambda^k = \begin{bmatrix} \lambda_0^k & & 0 \\ & \ddots & \\ 0 & & \lambda_m^k \end{bmatrix}. \tag{I.15}$$

The eigendecomposition of a matrix $A \in \mathbb{R}^{n\times n}$ therefore results in a form of A that allows a simple calculation of the powers A^k. It requires, however, that the matrix A has exactly n linearly independent eigenvectors, which is not always the case according to the above discussion about the eigenstructure of real square matrices.

Jordan decomposition

If a matrix $A \in \mathbb{R}^{n\times n}$ has less than n linearly independent eigenvectors, then the eigendecomposition is not possible. Using the generalized eigenvectors of A, however, a similar form can be found. This is called the Jordan normal form.

Let $\{\mathbf{v}_r^k, \ldots, \mathbf{v}_r^1\}$ be a chain of generalized eigenvectors of length k to the eigenvalue λ_i. According to Equation I.9, the generalized eigenvectors of two successive ranks are related by

$$A\mathbf{v}_r^l = \lambda_i \mathbf{v}_r^l + \mathbf{v}_r^{l-1} \tag{I.16}$$

and for the generalized eigenvector of rank one

$$A\mathbf{v}_r^1 = \lambda_i \mathbf{v}_r^1. \tag{I.17}$$

The matrix notation of both relation is as follows. The matrix

$$V_r = [\mathbf{v}_r^1, ..., \mathbf{v}_r^k] \in \mathbb{C}^{n\times k} \tag{I.18}$$

contains column-wise the generalized eigenvectors of a chain in ascending order. In addition

$$J_r = \begin{bmatrix} \lambda_i & 1 & 0 & \ldots & 0 \\ 0 & \lambda_i & 1 & \ldots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & \lambda_i & 1 \\ 0 & \ldots & 0 & 0 & \lambda_i \end{bmatrix} \in \mathbb{C}^{k\times k} \tag{I.19}$$

is a matrix where the main and secondary diagonal are populated and which is referred to as

a Jordan block. The relations I.16 and I.17 are then column-wise represented by

$$AV_r = V_r J_r. \tag{I.20}$$

This can be summarized for all chains of A in a single matrix equation. Let $\overline{r}$ be the number of chains of A. The generalized eigenvectors are summarized in

$$V = [V_1, \ldots, V_{\overline{r}}] \in \mathbb{C}^{n \times n}. \tag{I.21}$$

This matrix is always quadratic, because each matrix $A \in \mathbb{R}^{n \times n}$ has exactly n generalized eigenvectors $\mathbf{v}_i^k \in \mathbb{C}^n$ and V is consequently quadratic. The Jordan matrix

$$J = \begin{bmatrix} J_1 & & 0 \\ & \ddots & \\ 0 & & J_{\overline{r}} \end{bmatrix} \tag{I.22}$$

is obtained by placing the Jordan blocks J_r on the main diagonal of J. This results in the following equation:

$$AV = VJ. \tag{I.23}$$

Since V is not only quadratic, but all main vectors and thus all columns of V are linearly independent, V is also invertible. This applies without restriction to all matrices and the matrix A can now be decomposed into

$$A = VJV^{-1} \tag{I.24}$$

what is referred to as the *Jordan decomposition*. The resulting form of A is also called the Jordan normal form. With the Jordan normal form any power of A can be calculated similar to the eigendecomposition. The following applies

$$A^k = VJ^kV^{-1} \tag{I.25}$$

where due to the block structure for the Jordan matrix

$$J^k = \begin{bmatrix} J_1^k & & 0 \\ & \ddots & \\ 0 & & J_{\overline{r}}^k \end{bmatrix} \tag{I.26}$$

follows. The powers of the Jordan blocks J_r are explicitly given by

$$J_r^k = \begin{bmatrix} \lambda_i^{m,0} & \lambda_i^{m,1} & \lambda_i^{m,2} & \dots & \lambda_i^{m,l} \\ 0 & \lambda_i^{m,0} & \lambda_i^{m,1} & \dots & \lambda_i^{m,l-1} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \lambda_i^{m,0} & \lambda_i^{m,1} \\ 0 & \dots & 0 & 0 & \lambda_i^{m,0} \end{bmatrix} \tag{I.27}$$

where the elements are

$$\lambda_i^{m,l} = \begin{cases} \binom{m}{k}\lambda^{m-l} & \text{for } \lambda \neq 0 \text{ and } 0 \leq l \leq m \\ 1 & \text{for } \lambda = 0 \text{ and } l = m \\ 0 & \text{otherwise} \end{cases} . \tag{I.28}$$

The Jordan decomposition of a matrix $A \in \mathbb{R}^{n \times n}$ allows, similar to the eigendecomposition, a simple calculation of arbitrary powers A^k. In contrast to the eigendecomposition, the Jordan decomposition can be performed for any matrix A. With the Jordan decomposition a suitable tool for the analysis of arbitrary subdivision matrices is available.