MSc thesis: Computation & Performance

parametric modelling of architectural developables

Roel van de Straat

scientific research mentor: dr. ir. R.M.F. Stouffs
design research mentor: ir. F. Heinzelmann
third mentor: ir. J.L. Coenders

MSc thesis: Computation & Performance

parametric modelling of architectural developables

Roel van de Straat
1041266

Delft, April 2011

Delft University of Technology
Faculty of Architecture

preface

The idea of deriving analytical and structural information from geometrical complex design with relative simple design tools was one that was at the base of defining the research question during the early phases of the graduation period, starting in September of 2009. Ultimately, the research focussed an approach actually reversely to this initial idea by concentrating on using analytical and structural logic to inform the design process with the aid of digital design tools.

Generally, defining architectural characteristics with an analytical approach is of increasing interest and importance with the emergence of more complex shapes in the building industry. This also means that embedding structural, manufacturing and construction aspects early on in the design process is of interest. This interest largely relates to notions of surface rationalisation and a design approach with which initial design sketches can be transferred to rationalised designs which focus on a strong integration with manufacturability and constructability.

In order to exemplify this, the design of the Chesa Futura in Sankt Moritz, Switzerland by Foster and Partners is discussed. From the initial design sketch, there were many possible approaches for surfacing techniques defining the seemingly freeform design. The key to controlling the form was to make use of a polar grid association. Generally, the polar grid is a well applicable way of locating elements, such as windows, whose positions are based on a radial geometry. The geometric definition was based on four sectors and a number of subdivisions within each sector. This provided flexibility and control as well as a convenient coding and referencing system[1] and additionally, based on this definition, parametric relations were defined, providing smooth transitions between the sections, Figure P1.

The example of the Chesa Futura project shows how a logic of radial geometry within a parametric environment is used to generate plans and sections and as such defines a globally rationalised surface definition.

[1] Lenz, Chesa Futura

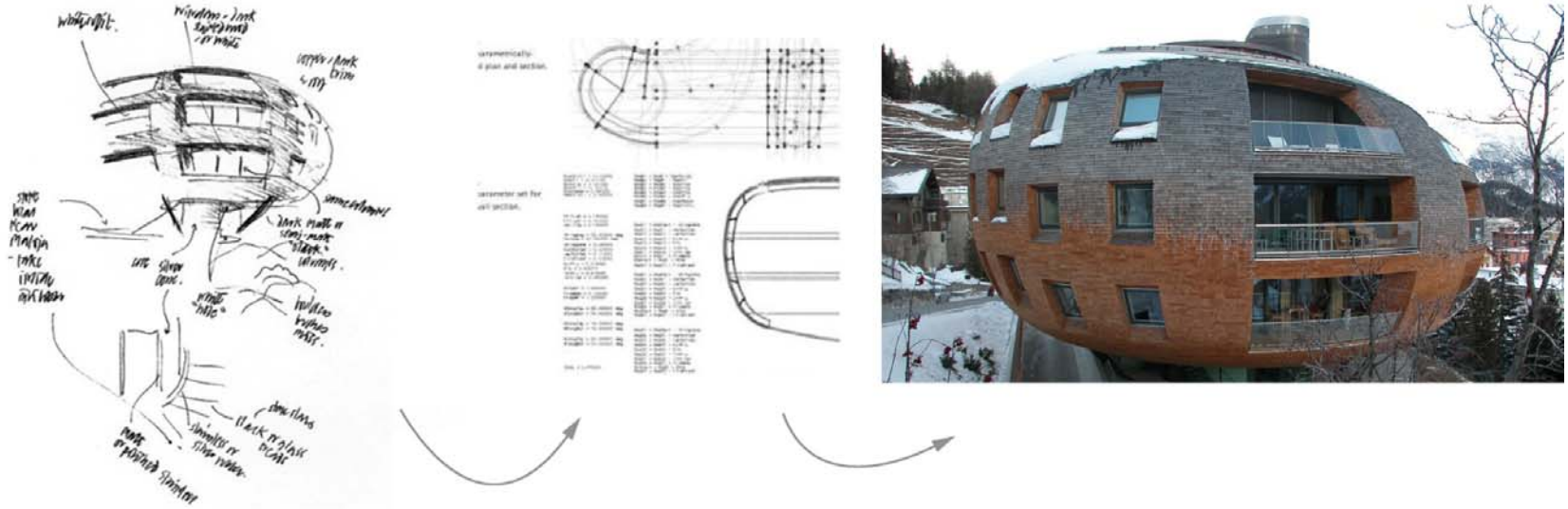


Fig. P1. From sketch design to geometric description based on parametric radial rationalisation of the Chesa Futura in Sankt Moritz, Switzerland by Foster and Partners [http://www.daapspace.daap.uc.edu, March 2011]

This thesis focuses on discussions related to definitions of digital rationalised surface geometry. As such, it is an integral part of the graduation project of the research group Computation & Performance of the department of Building Technology of the Delft University of Technology, Faculty of Architecture.

The thesis describes the research and background studies in rational design methods and parametric descriptions of surface definitions and describes a design approach based on digital design tools which allow for the analysis and generation of developable surfaces. The .gha assembly which allow access to these tools in Grasshopper for Rhinoceros is provided including a installation manual, see Appendix C.

The content of the research is presented on 8 April 2011.

acknowledgements

A number of people took time and effort to assist in working on this thesis. It is underlined here that their input is much appreciated and they are thanked accordingly.

The members of the graduation committee in general are thanked for their advice and feedback given during the process of writing this thesis and being available for questions related to the research. In particular, **dr. ir. R.M.F. Stouffs**, for his contribution to the Computation & Performance research group which provides for research topics as discussed in this thesis at the Faculty of Architecture of the Delft University of Technology and his insightful and valuable comments regarding the scientific approach taken in this research. **Ir. F. Heinzelmann** for sharing his insights from the architectural practice and for the fruitful discussions on digital design, complex geometry and utilisation of design tools in the design process. **Ir. J.C. Coenders** for his sharp and to the point feedback on a technical and scientific level and for sharing his knowledge on tool development related to, on one hand, the programming aspects, and on the other hand, deployment and implementation.

Colleagues at Arup Amsterdam are thanked for their positive feedback and their interest in the progress of the thesis. Especially, **Anke Rolvink** for assisting in setting up the code for the development of the Grasshopper components and helping with issues related to programming the tools. **Joost Lauppe**, who showed a great enthusiasm and interest regarding the development of the tools and the content of the research and was a great support on a personal level throughout the graduation period.

summary

general introduction

Over the last decades, the advances in digital surface modelling seemingly grew ahead of possibilities to follow the design intention in analysis and construction. This especially in relation to the control over the geometry and the costs of fabrication and assembly of non-standard elements. The lack of control over the geometry, in relation to both structural analysis and construction, in some cases proves to be a hurdle in realising a design. And although the current state of technology allows computation-driven design processes to incorporate tools and machines that automatically fabricate structural elements, formwork and building components, the costs of employing digital fabrication are not always exceeded by the profit. As such, geometrically complex design can put the relationship between design and realisation under pressure when manufacturing and construction methods cannot follow the envisioned design or when they result in a disproportionate increase of costs.

One general aspect in reducing the complexity of translation from modelling to construction is the rationalisation of geometry informed by manufacturing and construction characteristics. Two world renowned structures have been used to present notions of form and surface rationalisation. Firstly, the design process of the Sydney Opera House shows how the initial designed forms needed to be rationalised into spherical elements simply to be describable using contemporary drafting and engineering methods and to positively affect the constructability of the structure. Secondly, the Guggenheim Museum of Bilbao is presented as an exemplary project for the design of architectural surface geometry based on the principles of single-curved surfaces, also known as developable surfaces. Related to the surface descriptions of these projects, it is exemplified that rationalisation can greatly simplify the surface definition of a design, positively affecting the engineering and construction of a design. This, ideally, whilst still allowing for a vast design freedom.

Based on the above, this thesis tries to provide an answer to the question of how restrictive design conditions of rational surfaces be taken into account within a parametric environment to inform the design process which is focussed on the translation from design to realisation. As such, the goal is to embed the use of digital tools in a design process which on one hand deals with the boundary conditions related to rational surface description and on the other hand provides the designer possibilities in choosing parameters allowing for a design freedom within these restrictions.

rationalisation of surface definitions

Along with the digital possibilities in fabrication and construction, focusing on decreasing the complexity of the design by rationalising surface definitions can narrow the gap between digital modelling, fabrication and construction. Shaping the geometry based on rational surface classes, such as developable surfaces, allows designers to influence the buildability along with positively affecting the level of complexity in structural analysis. Shelden denotes that rationalisation serves as the resolution of rules of constructability into project geometry[2]. In this sense, rationalisation may allow for full and precise control over the structural dimensions, may avoid having to deal with limitations in CAD/CAM machinery and related software and may force a design to be constructed out of elements from a limited number of moulds.

parametric definitions of developable surfaces

In the field of surface rationalisation, developable surfaces play a specific role as they allow for expressing an overall sculptural appearance in R^3 space while being conform definitions of single-curvature geometries. In the words of Huffman: “Developable surfaces offer a complexity that is midway between a completely general surface and a plane surface”.

Developable surfaces have the advantage that they can be made from flat surface material, such as metal sheeting. It follows from this and the definitions above that on one hand, developable surfaces are more difficult to generate than standard Euclidean planar shapes and are also constrained compared to freeform surfaces. On the other hand, their strict definition allows them to be defined parametrically based on either theoretical formulas or on vector mathematical operations related to their structure of surface generators. As such, balancing between complexity and simplicity, rationalising by means of parametrically defining developable surfaces may provide a strong asset to the designer taking constructability into account and possibly making complex shapes and structures more cost efficient.

The surface generators which define the geometry of developable surfaces are generally denoted as rulings. As such, being a subset of the ruled surface class, the parametric representation of a developable surface is equal to that of a ruled surface

$$\mathbf{S}(u,v) = \mathbf{C}(u) + v\mathbf{a}(u)$$

However, the additional restraint for a ruled surface to be developable is denoted by

$$\dot{\mathbf{C}}(u) \cdot (\mathbf{a}(u) \times \dot{\mathbf{a}}(u)) = 0$$

[2] Shelden, Digital surface representation and the constructability of Gehry’s architecture

As architectural complex geometric definitions are generally not based on differentiable parametric equations, this research shifted its focus from a theoretical approach to a discrete differential geometry approach; approximating the fundamental properties, clarifying the structure of the smooth geometry. Whereas global differential geometry studies the influence of the local properties on the behaviour of the entire surface, a discrete approach relates to an extrinsically perspective, regarding the embedding in Euclidian space. As such, the combining of general intrinsic differential methods with discrete geometrical operations forms the base of the digital generation of developables as it provides for the generation of discretely defined rulings. Subsequently, the rulings are used as input information to generate smooth developable surfaces, which are in essence approximations of surfaces based on differential geometry. Where the limit goes to a zero distance between subsequent discrete rulings, the approximation is equivalent to the surface based on differential geometry. As such, definitions to define developables surfaces are relatively simple as the condition for developability can be geometrically described by regarding the rulings of the surface and their individual set of normal vectors which need to be parallel.

tool development and implementation

The design approach of parametric modelling of developable surfaces proposed in this research is deployed via a digital toolbox developed for Grasshopper, a parametric plug-in of the 3D modelling software Rhinoceros. The toolbox contains various algorithmic methods categorised in three toolbox 'compartments'. The algorithmic methods, or tools, allow for surface analysis and modelling and support the designer in the search for geometrically rationalised surfaces which answer to the initial ideas of the design. As such, the toolbox provides 'hints' to guide the generation of surfaces towards the constructible geometry of developable surfaces. This holds that the tools may suggest alterations of the design input and do generate surfaces based on the input, however they cannot be categorised as design tools. This in a sense that the toolbox provides the outcome of a predefined logic within a restricted workflow and therefore lacks the essentials of a set of design tools, which allow for a more creative process.

The development of the architectural developable toolbox tried to follow the Grasshopper framework setup in providing relatively simple components which perform basic operations. As such, the user is provided a new set of components which allow for a similar approach in utilisation as the standard Grasshopper components. Firstly, the input parameters are limited in number and generally do not require specific definitions or operations prior to providing data. Secondly, the surface definition components output is primarily focussed on two types of data; 1) the lofted surface defining a developable surface and 2) the rulings on which the developable surface is based on.

concluding remarks

This thesis presents the implementation of a rational design approach, which is exemplified by deploying a digital toolbox of components, supporting the design process focussed on analysing and generating developable surfaces and demonstrates how the restrictive design conditions of developable surfaces can be taken into account within a parametric environment. As such, it is exemplified how parametric rational designing provides for an active attitude towards creating geometric surface constructions which are based on a set of principles generated for the purpose of increasing manufacturability and constructability.

table of contents

part01 – surface rationalisation

Image on previous page:
Walt Disney Concert Hall [Nick P, http://www.flickr.com, December 2010]

1 introduction and project description

definitions of complexity in architecture

The digital driven building typology of the last decades has not received a name on which everyone – in various disciplines – agrees. By some, b*lob architecture* is considered an outdated terminology and using the term *free form architecture* is often in direct conflict with the defined rules on which digital designs are based. Defining the descriptive geometric design as complex might be dependent on skills or on discipline which makes the meaning of complex geometry a suggestive one. For instance, modelling a perfect sphere is no problem, building it is a totally different issue. Without directly advocating for the use of the terminology of *complex architecture*, this however does point out the quintessence of this research. The distinction whether a geometric description is complex or not is particularly important from an analysis and construction technology viewpoint in relation with the digital means of modelling this geometry.

digital fabrication and rationalisation of design

In building design, there is a general relationship among the development of tools for representing design information, the development of technologies for making complexly shaped members and the specific kinds of architectural design that were enabled by the relationships that existed at a specific point in time[3]. However, over the last decades, this relationship is put under pressure as the advances in digital modelling in some instances have surpassed the possibilities to follow the design intention in analysis and construction. This may lead to geometry which, although digitally specified, is difficult to set out on site and may result in a disproportionate increase of the costs of fabrication and assembly of non-standard elements. This is visually expressed by the diagram in Figure 1.1, abstractly denoting the discrepancy between the possible level of geometric complexity in architectural design and the possibilities in handling geometric complexity in construction.

However, after the first generation of digital design processes employing digital design tools for the development of new forms and relationships, new processes were and are emerging. Instead of modelling complex – here meaning sometimes nearly unbuildable – shapes, new design tools were developed that calibrate the digital with reality on the construction site. Parallel to this, digital fabrication tools were added to the design process. This does, however, require new approaches in design of buildings. As Lisa Iwamoto states: “as with all tools of production, the very techniques that open the investigations to new methods of production have their own sets of constraints and gear particular ways of working”[4].

[3] Schodek, Digital design and manufacturing: CAD/CAM applications in architecture and design

[4] Iwamoto, Digital fabrications – Architectural and material techniques

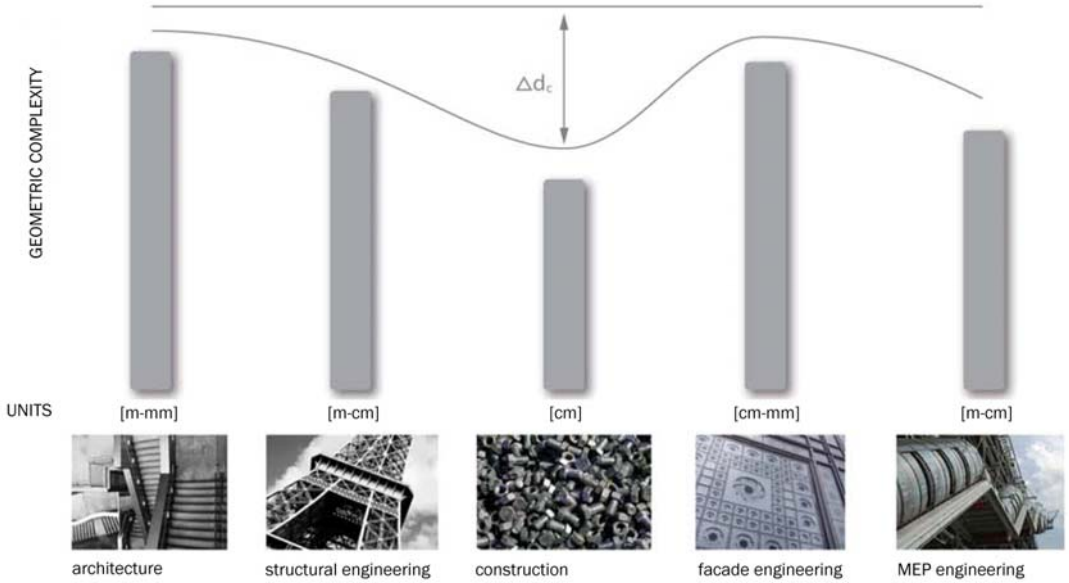


Fig. 1.1. Capabilities in geometric complexity of various disciplines: architectural design, structural design, construction, façade engineering and MEP engineering: current situation with large discrepancies in dealing with geometric complexity.

Nonetheless, the current state of technology allows computationally driven design processes to incorporate tools and machines that automatically fabricate structural elements, formwork and building components, if not now, then in the near future, Figure 1.2. However, currently, costs of employing digital fabrication are not always exceeded by the profits, since the relation between digital design and manufacturing and construction has not yet come to a mature growth. As Schodek puts it: "the marriage between [...] digital design and engineering environments [...] and sophisticated numerically controlled production machines [...] has yet to be culminated"[5]. Additionally, the lack of control over the geometry, in relation to structural analysis, may prove to be a big hurdle in realising a design.

[5] Schodek, Digital design and manufacturing: CAD/CAM applications in architecture and design

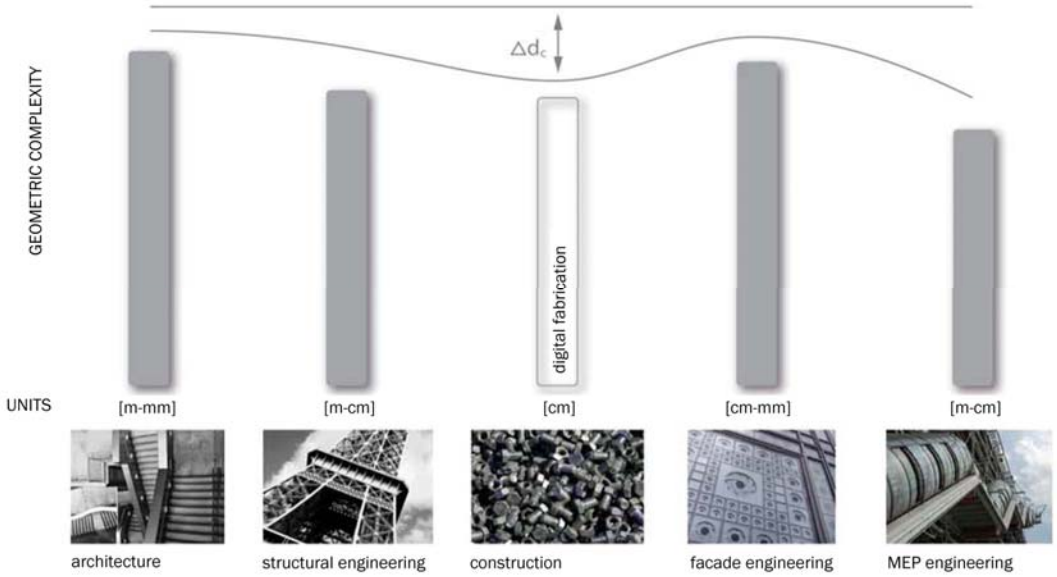


Fig. 1.2. Capabilities in geometric complexity of various disciplines: architectural design, structural design, construction, façade engineering and MEP engineering: increasing deployment of digital developments in construction

Although the building industry is catching up with the deployment of numerically controlled machines and CAM software, currently, discrepancies between disciplines exist in the capabilities of handling complex geometry. Therefore, along with the digital possibilities in fabrication and construction, focusing on decreasing the complexity of the design by rationalising surface definitions can narrow the gap between digital modelling and fabrication and construction. Shaping the geometry based on for instance rational surface classes such as developable surfaces, allows designers to greatly influence the buildability along with positively affecting the level of complexity in structural analysis. As such, the discrepancies in various disciplines in handling complex geometry might be partially resolved by diminishing the complexity by rationalising the geometry, Figure 1.3.

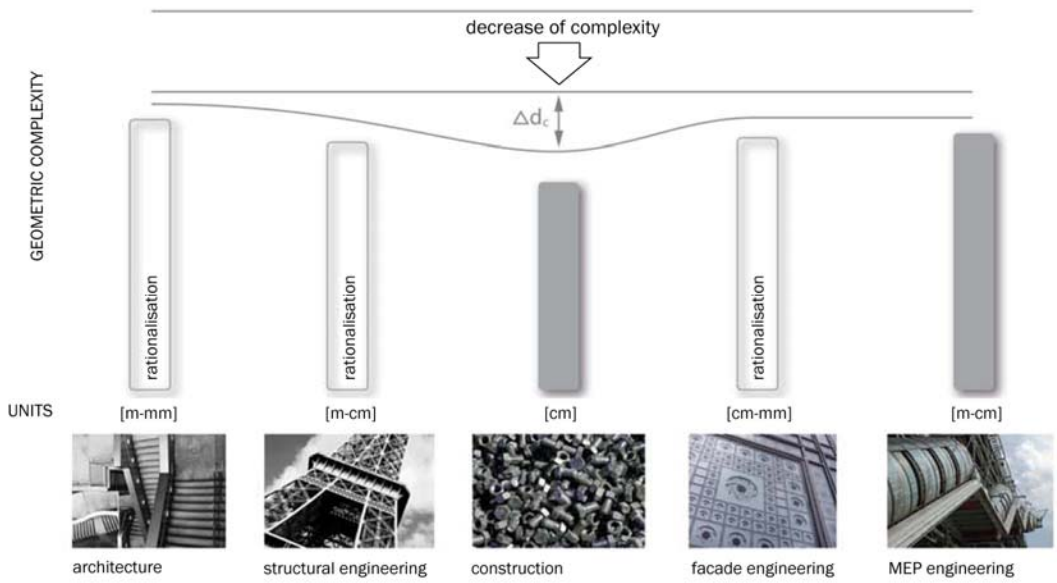


Fig. 1.3. Capabilities in geometric complexity of various disciplines: architectural design, structural design, construction, façade engineering and MEP engineering: decrease of geometric complexity by rationalisation of design

formulation of the objectives of the MSc thesis

Based on the discussion above, one of the main questions of the research is how can rationalisation strategies, which incorporate notions of structural design, manufacturability and constructability, inform a design process which is focussed on the translation from design to realisation. One such strategy is related to design based on a subset of rationalised surface classes, comprising for instance surfaces of revolution, ruled surfaces and translational surfaces, which posses different positive characteristics related to surface definitions for engineering and construction. As such, an initial objective is to present the advantages and shortcomings in geometric modelling of these surface types and their relation with construction principles and to discuss their architectural qualities and the accompanying descriptive freedom. To already exemplify this, the Chesa Futura project, described in the preface, shows how a surface of revolution positively improves control over the form, however, the resulting double curved surface poses additional difficulties in construction.

In this research, it is suggested that a design approach incorporating pre-rationalised surface definitions may be well implemented in a (digital) design process when custom developed tools related to these surface definitions are available to the designer. This especially, when these tools allow for the definition of the logic and restrictions defined by the surface definitions on one hand, and for a certain freedom by denoting design parameters on the other hand. As such, the general objective of this research is to present a set of digital parametric tools that can be employed in a rational design approach with which initial design ideas can be transferred to rationalised designs with the focus on utilising the characteristics of rational surface classes.

In order to present a coherent set of such tools related to a single type of the rational surface classes, the research in the rational surface classes converges to an in-depth research of the description of ruled surfaces and specifically on one of its subsets called developable surfaces. In order to support the suggested design approach based on pre-rationalisation, the development of the toolbox containing methods focussing on analysis and modelling of developable surfaces serves as a proof of concept rather than the exposition of fully functional and general applicable tools. Nonetheless, describing the individual functionality of the digital tools forms an important part in this thesis.

contents of the report

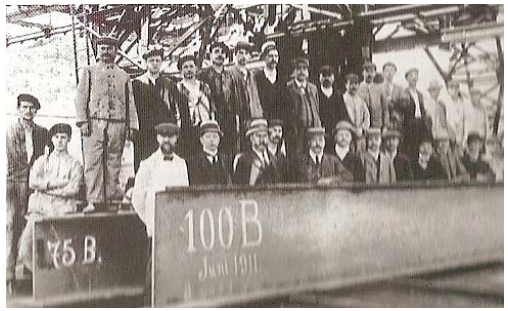
This thesis summarises the background research firstly by explaining the notion of rationalisation on the basis of two well-known precedents, secondly by describing the ways of presenting rational surface classes and polyhedral surfaces in architecture and their mathematical background. Chapter 2 focuses on the presence of rationalisation and rationalised surface geometry in architecture and setting boundary conditions for the implementation of rationalised surfaces in design. Chapter 3 deals with the general mathematical descriptions of rational surfaces and their metrical properties. Subsequently, Chapter 4 and Chapter 5 present the results of an in-depth research in developable surfaces, which are exemplary for potentially positive results of using a rational design approach. Design methodologies of developables in general and the design process which incorporates custom developed parametric components are described in Chapters 6 and 7. These chapters, therefore, focus mainly on the functionality and usability of a parametric rationalisation design approach for developable surfaces, which lies at the core of this research. In the final chapter, the research results will be discussed and the outcome will be concluded with concluding remarks and a reflection.

2 rationalised surface constructions for architectural purposes

Basically, rationalisation in the building industry comes in various forms. One basic example is rationalising through standardising to limit complexity and cost of manufacturing. Steel I-profile beams, for instance, generally follow certain predefined – or rationalised – rules on which the design is based, Figure 2.1.

This research is focused on rational design in the sense of incorporating knowledge and boundary conditions from engineering and construction in architectural design. On one hand, rationalisation can be seen as dealing with the solving of design problems in relation to construction in a later stage of the design process, or in other words, post rationalisation. On the other hand, designs can be pre-rationalised, where an active attitude towards geometric design is employed in which design characteristics are based on a set of principles generated for the purpose of increasing manufacturability and constructability. Figure 2.2 shows the abstract directions of these two forms of rationalisation in the design process.

Two world renowned examples are discussed in Section 2.1 in order to exemplify methods of rationalisation. Subsequently, Section 2.2 presents a concise overview of generic modelling and construction principles of surface design. Section 2.3 deals with the architectural and structural implications of extruded surfaces, surfaces of revolution, ruled surfaces and translational meshes, instances that fall under the common denominator of rational surface classes.



Designation	Dimensions						Dimensions for detailing					Surface	
	G kg/m	h mm	b mm	t_w mm	t_f mm	r mm	A cm^2	h_1 mm	d mm	ø	P_{min} mm	P_{max} mm	A_L m^2/m	A_G m^2/t
HE 700 AA*	116	670	300	13	17	27	150.9	636	562	M 27	122	198	2.448	14.46
HE 700 A	204	690	300	14.5	27	27	260.5	636	562	M 27	134	198	2.585	12.75
HE 700 B	241	700	300	17	32	27	306.4	636	562	M 27	136	198	2.630	10.48
HE 700 A	301	716	304	21	40	27	383.0	636	562	M 27	132	200	2.640	8.513
HE 700 x 353*	353	728	308	25	46	27	448.6	636	562	M 27	138	200	2.597	7.359
HE 700 x 414*	414	744	312	29.5	54	27	527.7	636	562	M 27	142	204	2.625	6.338
HE 800 AA*	175	770	300	14	19	30	218.5	734	674	M 27	130	198	2.640	15.01
HE 800 A	224	790	300	15	28	30	285.8	734	674	M 27	138	198	2.689	12.02
HE 800 B	262	800	300	17.5	33	30	334.2	734	674	M 27	134	198	2.713	10.34
HE 800 M	317	814	302	21	40	30	404.3	734	674	M 27	136	198	2.746	8.655
HE 800 x 373*	373	826	308	25	46	30	474.6	734	674	M 27	144	200	2.782	7.469
HE 800 x 434*	434	842	312	30	54	30	544.0	734	674	M 27	148	204	2.824	6.507

Fig. 2.1. Top: The first HE1000B at the Differdange rolling mill, Luxembourg [image courtesy of Arbed Luxembourg]
Bottom: Profile data for various HE profiles [http://www.sections.arcelor.com, January 2010]

2.1 introduction: precedents in rationalised architecture

form rationalisation: Sydney Opera House – Jørn Utzon

Initially, no definite geometry for the shells of the Sydney Opera House had been established and the geometric sketch design was being described by Ove Arup as follows: "the structure has a somewhat unusual shape, which was determined more by architectural than by structural considerations[6]."However, finally it has been designed and constructed within the constraints of a strong geometric discipline which dominates the form of the Opera House structure: four shells, made up of two half-shells symmetrical about the central axis of each hall they cover. The form rationalisation proved to be the solution to making the building buildable. This was made possible by rethinking the problems that had arisen with the initial geometric design[7]. As a result, the solution was depended on the new rationalised geometric definition and the inherent description of how to construct the shells.

[6] Arup, The evolution and design of the concourse at the Sydney Opera House

[7] Sydney Opera House Trust, Sydney Opera House, Utzon design principles

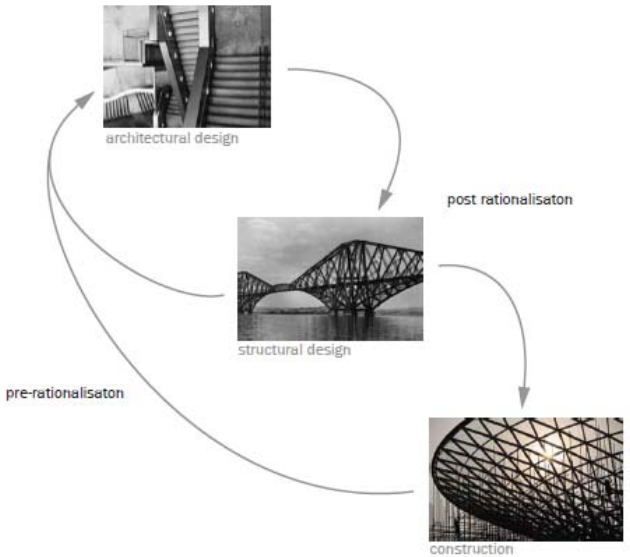


Fig. 2.2. Pre-rationalisation and post rationalisation in architecture, engineering and construction

The complexity of Utzon's concept of sketched curved shell surfaces, see Figure 2.3, was simplified through a seven-year design development phase. From 1957 until 1963, twelve shell solutions were considered, with each successive form becoming more rationalised and structural elements more repetitive. Ultimately, the complex curved surfaces were abandoned in favour of describing the shells with a standardised curvature of a sphere, defining sail-shaped sections from a 75 metre diameter sphere with a post-tensioned precast concrete rib structure[8].

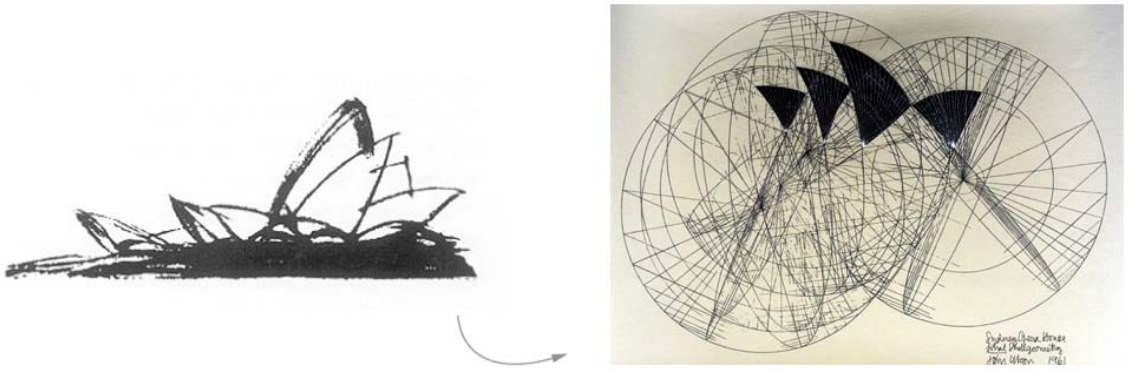


Fig. 2.3. From sketch design to geometric description of the shell structures [http://www.visit.heritage.nsw.gov.au, November 2008] and [http://www.utzonoperahouse.com, November 2009]

This lead to a common denominator; the same spherical surface to deal with, with a similar curvature throughout. This made it possible to standardise formwork and to precast the concrete-shells in smaller pieces and assemble these pieces on location.

Also the glass enclosures, see Figure 2.4, right, are compositions derived from basic geometric elements; a cylinder intersecting an upper cone and a lower cone. These enclosures were described in a series of faceted planes, with intersection points fully described in three axes displacements, with individual panels identified with corner points. By adjusting key displacement points within the glass wall, variations of geometry could be explored via subroutines to calculate the coordinates of all key structural glass surfaces[9].

[8] Schodek, Digital design and manufacturing: CAD/CAM applications in architecture and design

[9] ibid.

Although the roof geometry was substantially standardised, the large variation of the roof tile configuration needed a rationalisation method as well. Ultimately, six standard tiles were highly repetitive, while the tiles on the panel edges changed dependent on their location, Figure 2.4. The panel sizes were limited to 19.5 square metre for ease of handling. Within each chevron shaped panel, the square tiles were identical, with edge tiles custom-cut to fit each edge condition[10].

Fig. 2.4. Precast spherical element principles of the tiles and with the glass enclosures on the right[11]

In conclusion, of the resolutions to make this project buildable, the major one was the implementation of the structural rationalisation which led to a new geometric description of roof for the concert halls. Utzon's proposal was a shell structure, but generated problems because from his sketches it was impossible to calculate a regular geometry for it[12]. Based on the post rationalisation principles, the building ultimately proved to be buildable, although blowing off the project or severely altering the design was not far off during various stages.

surface rationalisation: Guggenheim Museum Bilbao – Gehry Partners

When Frank Gehry won the competition to design the Guggenheim Museum in Bilbao, Spain in 1991, his office was in the midst of changing from a traditional practice to a digitally adapted one. For his projects, Gehry wanted better control of the entire process from design through construction. The winning

[10] Schodek, Digital design and manufacturing: CAD/CAM applications in architecture and design

[11] Sydney Opera House Trust, Sydney Opera House, Utzon design principles

[12] Szalapaj, Contemporary architecture and the digital design process

competition model was accompanied with watercoloured rendered plan, section and elevation drawings that were made using traditional methods, as was most of the early design development for the project. As a result, most of the early development of the museum's complex three-dimensional form was tediously drawn by hand. It was not until the final design model was complete that the computer was extensively used[13]. Connecting the sketches with the computational models, Figure 2.5, Gehry made use of paper models.

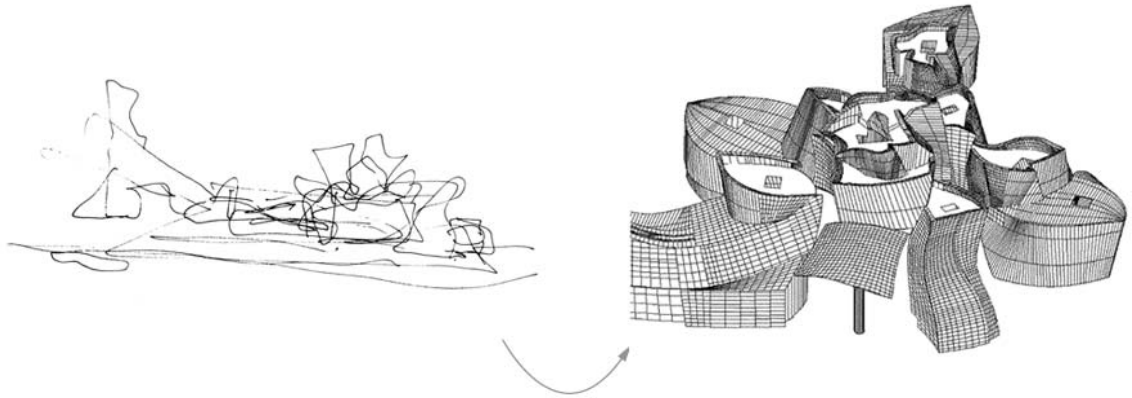


Fig. 2.5. From sketch design to digital model [http://images.allmoviephoto.com,May 2010] and [http://www.arch.mcgill.ca, May 2010]

As is usually the case in design, this process was iterative and nonlinear. The physical models were reverse-engineered using a digitiser to take coordinates off the paper model's surface and to import it into a 3D digital environment. The design subsequently moved back and forth between physical and digital surface models – physical models for aesthetics, digital models for 'system fit'[14].

[13] Lindsey, Digital Gehry – Material resistance, digital construction

[14] Iwamoto, Digital fabrications – Architectural and material techniques

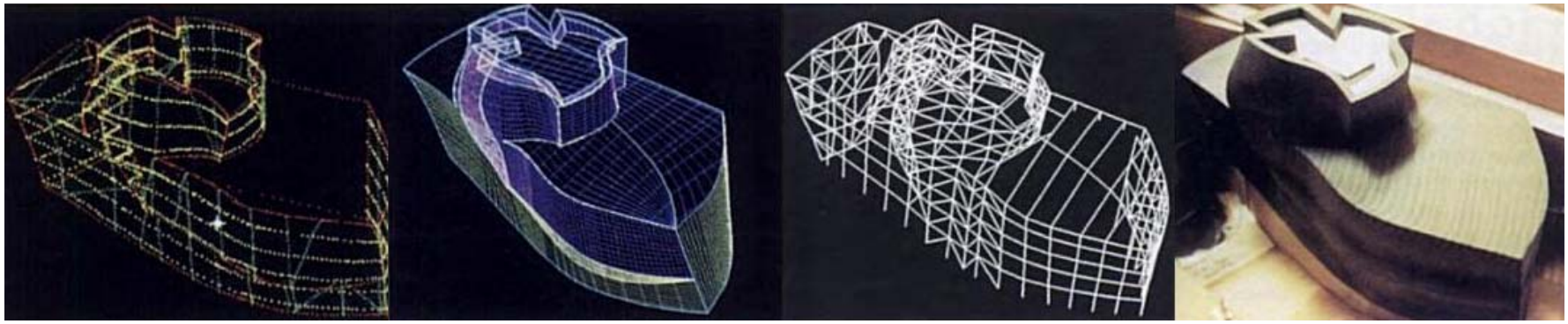


Fig. 2.6. Digitising sequence of the Guggenheim Museum; a series of points are generated by scanning physical models and are subsequently developed into surfaces. A wireframe model is extruded and a check model is milled to ensure accuracy[15]

In the process of reverse engineering, a pattern of points, called a point cloud, is distilled from a scanned physical model and is then interpreted by conversion software to produce a close approximation of the model's geometry, Figure 2.6. Typically, the patterns of scanned points are used to generate surface profile NURBS curves, which are then used to generate lofted NURBS surfaces[16].

The digital model became the dimensional reference that allowed the working drawings to be developed and later helped to coordinate the construction of the project. Instead of a 'simple' loft of the generated curve profiles, the architecture of the museum is based on the use of developable surfaces for form finding and the efficient cladding of substructures (more on developable surfaces in the subsequent chapters).
Gehry's office used the Gaussian analysis to determine the areas of excessive curvature, as there are limits as to how much the sheets of metal could be bent in one direction. This analysis was also used in the Experience Music Project in Seattle to determine which of the apparently double curved surface patches could be converted into developable ones and which ones need to be complexly shaped[17]. And although no two elements of the structure for the 24.000 square metre building are the same, the rationalisation step of creating developable surfaces entailed that all of the titanium cladding panels were supplied flat and only four panel sizes were used for cladding 80 percent of the surface.

[15] Lindsey, Digital Gehry – Material resistance, digital construction
[16] Kolarevic, Architecture in the Digital Age – Design and Manufacturing
[17] ibid.

comparing the Sydney Opera House and the Guggenheim Museum of Bilbao

In 'Digital design and manufacturing', Schodek writes: "the Sydney Opera House illustrates pioneering applications of computer aided geometric design for manufacturing in architecture. Digital modelling facilitated a detailed layout of building assemblies and many digital component models of concrete ribs, precast roof tiles, glass panels and steel structures that were generated to produce full-size patterns for fabrication[18]". Kolarevic calls the Guggenheim Museum probably the best known example that captures the zeitgeist of the digital information revolution [...] since it challenges not only how buildings are designed, but also how they are manufactured and constructed[19]. Clearly, the influence of both presented examples of rationalised structures on the building practice is great. Nonetheless, the process of realisation of the design and the structure itself differs to large extends. This can be summarised by two important characteristics of an architectural design process.

The first is the management of the design process itself. There are many examples of good architecture that arose out of unmanaged projects and even out of badly managed projects. The Sydney Opera House is one of them. Frank Gehry, however, has a reputation for building on time and within budget, even for large, complex and innovative structures. In comparison with the Sydney Opera House, point by point the approach was exactly opposite. In Sydney, the original budget of seven million Australian dollars was not a real but a political budget. The Labour government of New South Wales, the main proponent of the Opera House, wanted the project approved and construction started before elections in March 1959 and before either drawings or funds were fully available. If one principal cause can be identified for the troubles and the cost overrun of 1400% of the opera building, this is it[20]. According to Gehry, continuing relationships with the individual building trades is an important ingredient in keeping within budgets, especially since various designs by Gehry would not have been possible without the local steel and shipbuilding industry[21], and the political and business interests must be kept at arm's length from the design process.

The second design characteristic in which the Opera House and the Guggenheim differ, and more related to the theme of this research, is the employment of rationalisation of the surface geometry in relation with the computational modelling of these surfaces. In the late 1950s and early 1960s, the design team of the Opera House did not have access to advanced design technology and computing power which was available to Gehry around the turn of the century. As described above, consequently, Utzon and Arup had great difficulties finding a practical way of building the curved concrete shells. The years of experimentation

[18] Schodek, Digital design and manufacturing: CAD/CAM applications in architecture and design

[19] Kolarevic, Architecture in the Digital Age – Design and Manufacturing

[20] Flyvbjerg, Design by deception: The politics of megaproject approval

[21] Kolarevic, Architecture in the Digital Age – Design and Manufacturing

translated into years of delay, which again translated into cost overruns. The simplification of the surface shells form allowed exact calculation and the use of prefabricated elements, reducing costs to more acceptable levels. Nonetheless, based on the rationalised geometric descriptions, an early example of parametric modelling was evident in the algorithmic relationship established between the shell and glass geometries. It allowed for multiple changes in the design development of form and structure. As the design evolved, the computer produced key dimensions and component templates that were transferred onto working drawings and schedules. The surface modelling of the shell geometries and the glazing were also translated into various dependent structural geometries that were offset from the surface to describe the post-tensioned concrete structures and the steel glazing supports and were thus controlled by the surface geometry[22].

Gehry's initial sketches and models for the Guggenheim Museum in Bilbao were more daring in their use of freeform curved surfaces than Utzon's, but now accurate modelling was no longer a problem. Gehry's office employed CATIA, an advanced CAD system mostly used, until then, in aerospace and automobile design, see Figure 2.7. As a result, Gehry could employ visualisation software to create, almost instantly, whatever views he needed. He could also use rapid prototyping tools to produce physical models automatically. But most important, the digital model provided the data needed for the precision in documentation which Gehry says is crucial for estimating and controlling costs correctly and thus keeping architectural design at a desired distance from controversy and political debate. Still, also for the Guggenheim Museum and other projects by Gehry Partners, rationalisation of surface geometry proved to increase the buildability of the design. Gehry and his associates on one hand have pioneered the use of digital design models that greatly facilitate production of the data needed for arriving at accurate budgets. On the other hand, making use of rationalised surface geometry contributed greatly to the explanation of how the Guggenheim Museum was built on time and within budget.

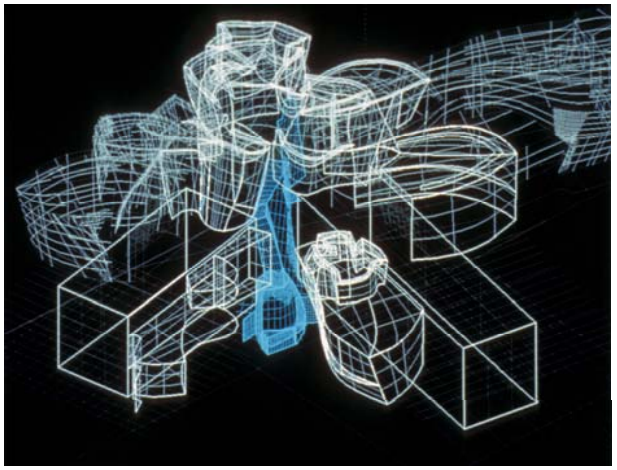


Fig. 2.7. Screenshot of the Guggenheim Museum in Bilbao in CATIA [www.dac.dk, May 2010]

2.2 generic principles of surface modelling and construction

Although the surfaces of the Sydney Opera House and the Guggenheim Museum in Bilbao are visually complex, the applied rationalisation methods lightened the complexity of the fabrication of elements and the assembly onsite. These projects exemplify that construction (as also analysis) principles may benefit greatly from rationalisation in architectural design. This section focuses on the general characteristics of surfaces and discusses some points of attention in modelling and constructing them.

[22] Schodek, Digital design and manufacturing: CAD/CAM applications in architecture and design

2.2.1 general surface properties

In general, geometric modelling of surfaces deals with two major aspects. Whereas the visual representation focuses on aesthetically appealing surface constructions and representation of forms, the analytical representation refers to mathematical descriptions and analysis of geometric properties in relationship to the forms. Subsequently, according to Mortenson, geometric modelling is also closely linked to the assembly of these forms into complex objects[23].
The two aspects of geometric modelling of surface forms and the assembly of them (both digital as onsite) can be circumscribed by three surface properties:

- surface curvature
- continuity between surfaces and
- offset properties

surface curvature

Certain curved shapes, such as hyperbolic paraboloids exhibit membrane action, wherein internal forces are efficiently transmitted through the surface in an in-plane manner. However, membrane action depends on the existence of particular combinations of surface shapes and types of loading conditions[24]. In other words, curvature does not automatically equals structural efficiency and no direct implications based on curvature analysis can be made without structural analysis. This research does not focus on the structural analysis or optimisation of surface geometry, however, it does exploit positive characteristics of rationalised surface versus non-rationalised descriptions in relation with geometric curvature analysis. In this respect, especially the distinction between surfaces of zero and non-zero Gaussian curvature is of importance. Mathematically, the concept of curvature relates to a measure of the amount of bending of a curve or surface at a point on this curve or surface. The curvature κ at a point on a surface takes on a variety of values as the plane through the normal varies. As κ varies, it achieves a minimum and a maximum (which are in perpendicular directions) known as the principal curvatures[25]. The Gaussian curvature is the product of these principal curvatures.

Generally, surfaces of zero Gaussian curvature, known as developable surfaces, contain positive characteristics in relation of fabrication and construction. However, the principal curvature in the non-zero direction cannot be neglected since the fabrication of elements is of course restricted to material properties such as flexibility. For a more in-depth description of the Gaussian curvature analysis methods, reference is made to Section 3.2 or http://mathworld.wolfram.com/GaussianCurvature.html.

---

[23] Mortenson, Geometric Modeling

[24] Schodek, Digital design and manufacturing: CAD/CAM applications in architecture and design

[25] Wolfram Mathworld, Curvature

continuity between surfaces

When combining two or more single curves into one composite curve, the continuity conditions at the joints are determined by the geometrical relationship of the control points adjacent to the joints[26]. Joining the segments at a common point so that their first *n* derivatives are equal at that point creates a condition of the *n*th order continuity. There are two kinds of continuity; geometric continuity (denoted as G^n) and parametric continuity (denoted as C^n)[27]. The least restrictive form of the *n*th order continuity is the geometric continuity; C^1 continuity implies G^1 continuity, however, the inverse is not necessarily true.

To exemplify the distinction, consider two curve segments, curve01 and curve02, with tangent vectors at respectively their end and begin point, $\mathbf{p}^u_{01}(1)$ and $\mathbf{p}^u_{02}(0)$, see Figure 2.8. When these curves are blended by another curve between these points, the composite curve is said to be a G^1 curve when the tangent vector of the blend curve at the joints is in the same direction. They only need to be scalar multiples of each other and point in the same or directly opposite direction. If the magnitude would be the same as well, or in other words if $\mathbf{p}^u_{01}(1) = \mathbf{p}^u_{04}(0)$ and $\mathbf{p}^u_{02}(0) = \mathbf{p}^u_{04}(1)$, the curve would have been a C^1 curve (a curve with first order parametric continuity).

There are different levels of continuity. If two curve segments are simply joined together at respective end points, the resulting curve is said to have G^0 or C^0 continuity, i.e. positional continuity at the join. Usually, this is far from satisfactory. If the tangent vectors at the joint for both curves point in the same direction, i.e. their geometric slopes are equal, then G^1 or C^1 continuity exists at the joint, similar to the example given above. The curve is visually continuous (smooth) but may have a discontinuity in the parameterisation[28]. G^2 or C^2 continuity exists if at the joining point the second derivative is parallel to the tangent vector at the joining point.

For n = 0, 1 and2 the characteristics of G/C^n parametric continuous curves are summed up below.

G/C^0 – *positional continuity* holds whenever the end positions of two curves or surfaces are coincidental. The curves or surfaces may still meet at an angle, giving rise to a sharp corner or edge and causing broken highlights.

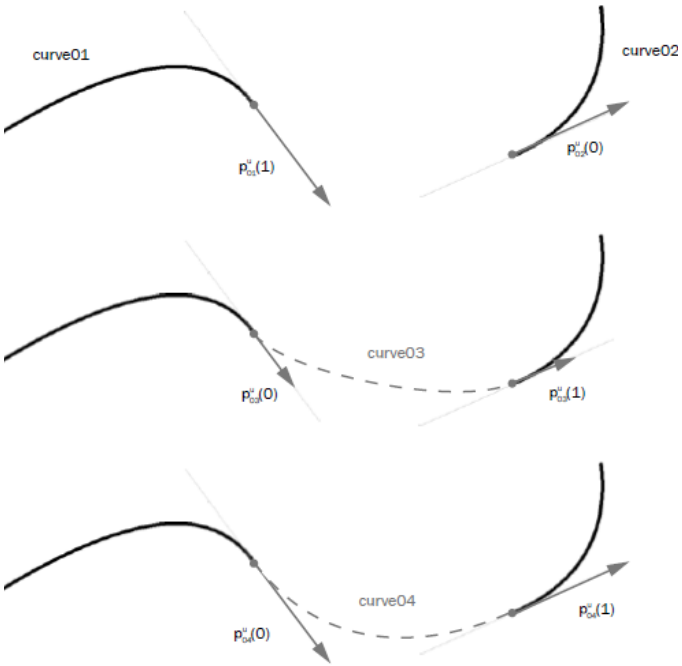


Fig. 2.8. Blending a curve between two existing curves with G^1 continuity (middle) and C^1 continuity (bottom)

[26] Watt, 3D Computer Graphic
[27] Mortenson, Geometric Modeling
[28] Piegl, The NURBS Book

G/C^1 – *tangential continuity* requires the end vectors of the curves or surfaces to be parallel, ruling out sharp edges. Because highlights falling on a tangentially continuous edge are always continuous and thus look natural, this level of continuity can often be sufficient.

G/C^2 – *curvature continuity* further requires the end vectors to be of the same direction and length and rate of length change. Highlights falling on a curvature-continuous edge do not display any change, causing the two surfaces to appear as one. This can be visually recognised as 'perfectly smooth'[29].

Since it is very difficult to visualise the difference between C^4, C^3 and even C^2 continuity, high order continuity is usually not needed. However, for applications that depend on the fairness or smoothness of a curve, especially those that depend on a smooth transition of reflected light, e.g., automobile bodies, G^1 or even G^2 continuity might not be adequate, see Figure 2.9. For these applications, at least C^2 continuity is required to achieve the desired result[30].

Since the definition of geometric continuity for surfaces is much more complex to deal with and since the scale of buildings is relatively large, usually G^1 continuity is found to be adequate in architectural design. For two surface patches with a common boundary curve it holds that they are of continuity G^1 if they have a single continuously varying tangent plane along that boundary curve[31]. Obviously, obtaining this, including the restrictions for rationalised surface constructions poses major challenges in the design process.

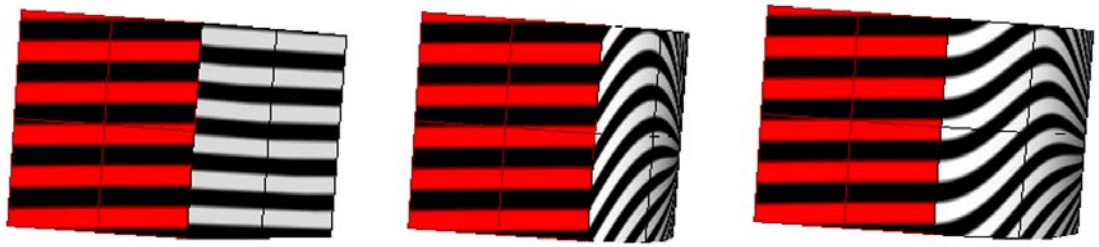


Fig. 2.9. Zebra analysis of two connected surfaces in Rhinoceros 4.0 with G^0, G^1, and G^2 continuity [www.rhino3D.com, May 2010]

[29] Wikipedia, Geometric continuity

[30] Watt, 3D computer graphics

[31] Farin, Curves and surfaces for CAGD

offset properties

In order to define distances between surfaces, for instance in modelling the top and bottom surface of a concrete shell, usually offset methods are used. Mathematically, for every surface there are two dual surfaces at a distance *d*. However, the formula for defining the offsets does not generally yield to the parameterisation of the same type as the original surface. For instance, see Figure 2.10; the generated inward offset surface is not defined by the same parametric definition as the original surface which shows that, unlike for a spherical surface, the offset of freeform surfaces generally cannot be described by the same parametric definition.

The discrepancy in parametric surface definition for surfaces and their offsets is the most clear when a sufficiently large negative distance *d* is chosen. However, in many cases in architecture, the dimensions of the surface patches are large in relation to the offset distance which makes it not too problematic that offsets are not of the same type. Nonetheless, when rationalised surface constructions are used, it may be desired that their offsets are of the same rationalisation class. Also this proves to be a complex matter to achieve. There do exist theoretical approximate methods to find the offset surfaces, but these are restrictive in usage. One generic method is the rationalisation of the designed geometry into spherical surfaces or with the help of defining curves as polynomial arcs which can be used to define rationalised geometry, see Section 2.2.3.

2.2.2 construction principles of geometric complex surfaces

Construction principles of surfaces can largely be influenced by the amount of curvature, the level of continuity and the description of their offset surfaces. Coarsely seen, there are three different manufacturing processes that exist; processes based on material removal, material deformation and casting based on moulds. Many of these processes have been used long before the use of any kind of computer technologies; they basically only have been adapted to newer control approaches[32].
An important factor in surface construction principles is the consideration of whether the surface serves as a load-bearing or non-load-bearing function, and if not, how the surface relates to a supporting primary structural system. In relation to that, it needs to be considered whether there are both external and internal surface definitions to the building volume or whether the external surface inherently defines both the external shape and the internal building volume as well. Materials such as reinforced concrete can serve as both structure and enclosure for instance. When there are both external and internal surface

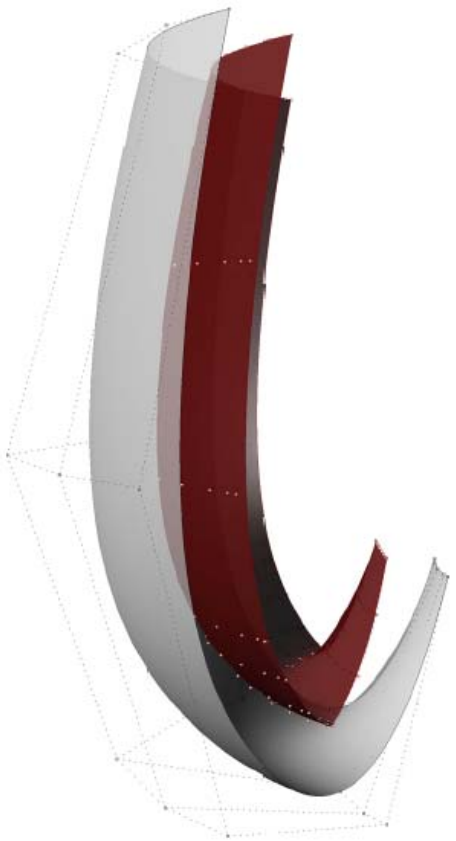


Fig. 2.10. The generation of an offset surface (red) of a NURBS surface (grey). Both surfaces are defined by different parameterisations

[32] Schodek, Digital design and manufacturing: CAD/CAM applications in architecture and design

definitions, choices are extended since structural elements can be incorporated within the space between surfaces[33].
A well-known example of a seemingly geometrically complex design is the Einstein Tower in Potsdam, designed by Erich Mendelsohn and built from 1920 to 1921, Figure 2.11. This design is of special interest since the relation between the initial design sketch and the intended method of construction – a monolithic concrete structure – could not be made in that time. Instead, it had to be built of brick, forming the load-bearing structure, with non-load-bearing thick layers of plaster on its interior and exterior faces to define surface continuity[34].

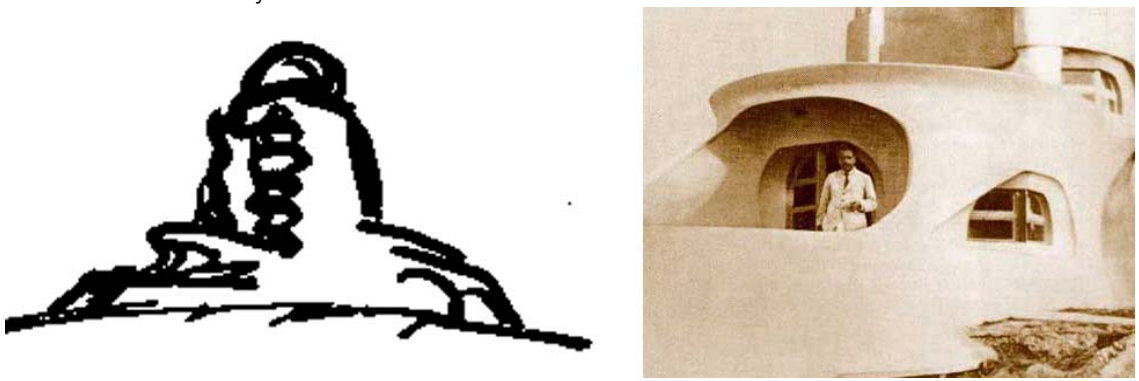


Fig. 2.11. Sketch of the Einsteinturm by Erich Mendelsohn [http://www.aip.de, June 2008] and a photograph of the tower [http://almale.blogia.com, June 2008], the only photograph with Albert Einstein standing on the tower named after him.

Another reference to pre-digital surface construction is the work of Antoni Gaudí. He faced the situation of having to provide a rational and transmittable description of highly complex shaped structures. Originally, for the Casa Milà and the Park Güell, masons literally sculpted forms sketched by the architect and under his close supervision. In the Sagrada Família, this personalised method failed to be feasible because of the sheer size and complexity of the project. During the period from 1914 till 1926 in which he worked on the Sagrada Família, Gaudí developed a set of rules that masons could follow. He generated the geometry of the principal architectural elements on the basis of ruled surfaces. These generally doubly curved surfaces can be formed by sweeping a straight line between two edge curves. The presence of straight-line

[33] ibid.

[34] Schodek, Digital design and manufacturing: CAD/CAM applications in architecture and design

generators greatly eased the work of masons, much of which could then be done without the direct supervision of Gaudí[35].
Currently, when discussing the construction of complex geometric structures, often reference is made to numerical controlled technology where fabrication instructions are given to machines via numerical codes. Computer numerical controlled (CNC) applications are for instance deployed for mass customisation of discrete elements or for the milling of complex shaped moulds by removing material from a stock shape to produce the desired object. An example of the latter application for an actual architectural object is 'The Amazing Whale Jaw', a bus station in Hoofddorp, The Netherlands by NIO Architects, Figure 2.12.

Fig. 2.12. Design, fabrication, assembly and final structure of 'The Amazing Whale Jaw' [http://architettura.supereva.com, April 2010]

Usually, this type of manufacturing is deployed when coordinates on site are not easy to set out. And although the use CNC milled foam moulds is increasing, currently manufacturing costs are usually larger then for conventional methods for moulding as are environmental costs.

CNC applications can also be used for more traditional construction methods. For the EPFL Rolex Learning Center in Lausanne, Switzerland, the contractor poured concrete over wooden formwork of more than 10.000 m^2. Due to the complexity in the shape of the shells a specific formwork solution was required. For that purpose, besides the architectural geometric model, a construction model was made[36].

[35] ibid.

[36] Architectenweb Magazine, Rolex Learning Centre - SANAA

The smoothly curved surfaces were constructed in combination with mass-customised scaffolding components, using nearly 1,500 individual wooden boxes from CNC-cut individual cleats that were positioned on exact locations using adjustable scaffolding and GPS[37], Figure 2.13.

Fig. 2.13. A digital visualisation of the plans for all of the 1,500 formwork tables consisting of almost 10,000 CNC-cut individual cleats [http://www.designtoproduction.ch, May 2010], the construction site [http://genevalunch.com, May 2010] and the finished concrete shells [http://davidgarciastudiomap.blogspot.com, May 2010]

The given examples in this subsection are continuously smooth surfaces which are constructed based on the afore mentioned processes of material removal, material deformation or casting. Generally, surfaces can be made directly from a single material that is more or less homogeneous and continuously smooth, built up in a series of layers that have been moulded or deformed or built up as a series of smaller aggregated panels that in turn may be individually homogeneous or layered[38]. Besides continuous and smooth surfaces, surfaces can also be made up of thin bendable strips that provide surfaces that appear continuous and smooth or made of a series of faceted planar faces, Figure 2.14. In order to define the three different surface types varying control elements are needed, especially if the surface types have to follow a rational description.

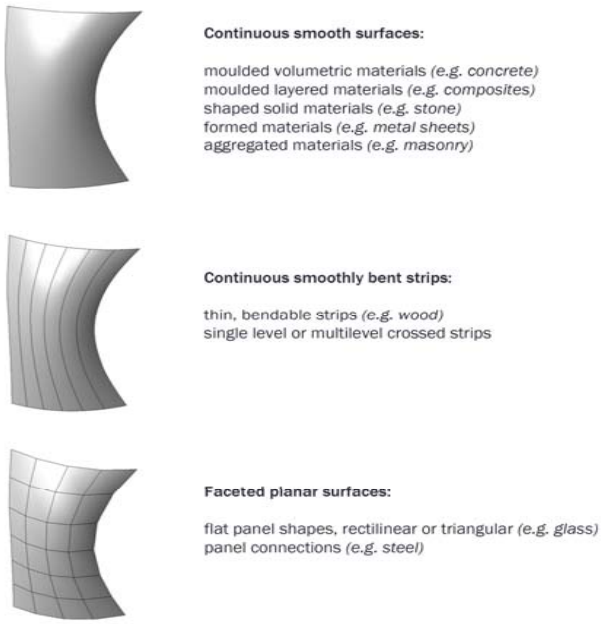


Fig. 2.14. Three primary approaches of making surfaces with compound curvatures out of rigid materials

2.2.3 control elements in surface modelling

When applying rationalisation methods, the geometry is restricted to certain rules. These rules generally are applicable to geometric objects; points, curves and surfaces. On one hand, when a designer models an object, usually the points on the object would not be thought of as a triple of coordinates, but rather in functional terms; as a corner, the midpoint between two other points, a centre point, etc. In parametric

[37] designtoproduction, EPFL Learning Center, Lausanne 2008

[38] Schodek, Digital design and manufacturing: CAD/CAM applications in architecture and design

and associative modelling, the associations between these objects play an important role in the definition of the geometry. On the other hand, the objects need to be defined in mathematical terms, the language that lends it to computer implementation. A first step toward a mathematical description is the definition of a coordinate system in which objects are described. Theoretically, the coordinate system should not affect any properties of the object itself since the interest is not in the relationship to some arbitrary coordinate system[39]. This subsection touches upon a number of geometric objects that form the control elements in describing rationalised geometry. They are the elements that are needed to control the definition and qualities of rationalised geometry.

points and vectors

Points may be used to give a reference value, describe locations on a curve or surface, or may be used as control values to describe a curve or surface[40]. Points are locations usually defined in the Cartesian space and can be represented by vectors – quantities with magnitude and direction. A vector associates the distance and direction of a point from the origin of the Cartesian coordinate system. Considers the two vectors **v** in Figure 2.15. In essence, they are exactly the same, whereas the points they connect are different since they occupy different positions relative to the coordinate axes. Basically, for every two points $\mathbf{p}_{01}$ and $\mathbf{p}_{02}$, there is a unique vector **v** that points from $\mathbf{p}_{01}$ to $\mathbf{p}_{02}$. On the other hand, given a vector **v**, there are infinitely many pairs of points $\mathbf{p}_{01}$, $\mathbf{p}_{02}$[41].

reference points

The most frequently used reference point is the origin point of the chosen coordinate system, usually denoted by (x,y,z) = (0,0,0) in the Cartesian coordinate system. As mentioned before, locations of other points can be defined by means of vector representations. These points can then be used as reference points themselves of course. Consider for instance the centre point of an arc segment. A reference point may also imply certain characteristic values on curves or surfaces. From a point on a surface, a normal vector to that surface can be distilled.

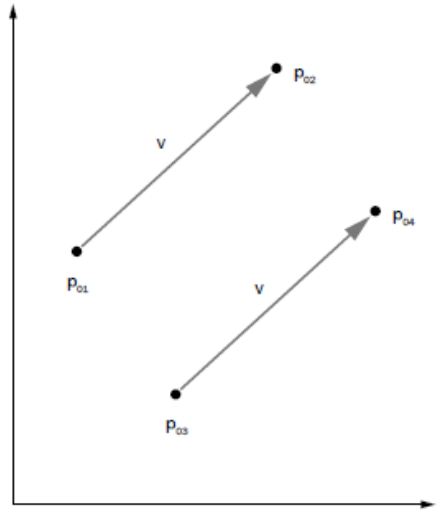


Fig. 2.15. Point-vector relations; different point sets but similar vectors

[39] Farin, Curves and surfaces for CAGD

[40] Hardy, Mathematical tools in computer graphics with C# implementations

[41] Farin, Curves and surfaces for CAGD

interpolation points

The simplest form of using interpolation points is by drawing a polyline through those points after defining the order of them. However, when the points are used to define freeform curves, every point needs additional input. Since there are infinitely many different interpolation curves that pass through the same set of points, the curve tangents at the interpolation points have to be given as well[42].

vertices

From a point set *P* on a surface a polygon mesh can be described. The polygons are described by a set of faces *F* where each element of *F* is a set of edges that describe that face. The edges are simply described by [$\mathbf{p}_{01}$, $\mathbf{p}_{02}$] where $\mathbf{p}_{01}$, $\mathbf{p}_{02} \in P$. This means that points only store positional information. In a parametric system where face edges are based on their association with points, the set of faces *F* stores only connectivity information[43].

control points

When points are used to locally control the surface shape they are defined as control points. The control points still only contain positional information. The resulting surface, however, is based on a relatively complex description. As a result, interesting questions arise when a Cartesian point set needs to be mapped on a surface defined by control points.
For instance, a non-uniform rational B-spline [NURBS] surface of degree (u, v) with control points $\mathbf{p}_{i,j}$, i = 0,...,k, j = 0,...,l, weights $w_{i,j}$, i = 0,...,k, j = 0,...,l, is defined by

$$S(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} w_{i,j}\mathbf{p}_{i,j} N_{i,k}(u)N_{j,l}(v)}{\sum_{i=0}^{n}\sum_{j=0}^{m} w_{i,j}N_{i,s}(u)N_{j,t}(v)} \qquad \begin{matrix} u \in [0,1] \\ v \in [0,1] \end{matrix} \qquad \text{(Eq. 2.1)}$$

where $N_{i,k}$ and $N_{j,l}$ are the B-spline basis functions[44].

[42] Pottmann, Architectural geometry
[43] Hardy, Mathematical tools in computer graphics with C# implementations
[44] Wolfram Mathworld, NURBS Surface

point cloud

Point clouds – patterns of points – are the result of digitising models, also known as reverse engineering. The point cloud is created from the physical model through scanning, see Figure 2.16, and is then interpreted by conversion software to produce a close approximation of the model's geometry. A digitiser creates a correspondence between physical points and virtual points from which the geometry of an object can be described[45].
As mentioned before, Gehry uses reverse engineering as a medium for translation in a process that takes as its input the geometry of the physical model and produces as its output the digitally-encoded control information which is used to drive various fabrication machines[46]. See also Figure 2.6.

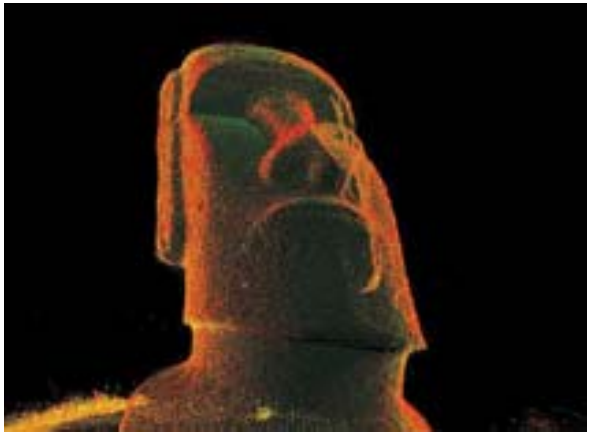


Fig. 2.16. Point cloud of a digitised Moai made by a team of AutoDesk experts in 2007. The team was tasked with creating and using 3D digital models to better visualise and analyse how development plans will impact residents and resources [http://www.acronymonline.org, May 2010]

curves – types and geometry

Curve representations are specified by points and are based on interpolation or approximation for respectively data (or interpolation) points or control points. Basically, a curve is not much more than the connection of one-dimensional series of points. But curves, obviously, do contain more information than just the coordinates provided by the points on the curve. The curve length and the local normal and tangent are examples of this. Additionally, depending on the method of description of the curve, other types of information, such as the 'unit speed' reparameterisation, is added (see Section 3.2 for a description of the parametric representation which allows for the unit speed reparameterisation).

Below, two curve types and two curve geometry types are discussed which are widely used in the description of rational surface classes.

surface profiles

Whereas curves are one-dimensional series of points, surfaces can be considered as a two-dimensional object in space based on specific associations between curves. Translational surfaces (see Section 2.3.2 for more information), for instance, are generated by moving a curve along another curve according to a specified way. The curves that define the surface profiles are called a directrix and generatrix. Consider for instance the generation of the hyperbolic paraboloid in Figure 2.17. One parabola, the generatrix, is copied over the second parabola, the directrix, to create the surface geometry.

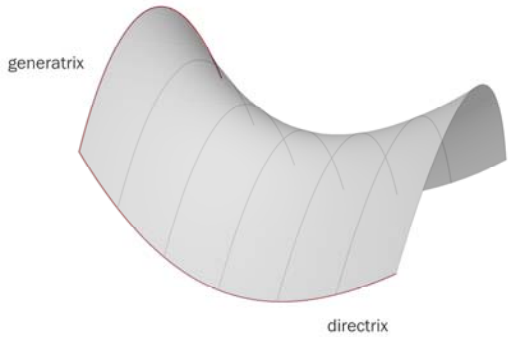


Fig. 2.17. A possible generation of a hyperbolic paraboloid, or HP surface, based on the translation of a parabolic generatrix over a parabolic directrix

[45] Lindsey, Digital Gehry – Material resistance, digital construction

[46] Kolarevic, Architecture in the Digital Age – Design and Manufacturing

The surface profiles themselves can take any desired shape. However, in order to generate rationalised surfaces, the surface profiles have to follow specific rules dictated by the type of the surface rationalisation method.


generators

A special type of generatrix is the contour generator or ruling. In Subsection 2.2.2 it was already mentioned that Antoni Gaudí informed his masons of the possibilities of creating double curved surfaces with these rulings. Generators can be straight lines that move along a directrix or a line joining curve points on directrices corresponding to the same parameter value.
When the rulings follow specific geometric rules, a subclass of ruled surfaces can be generated, called developable surfaces, see also the texts on Frank Gehry, Section 2.1. Both ruled and developable surfaces play an important role in the rationalisation of surface geometry. These surface types are described and explained in greater detail in part02 of this thesis.


straight curves

The straight line is the simplest of curves in geometric modelling. Concerning construction, straight structural elements are preferred above bent or curved elements in relation to costs and analysis, hence the beneficial characteristics of a ruled surface. As a result, straight curves are widely used in geometric modelling. Straight curves are also frequently used to define axes for rotational surfaces, as directrix for extrusion surfaces and as face edges in polygonal surfaces.


radial curves

For complex curves, rationalisations based on the radial geometry of circles can be deployed to approximate these curve, Figure 2.18. Radial curves are also often used as the generative element of a design. Consider for instance the International Terminal at Waterloo Station in London, Great Britain. The design by Grimshaw and Partners consists of parametrically configured arches, defined by two arcs with different radii. Similarly, many surface constructions are based on the radial geometry of spheres, cones, cylinders and tori, of which the afore mentioned Sydney Opera House is one.

Fig. 2.18. Approximation of some freeform curves on a Porsche 911 Turbo [http://www.dieselstation.com, January 2010]. It must be noted that the complexity of the geometric description of curves and surfaces, both in modelling as in fabrication, in the automotive industry is more advanced than in the building industry, partly because of the possibilities of mass-production, making this example a little presumptuous

2.3 architectural features of rationalised surfaces constructions

Section 2.1 showed examples of two architectural designs where rationalisation methods were embedded in the design process or even formed the driving force of the geometric description of the surface construction. This section discusses the architectural qualities of rationalised surface classes and their advantages and shortcomings in geometric modelling and construction. The focus is on three types of surface construction; surfaces of revolution, ruled surfaces and translational surfaces.

2.3.1 rationalised surface classes in architecture

Many surface definitions can be categorised in multiple rationalised surface classes. Consider for instance an open cylinder, Figure 2.19. Examples described in the following texts, therefore, are not necessarily restricted to the rationalised surface construction which is being discussed.

surfaces of revolution in architecture

A spherical surface is exemplary for the description of surfaces of revolution as the surface is simply generated by revolving a half circle about an axis over 360 degrees, Figure 2.20. The profile curve can take any shape in order to create other surfaces of revolution, however, it is recommended that planar meridians curves are used rather than arbitrary spatial curves, since they are symmetric to the rotational axis[47]. When the profile start and end point do not intersect with the axis, the resulting surface is open. However, when the meridian curves intersect the rotational axis at an angle different from a right angle, a singular point is obtained on the rotational surface which can become very critical in the subsequent design process[48]. It is also possible to use closed profile curves to describe surfaces of revolution, as is the case in the generation of tori, where a circle is revolved around an axis.

The shape and location of the surface thus depends on the location and direction of the generator and the type and location of the profile relative to the generator. Based on this relatively small amount of data to describe surfaces of revolution, they contain various advantageous characteristics, amongst which the relatively straightforward setting out of reference points and points on the surface on site and the fact that offsets of surfaces of revolution are generally easier to describe than for other types of surfaces, Figure 2.21.

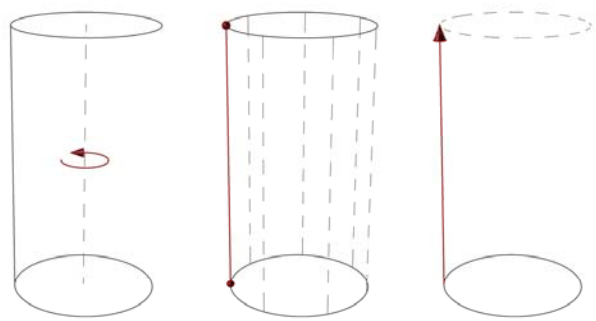


Fig. 2.19. Three methods of surface descriptions for an open cylinder. From left to right: surface of revolution, ruled surface and translational surface

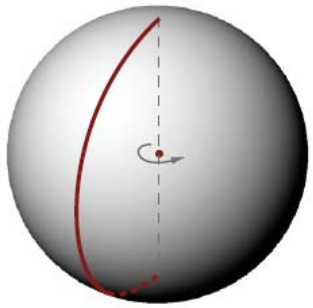


Fig. 2.20. A spherical surface constructed by revolving a radial curve (half circle) over 360 degrees about an axis, the generator

[47] Pottmann, Architectural geometry

[48] ibid.

Because of these positive characteristics, surfaces of revolution are frequently used in architectural design where they are used to approximate initial design intentions or where they are the leading geometric descriptive force. Especially surfaces that are based on circular profile curves – see the spherical surfaces of the Sydney Opera House, the arc based surfaces of the Chesa Futura in St Moritz, Switzerland by Foster and the toroidal shapes of the TGV train station in Avignon, France by Arep – are well known.

A negative issue that needs to be dealt with in the design of surfaces of revolution is that they might be hard to construct when circular or arbitrarily curve profiles are used, Figure 2.22. This also relates to the fact that spherical surfaces are nondevelopable. The complexity in construction is greatly reduced when for instance straight lines can be used to shape the surface and to define structural elements.

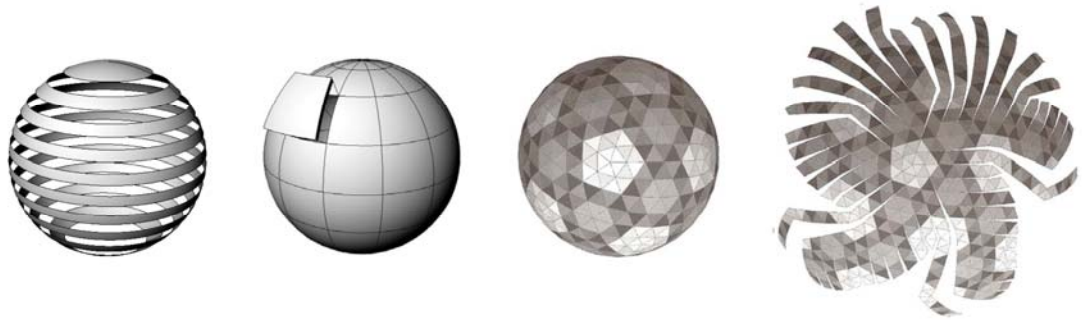


Fig. 2.22. Possible construction principles of spherical surfaces. From left to right: parallel strips, double curved segments and approximation as a polyhedral surface (an icosahedral geodesic sphere). The far right: the unfolded geodesic sphere[49]

ruled surfaces in architecture

Definitions of rules surfaces are based on the sweeping motion of a straight line, resulting in surfaces that contain straight-line generators. But the definition by straight lines is not only beneficial to the construction of the surface itself, it also allows for easy incorporation of straight structural elements, such as beams, making ruled surfaces well applicable for architectural purposes.

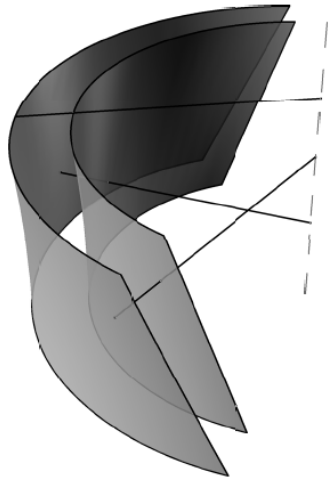


Fig. 2.21. Normal vectors of the surface of revolution and its offset intersect with the rotational axis making the description of offset surface relatively easy

[49] Szalapaj, Contemporary architecture and the digital design process

Several possibilities exist for defining the motion of a line generating a ruled surface. Pottmann et al[50]. describe two methods for ruled surface generation, Figure 2.23:

- ruled surfaces by moving a straight line along a directrix curve
- ruled surfaces by connecting corresponding points of two generating curves

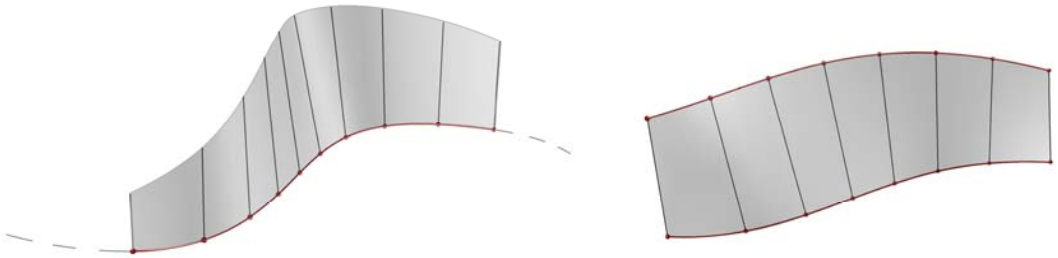


Fig. 2.23. Two methods for generating ruled surfaces. Left: moving a straight line along a directrix curve. Right: connecting corresponding points of two generating curves

An example in architecture where the use of rulings is strongly expressed is the HtwoOexpo fresh water pavilion on the Neeltje Jans Island, The Netherlands by NOX, Figure 2.24. Similar to the design of the salt-water pavilion by Kas Oosterhuis, the fresh water pavilion design was largely based on digital design methods, however construction followed a more traditional approach. This is expressed by the form itself which is shaped by the fluid deformation of fourteen ellipses spaced out over a length of more than 65 meters[51] and substructure of the covering steel sheeting that was supported by the relatively straightforward construction principle of straight beams which span between these ellipses. These beams followed the translation of a straight line over the elliptic curve profiles, resulting in the ruled surface that defined the final shape of the building.

Depending on the definition of the sweeping motion of the straight line, different subclasses of ruled surfaces exist, each with their advantages, but also with restrictions to the architectural descriptive language. Two of these subclasses are extruded surfaces and developable surfaces.

[50] Pottmann, Architectural geometry

[51] Jodidio, Architecture NOW!

Fig. 2.24. Top: the exterior surface of the Htwo0expo pavilion
Bottom: beam structure supporting the steel sheeting

extruded surfaces

An extruded surface is obtained when a spatial generating curve is translated in the direction of a straight trajectory line[52], also called the sweep vector. This makes an extrusion a special case of a lofted surface, see also Section 2.3.3), which is a basic ruled surface generated by interpolating two curves. Extruded surfaces can also be classified as sub classes of translational surfaces, with the restriction that the path curve is a straight line.

This makes extruded surfaces rather restrictive in the possible surface configurations. Parallel extrusions of a smooth curve generates a cylinder surface, whereas a central extrusion generates a cone surface, Figure 2.25, which are also two basic typologies of developable surfaces.

developable surfaces

David Huffman states in his paper 'Curvature and creases: A primer on paper' that a developable surface offers a complexity that is in a very real sense exactly midway between that of a completely general surface and that of a plane surface. Consequently, developable surfaces constitute a class that might be ideally suited to be both richer than that of plane surfaces and more tractable analytically than that of totally arbitrary surfaces[53]. To support this quote, it can be noted that it is unlikely that one can say much of practical value about surfaces of complete generality. Usually, no two neighbouring points on an arbitrary surfaces have the same tangent plane. On a developable surface, all points on a given isoline embedded in the surface have the same tangent plane, i.e. the neighbourhood of a point on a developable surface can be characterised by a single parameter family of tangent planes.

As mentioned before, heavily touching upon this duality between complexity and analytical description is the architecture of Frank Gehry, see also Section 2.1. Multiple designs by Gehry's office follow the visual complexity of arbitrarily shaped surfaces, Figure 2.26, in combination with the strict geometrical rules of developable surfaces.

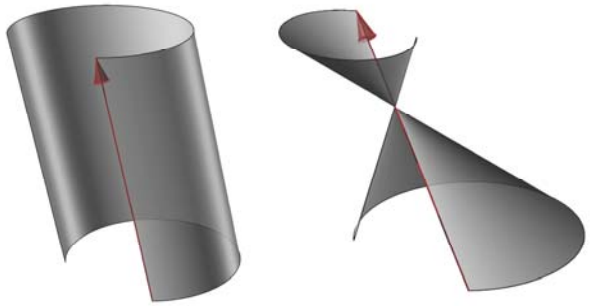


Fig. 2.25. Parallel and central extrusion of a open freeform curve

[52] Marsh, Applied geometry for computer graphics and CAD

[53] Huffman, Curvature and creases: A primer on paper

Fig. 2.26. Gehry's way of designing according to the makers of The Simpsons. Left: the 'real' Frank Gehry [http://www.metropolismag.com, April 2010]. Middle: screenshot from the Simpsons episode featuring Frank Gehry as a cartoon character [http://www.yangsquare.com, April 2010]. Right: the Starwood hotel in Elciego, Spain [http://www.starwoodhotels.com, April 2010]

Similar to general ruled surfaces and as mentioned above, developable surfaces contain sets of straight lines which simplifies construction of these surfaces. Additionally, developable surfaces have the advantage that they can be made from flat surface material, such as metal sheeting. As such, next to plane surfaces, they can be categorised in three basic types: cylinders, cones (which, as mentioned before, can be classified as extruded surfaces as well) and tangent surfaces of space curves[54].

Regarding the latter type, consider the problem of creating a surface based on two curves in space. An infinite number of surfaces, including ruled surfaces, can be found to span the curves, but a developable surface is unique. Consider the following; a point on a surface denotes a plane which is tangent to the surface at that point. Generally, for ruled surfaces different points on the same generator have different tangent planes; the tangent planes rotate around the generator[55], Figure 2.27. Such rulings are called non-torsal generators. Developable surfaces contain exclusively rulings where a single tangent plane touches the surface along the entire line. These rulings are called torsal generators, Figure 2.28.

However, not every set of two arbitrary curves can be used to generate a developable surface. For design purposes that adopt developable surfaces it is needed to generate the input curves based on specific rules. Nolan describes it as follows: Rulings reflect the character of the curves they span. If the boundary curves are fair, the locus of rulings defining a developable surface between them will also be fair and continuous. The problem is thus reduced to one of finding a fair representation for the boundary curves[56].

[54] Pottmann, Architectural geometry

[55] ibid.

[56] Nolan, Computer-aided design of developable hull surfaces

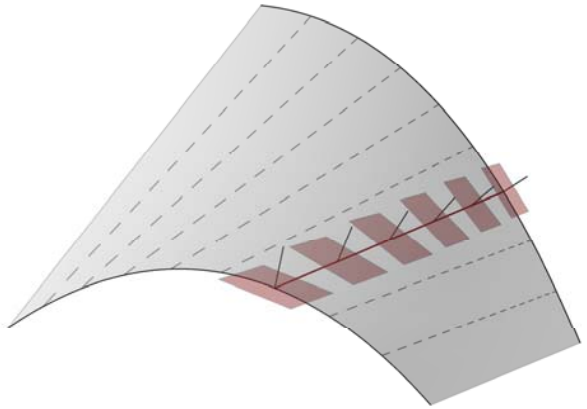


Fig. 2.27. Rotating tangent planes on a non-torsal generator for a ruled surface

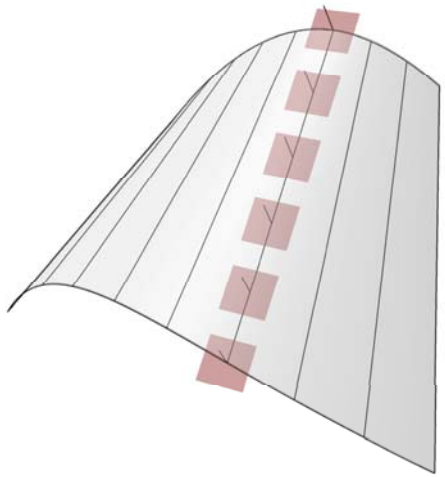


Fig. 2.28. Developable surface with torsal generators. The surface tangent planes of the generators lie in one plane, indicating that for each generator there is a single plane tangent to the surface along the entire generator

Following from the geometric and algorithmic prerequisites, developable surfaces have zero Gaussian curvature. The other way around, surfaces having zero Gaussian curvature by definition contain embedded 'generating' lines in at least one direction. Another special characteristic of developable surfaces is that they can be mapped isometrically (i.e. mapping between surfaces that preserve the length of any curve) onto a plane. As a result, the planar isometric image of a developable surface is the planar unfolding of the surface[57].

To exemplify this, regard the mapping of a sphere onto a plane. Since a sphere is a nondevelopable surface, there is no unambiguous way in which a sphere can be mapped to a planar rectangle outline. As a result, multiple methods for projecting the earth's surface onto a planar map exist.

Generally, the earth's surface – approximately a sphere – is mapped via the Mercator projection, Figure 2.30. This is a cylindrical projection, meaning that it is a projection in which lines of longitude (meridians) are mapped to equally spaced parallel lines and circles of latitude are mapped to parallel lines with arbitrary mathematically spaced separations[58].
When unfolding the earth's surface, the map is stretched out in east-west direction in an increasing degree away from the equator, Figure 2.29. In the Mercator map, this is accompanied by a corresponding north-south stretching keeping the direction and shapes of land masses accurate, but sizes distorted. In other words, being a conformal projection, the Mercator projection preserves angles around all locations, however scale varies from place to place, distorting the size of geographical objects[59].

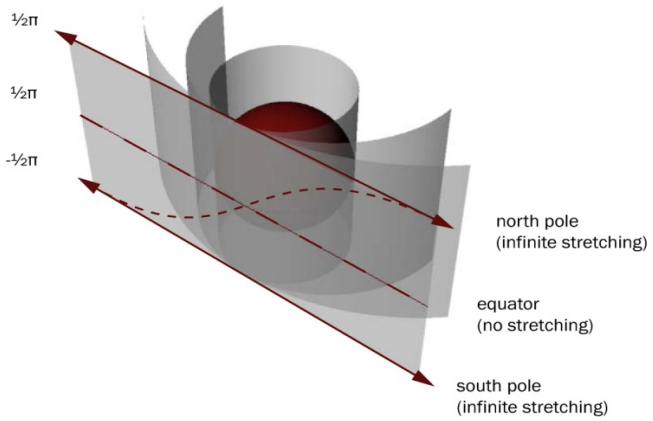


Fig. 2.29. Visualisation of the stretching of a cylindrical projection from pole to pole

In order to determine the x and y coordinates of a point on a Mercator map from its latitude φ and longitude λ the following mathematical equations need to be applied

$$x = \lambda - \lambda_0 \qquad \text{(Eq. 2.2)}$$

$$y = \ln\left[\tan\left(\frac{\pi}{4} + \frac{1}{2}\varphi\right)\right] \qquad \text{(Eq. 2.3)}$$

These equations place the x-axis of the projection on the equator and the y-axis at longitude λ_0.

[57] Pottmann, Architectural geometry
[58] Wolfram Mathworld, Cylindrical projection
[59] Design IntelligenceS, Concepts as tools

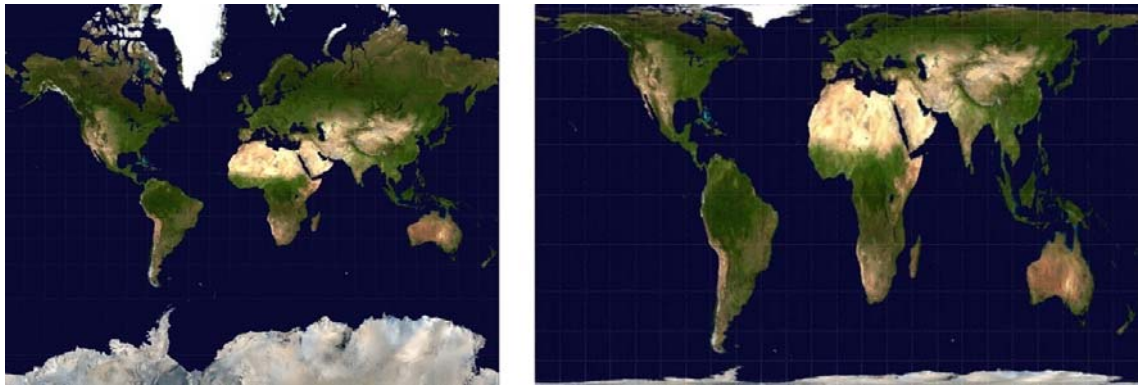


Fig. 2.30. Left: The Mercator projection of the earth's surface Right: The Gall-Peters projection, known for the cylindrical equal-area projection* [http://designintelligences.wordpress.com, March 2010]

For the projection from a Mercator map back to a spherical surface the inverses of equations 2.2 and 2.3 are needed.

$$\lambda = x - \lambda_0 \qquad \text{(Eq. 2.4)}$$

$$\varphi = 2\tan^{-1}\left(e^{y}\right) - \frac{1}{2}\pi \qquad \text{(Eq. 2.5)}$$

With these equations, the earth's land masses can be placed on a spherical surface based on points lying on the continental outlines of a Mercator map. In Rhino's Grasshopper, points that subdivide the continental outlines on a plane – which are modelled in Rhino, based on a background image of a Mercator map – can be transposed to points that describe the outlines on a spherical surface via these equations. Figure 2.31 shows the resulting image when these points are interpolated by curves which form the enclosing boundaries of the earth's land masses.

*Each type of projection creates different perceptions of the world. The Mercator projection distorts the sizes of the earth's land masses to a greater extent when farther away from the equator, for instance 'enlarging' North America and Europe. We accept this kind of mapping basically because we are used to it. A more honest projection in relation to sizes – not shapes – is the Gall-Petersen projection which preserves the surface area of the continents, showing that Africa actually is 14 times larger than Greenland.

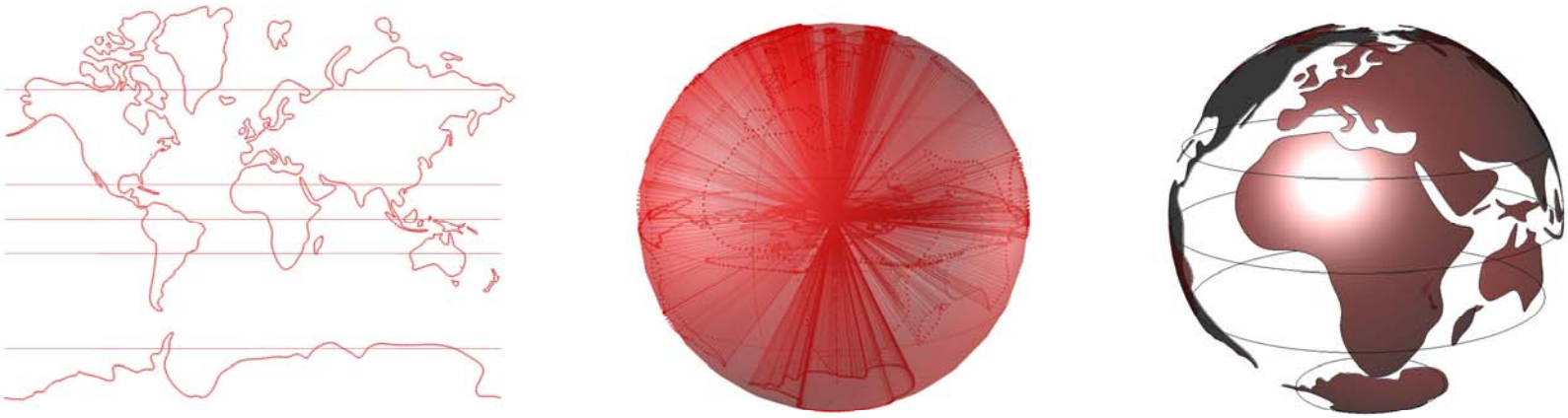


Fig. 2.31. Left: outline of the earth's land masses based on the Mercator projection. Middle: projection of points on the outline onto a spherical surface. Right: representation of the continents by trimming the spherical surface with the continental outline

2.3.2 polyhedral surfaces in architecture

Apart from imposing constraints to the geometric description of surfaces via mathematical definitions, a geometric complex design can be rationalised using meshes. Meshes are frequently used in digital animation where subdivision surfaces are used which are generated by a simple refinement rule for meshes and which is applied iteratively till the surface is considered smooth enough and use a secondary, more complex algorithm to approximate curvature. Also in engineering, where Finite Element Models (FEM) are being used for simulations and analysis, meshes are of importance. In architecture, meshes are used to rationalise surfaces, both complex and noncomplex, and subdivide them in usually planar faces, straight edges and vertices, Figure 2.32. These types of surfaces are generally called polyhedra or polyhedral surfaces.

Fig. 2.32. 'Meshed' surfaces in architecture. Prada store in Tokyo by Herzog & De Meuron [http://liaoyusheng.com, May 2010] and the Cockpit of the Acoustic Barrier by ONL [http://www.miwian.nl, May 2010]

polyhedral surfaces and polyhedra

Polyhedral surfaces – also piecewise-planar surfaces – are generally simple approximants to more complex surfaces and are used to subdivide complex shapes into individual components, often steel bars and glass panels. In other words, meshing and subdivision techniques break the surface into 'tiles'. Hence, a polyhedral surface or subdivided mesh is also called a tessellated surface. When evaluating tessellation strategies, if the aim is to calibrate the initial form with a structural system, the size and resolution of the faces are best determined relative to the overall geometry and design intention and with regard to the final building materials and construction processes.

In the building industry, usually three, four or six sided regular polygons are used; equilateral triangles, squares (quadrilaterals) and hexagons. The characteristics of these polygons are well known. They are symmetrical, equilateral (having sides of equal length), equiangular (having equal angles) and they can be inscribed by a circle[61].

Early examples of subdividing surfaces into smaller components in architecture are mosaics of walls and floors of buildings of ancient Rome and those of the Byzantine Empire or the Islamic architecture. Overall patterns were typically achieved by assembling many small pieces into a coherent design based on sophisticated mathematics. Consider for instance girih patterns found on many walls of medieval Islamic buildings. Harvard and Princeton University researchers believe that the girih appear to show an advanced geometry, called decagonal quasicrystal geometry, that was not developed in the west until the 1970s. Figure 2.33 shows a Girih pattern of a decorated arch in the Sultan's Loge of the Ottoman-era Green Mosque in Bursa, Turkey, which was completed in 1424[62]. On the right side of the figure, the five non-regular, but equilateral polygons — a hexagon, a bowtie, a decagon, a rhombus and a pentagon – are highlighted in a random combination of tiles. When closely looked at, it becomes clear that not the joints between the tiles form the pattern, but that the girih lines that are interconnected over the tiles give expression to the shape on the decorated arch. Despite their non-periodic nature, the resulting configurations are not chaotic. Rather, they fit together in a way that is predictable but difficult for the brain to perceive.

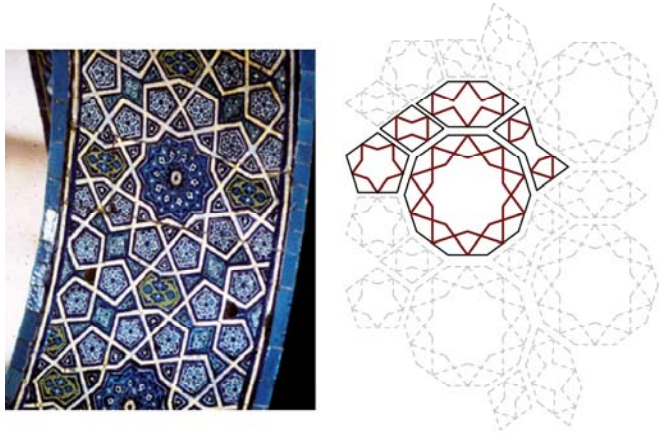


Fig. 2.33. Left: a decorated arch in the Sultan's Loge of the Ottoman-era Green Mosque in Bursa showing a girih pattern based on specific allocation of girih tiles.
[http://www.saudiaramcoworld.com, May 2010]
Right: a random combination of girih tiles with the base equilateral polygons highlighted

Buckminster Fuller's geodesic domes are more recent examples of the smooth surface approximation via tessellation. In Fuller's pursuit of lightness and engineered efficiency, the spherical shape is redefined as a pattern of triangles or hexagons with every strut, opening and joint being identical. While this uniformity contributes to ready constructability and overall material efficiency, it is unrelenting in terms of form.
Today, the need for standardising structural elements is less relevant since digital fabrication allows for the design with nonstandard units (mass customised) to approximate these surfaces and as such, digital technologies have revitalised the interest in patterning and tessellation because they afford greater variation and modulation through nonstandard manufacturing[63]. Much of the costs of tessellated structures goes into node connections and the curvature of the mesh faces.

[61] Blackwell, Geometry in architecture
[62] Saudi Aramco World, The tiles of infinity
[63] Iwamoto, Digital fabrications – Architectural and material techniques

Since flat panels are less expensive than (double) curved panels, triangular meshes are the governing type for the discretisation of complex surfaces. Generally, triangular meshes have been used in architecture whenever surfaces cannot be easily planarised in another way, and as such are frequently adopted, since triangle faces are always planar. But, twice as many triangles as quads are needed to represent the same shape[64]. When economics are governing, the focus usually is on quadrangular meshes with planar faces. This mainly because they tend to have less weight and can be constructed with geometrically optimised nodes in the supporting beam layout. However, the geometry of such meshes is more difficult than that of triangle meshes[65]. In order to arrange for the design with planar quadrangular meshes on complex surfaces, surfaces can be modelled based on specific rules. Amongst others, translational surface follow such a set of certain rules.

translational surfaces in architecture

The extruded surfaces that were discussed in the previous subsection can be generalised to translational swept surfaces which are obtained by translating a generating curve (generatrix) along a trajectory curve (directrix)[66]. These translational surfaces embed the specific rules needed to generate planar quadrangular faces on double curved surfaces. However, since both input curves may be arbitrarily shaped, the resulting surface may well have high global curvature[67].

Since a translational surface contains two sets of section curves that are congruent with the profile curves (meaning that the section curves can only be generated by transformation of the profile curves by an isometry[68], i.e. a combination of translations, rotations and reflections), this method does not allow any arbitrary surface shape to be created. Nonetheless a large variety of surface forms is possible, especially if a surface has one strong direction.

The translational surface method is based on the mathematical principle that two parallel vectors in space always define a planar surface. The vectors and the connection between their points of origin and end points make up the four edges of the quadrangular surface. However, a planar surface may also be defined by two vectors not running parallel to each other, but do lie in the same plane.

64 Pottmann, Architectural Geometry

65 ibid.

66 Marsh, Applied geometry for computer graphics and CAD

67 Farin, Curves and surfaces for CAGD

68 Wolfram Mathworld, Congruent

To exemplify this, regard a quadrangular mesh with a sectional curve $\mathbf{C}_{01}$ in one direction and its individual sections being the lateral edges of the mesh faces and another sectional curve $\mathbf{C}_{02}$ in a different direction with its individual sections being the longitudinal edges of the mesh faces, Figure 2.34. For translational surfaces, both the lateral (direction of blue vectors) and the longitudinal (direction of green vectors) edges of a row of meshes form parallel vectors. Subdividing the directrix and the generatrix equally, results in a grid with constant bar length and planar mesh. If, for example, one parabola translates against another parabola perpendicular to it, the result will be an elliptical paraboloid with an elliptic layout curve. Two identical parabolas generate a rotational paraboloid with a circular layout curve. A directrix curving anticlastically to the generatrix results in a hyperbolic paraboloid, which can also be formed by two systems of linear generatrices[69]. Figure 2.35 shows an example based on arbitrarily shaped curve profiles to generated a planar mesh structure.

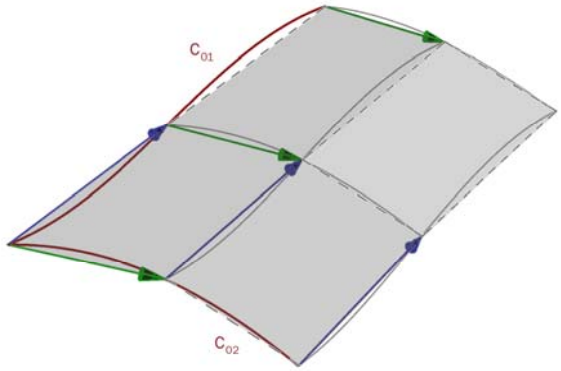


Fig. 2.34. Generation of a translational surface; the translation of curve $\mathbf{C}_{02}$ along $\mathbf{C}_{01}$ or vice versa

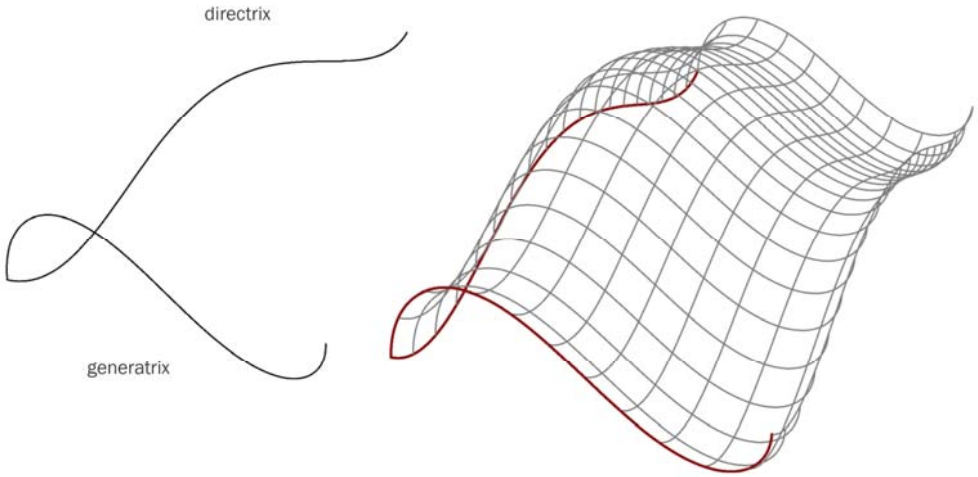


Fig. 2.35. Translational surface based on arbitrarily shaped curve profiles

[69] Pottmann, Architectural Geometry

When only the lateral or longitudinal edges of a row of meshes form parallel vectors, a special type of the general translation surfaces is generated; the scale-trans surface. A restriction is that the profile curve that is translated is planar itself. In Figure 2.36, a scale-trans surface patch is presented where the section curves in the lateral direction are scaled, respectively with a scale factor of 0.75 and 0.825. Due to the scaling of the lateral section curves, the longitudinal edges of the faces are not parallel any more.

The scaling of any sectional curve yields a new curve with parallel edges in the scaling direction. The base point of the scaling operation may be chosen at random. In other words, the scaling causes each lateral edge vector of the sectional curve to lengthen or shorten by the same factor, while maintaining its direction. The longitudinal edges are determined by the line between the points of origin and the end points of the respective lateral edge vectors. The next row of meshes will be created following the same principle with the shape of the new sectional curve depending on the scaling factor, where it is also possible to translate the centre of scaling of the sectional curves along the directrix. Figure 2.37 shows a scale-trans surface, generated by a scaled translation of a open circular-like curve along an arbitrarily directrix.

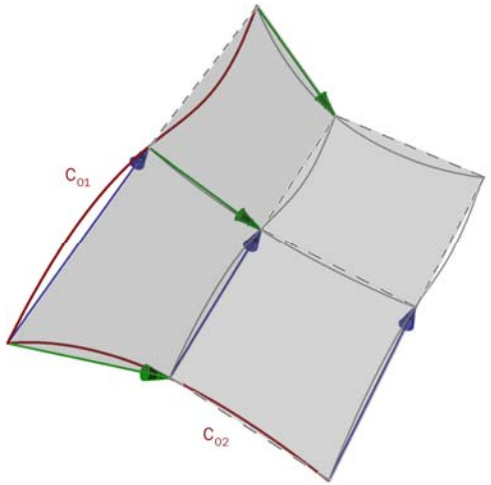


Fig. 2.36. A scale-trans surface patch with parallel face edge vectors in only one direction

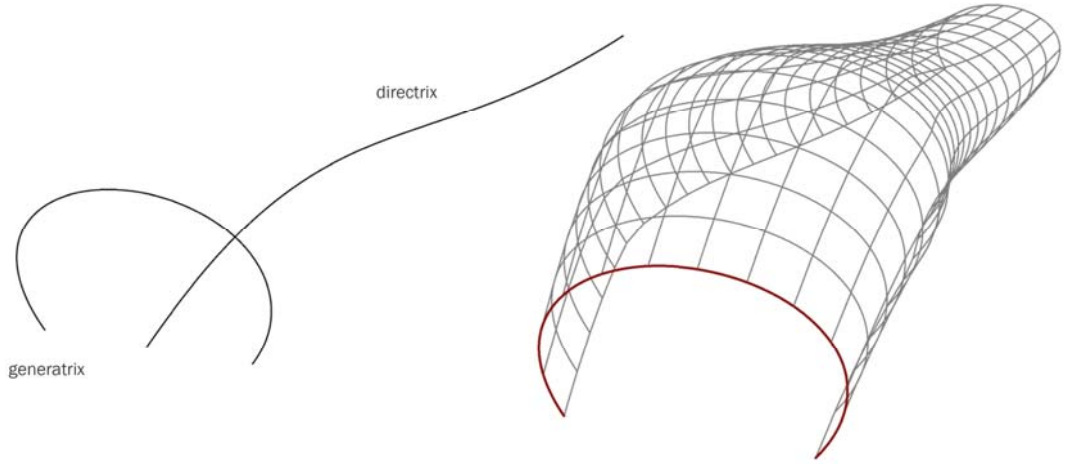


Fig. 2.37. Trans-scale surface based on a scaled translation of a open circular-like curve and an arbitrarily directrix

Translational surfaces and scale-trans-surfaces have frequently been used by the engineers of Schlaich, Bergermann and Partners. In Leicht Weit – Light structures, Schlaich states that the aim behind creating geometric surfaces with translational surfaces and scale-trans-surfaces is to produce meshes in which all four joints are on one and the same level to ensure that they can be covered economically with planar glass panels[70]. Two examples are shown in Figure 2.38.

Fig. 2.38. Left: the Hippo House at the Berlin Zoo, Berlin, Germany. Right: the Hauptbahnhof Berlin, Berlin, Germany [both from http://www.sbp.de, May 2010]

In conclusion, this research does neither focus on meshing strategies of complex geometry, nor does it research new methods of meshing or subdivision or look for optimal solutions to known mesh strategies. Firstly, because meshing itself is a fully new topic in relation to the generation of rationalised surface classes. And secondly, because meshing strategies are more based on post rationalisation, whereas the main goal is to incorporate pre-rationalisation design methods.

2.3.3 methods of surface generation based on motion

Although these surface are not classified as rationalised surfaces, some surface descriptions based on generation by motion are closely linked to rationalised surface classes which are discussed above. Below, some of these methods of surface generation which are based on motion of curves are discussed briefly.

[70] Schlaich, Leicht Weit – Light Structures

sweeping

Sweeping is a frequently used method in architectural design to create freeform surfaces. Generally, sweep representations of surfaces are based on the notion of moving a curve along a path. The relatively constrained geometry of a sweep shape means that only a small data set is needed to specify the shape[71].

Sweep representations correspond to various surface classes. Three rational surface classes that are based on 3D sweeping motions are surfaces of revolution, translational surfaces and ruled surfaces, in particular extruded surfaces. A standard rotational sweep is defined by rotating a planar curve about an axis. The standard translational sweep and extrusion are defined by respectively moving a straight line along an arbitrary curve normal to the plane of the curve and moving an arbitrary curve along a straight line. When the shape of the curve being swept does not change, a linear sweep representation is generated.

A general sweeps is one whose generating curve, the generator, follows some arbitrary curved path, the director curve, and which itself may change size, shape and orientation. The parametric generalisation of a sweep surface is

$$\mathbf{S}(u,v) = \mathbf{C}(u) + f_1(v)\mathbf{u}_1(u) + f_2(v)\mathbf{u}_2(u) + f_3(v)\mathbf{u}_3(u) + \ldots \qquad \text{(Eq. 2.6)}$$

where $\mathbf{C}(u)$ is the curve along which a set of coordinate system frames ($\mathbf{u}_1(u)$, $\mathbf{u}_2(u)$, $\mathbf{u}_3(u)$,...) is swept. Within these coordinate system frames, the curve $f(v) = (f_1(v), f_2(v), f_3(v),...)$ determines the shape of the curves over the directrix $\mathbf{C}(u)$, Figure 2.39.

Depending on the characteristics of the components in Equation 2.6, a surface of revolution, translational surface or extruded surface is generated. It is also possible to describe a sweep surface based on the space curve $\mathbf{C}(u)$ and a transformation matrix dependent of v. The transformation must include translation and/or rotation and may also include scaling, shearing, etc[72].

A problem with the sweep method is that it may result in surfaces which are non-homogeneous. This occurs when a curve is swept in a specific direction, resulting in dangling edges, Figure 2.40. Another example that produces a similar problem is the rotational sweep of a generator curve passing through the axis of rotation, producing a surface with a singularity which is usually unacceptable for architectural design.

[71] Mortenson, Geometric modeling

[72] Salomon, Computer graphics and geometric modelling

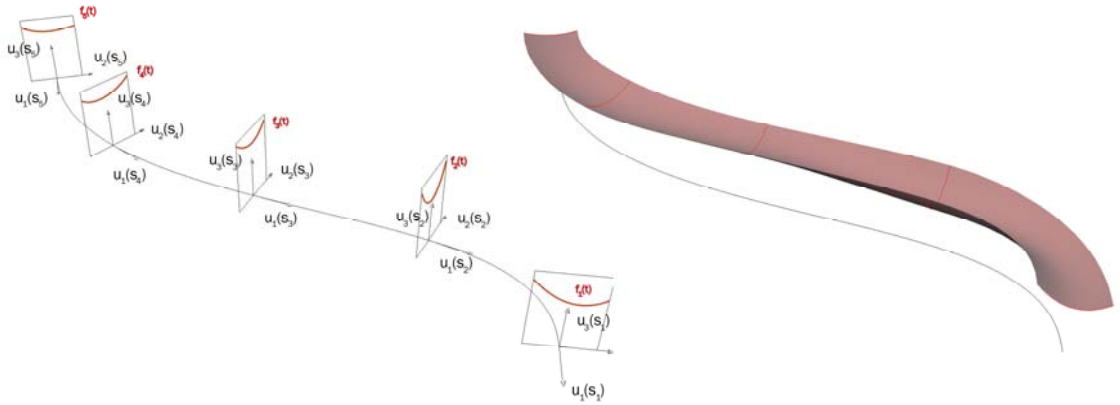


Fig. 2.39. Sweeping a curve $f(v)$ along a framed curve $\mathbf{c}(u)$ creating a sweep surface $\mathbf{S}(u,v)$

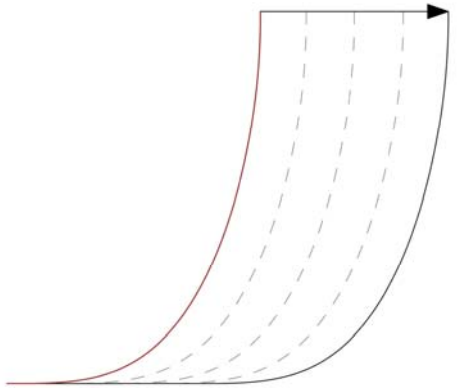


Fig. 2.40. Due to the coinciding direction of the sweep and a part of the curve, the surface becomes non-homogeneous

skinning and lofting

Skinning and lofting* are operations of constructing a surface that interpolates a number of user specified section curves, $\mathbf{C}_i(t)$[73]. Generally, these curves are sections in the v direction of the surface to be constructed, with the u direction as the blending direction[74]. There is an infinite number of surfaces passing through these curves. Although approximation across the section curves can be used, skinning methods usually interpolate through the $\mathbf{C}_i(t)$ with the result that these curves are the isoparametric curves on the resulting skinned surface. In order to generate a skinning or lofted surface the $i = N + 1$ curves must be linearly interpolated.

$$\mathbf{S}(u,v) = (1-v)\mathbf{C}_i(u) + v\mathbf{C}_{i+1}(u) \qquad 0 \le u,v \le 1 \qquad i = 0,1,2,\ldots,N\text{-}1 \qquad \text{(Eq. 2.7)}$$

* Sources in literature are contradictory in denoting the difference between skinning and lofting. Marsh states that some authors reserve the term lofting to mean a skinning operation where the interpolating surfaces satisfy specified derivative conditions along the section curves. Other sources, such as Piegl, state that skinning is simply a newer term for lofting. According to Davies, the latter terminology derives from the days when ships were built of wood. Since drawings had to be produced to scale, the attics (or lofts) of buildings were used to accommodate the large size drawings.

[73] Marsh, Applied geometry for computer graphics and CAD

[74] Piegl, The NURBS book

For two curves, with a fixed $t = t^*$, then $\mathbf{S}(u^*,v)$ is the line connecting the two points $\mathbf{C}_0(t^*)$ and $\mathbf{C}_1(t^*)$ and which passes through each point on this surface[75]. Consider the following examples:

example01
Two non-parallel straight curves in parametric representation,

$$\mathbf{C}_0(t) = (t, 0, 0)$$
$$\mathbf{C}_1(t) = (t, t, 1)$$

The surface (Figure 2.41, left) is then defined by

$$\mathbf{S}(u,v) = (1-v)\mathbf{C}_0(u) + v\mathbf{C}_1(u)$$
$$\mathbf{S}(u,v) = (u, uv, v)$$

example02 (after Salomon[76])
Two quadratic curves in parametric representation,

$$\mathbf{C}_0(t) = (2t-1, -2t(t-1), 0)$$
$$\mathbf{C}_1(t) = (2t-1, 4t(t-1), 1)$$

The surface (Figure 2.41, right) is then defined by

$$\mathbf{S}(u,v) = (2u-1, 2u(u-1)(1+v), v)$$

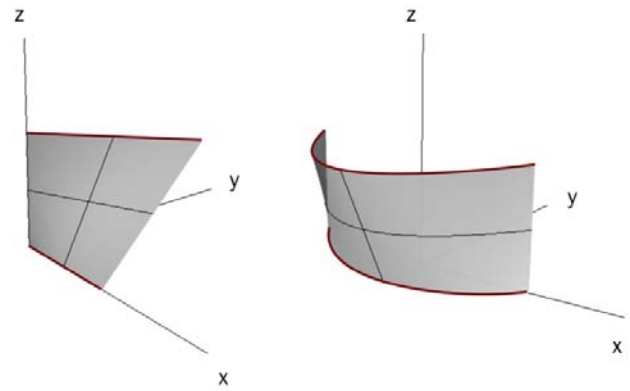


Fig. 2.41. Two skinned surfaces based on straight section curves (left) and quadratic section curves (right)

General skinning surfaces can be achieved by replacing the blending functions (1 - u) and u in Equation 2.7 by continuous functions $f_0(t)$ and $f_1(t)$ that satisfy $f_0(0) = f_1(1) = 1$ and $f_0(1) = f_1(0) = 0$.

As will be discussed in part02, developable surfaces are based on the same premises as skinned and lofted surfaces. However, since developable surfaces cannot be created from just any two curves, results can be unpredictable. In other words, the loft methods are sensitive to the makeup of the curves being lofted. In general, it is best if they are as simple as possible and have the same parameterisation.

[75] Goldman, Pyramid algorithms
[76] Salomon, Computer graphics and geometric modelling

2.4 epilogue: research in parametric modelling of developable surfaces

As mentioned before, many of the previously discussed rationalised surface definitions can be categorised in different rational surface classes. For instance, different parametric equations of surfaces of revolution, ruled surfaces (including extruded and developable surfaces) and translational surfaces can result in an open cylindrical surface. With respect to this and by incorporating certain prerequisites, developable surface may contain specific characteristics that follow from the definitions of surfaces of revolution, extruded surfaces and translational surfaces. Developable surfaces are also surfaces of revolution when a straight curve profile is rotated around the generatrix, resulting in cylindrical or conical surfaces depending on the direction of the curve profile. Also, when curve profiles are extruded or translated along a straight directrix, the resulting surface is by definition a developable, and therefore a ruled surface.

More importantly, developable surfaces contain a number of characteristics that make them well applicable for architectural and structural purposes, such as the possibility to generate smooth discrete surfaces that are aesthetically appealing and the advantages in relation to structural layouts and manufacturing. In particular, developable surfaces offer definitions of single-curvature geometries with an overall sculptural appearance in $\mathbb{R}^3$ space and are structured for constructability within specific design constraints. However, they remain difficult to model, particularly for non-expert users since using existing tools requires significant geometric expertise and time[77].

Therefore, the focus of this research lies on the parametric geometric description of developable surfaces, and how these descriptions can be used in architectural design. The next chapter deals with the general parametric and mathematic description of rational surface classes. Subsequently, the focus directs towards the description of developables and their parametric and mathematical description. As such these chapters form the base of the digital toolbox which will be discussed in part03.

[77] Rose, Developable surfaces from arbitrary sketched boundaries

3 parametric mathematical description of surface properties

The mathematical development of differential geometry and related general theories provide the needed theoretical basis for understanding surfaces properties. The mathematical formulations, however, usually do not find their way to designers in an explicit manner. Generally, digital design applications, such as Bentley's Microstation and McNeel's Rhinoceros account for the representation of complex geometric shapes, leaving the mathematical description hidden for the designer's eye. This chapter deals with elementary differential geometry and mathematical descriptions which support the parametric design of the afore presented surface classes.

3.1 introduction: general aspects of parametric description

Although the design process is generally considered to be cyclical, the process of developing a conceptual design into a tangible structure involves specific sequential steps that need to be taken. When it is not possible to record these steps taken in the design process, revision of a design can become an arduous task. Digital design environments with hierarchically based structures have been developed that allow the design history to be recorded and listed in relation with parameters and associations. And since there is a general need to be able to quickly alter certain design characteristics of a shape, from basic dimensions to a more detailed level, parametric and associative computational methods are increasingly used to reduce time spent on remodelling a design. From this point of view, computation can be seen as a process of reducing labour intensive, repetitive processes into relatively simple expressions of code.

However, computational design is more than a time-saving instance in a design process. In the current practice, design is increasingly about showing logic, clarity of process and creating purpose and flexibility[78].

A family of parametric variations all stem from the same characteristic shape but vary in dimension or shape from one to another; they are all instances of the same design[79], Figure 3.1. Based on these descriptions, computation is a linking factor between parametric associative modelling and performance based architecture. Or put in the words of Branko Kolarevic;

> **'One of the most profound aspects of contemporary architecture is [...] the new found ability to generate construction information directly from design information through the new processes of digital design and production'**[80]

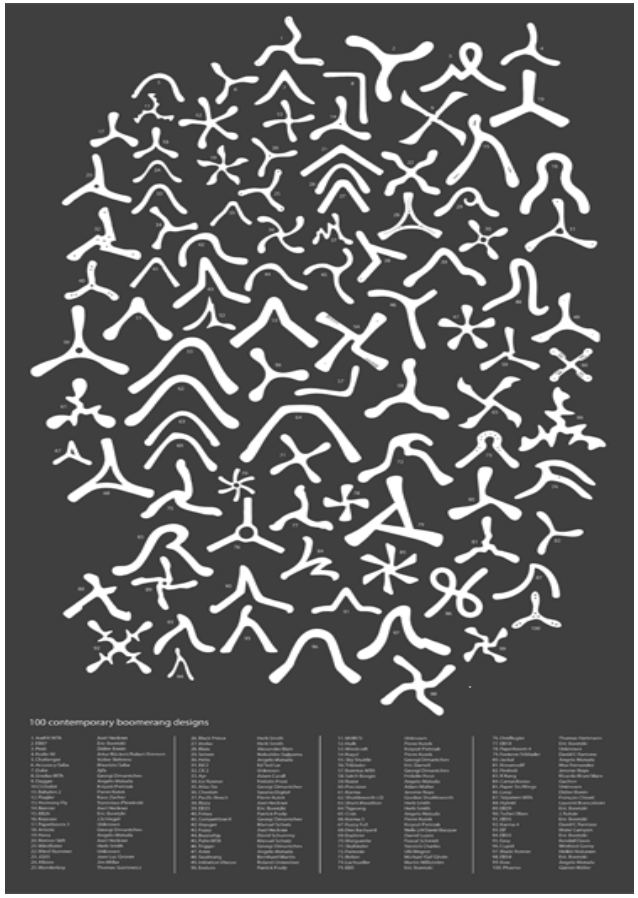


Fig. 3.1. 100 Contemporary boomerang designs [http://www.seblester.co.uk, November 2009]

[78] Meada, Design by Numbers

[79] Schodek, Digital design and manufacturing: CAD/CAM applications in architecture and design

[80] Kolarevic, Architecture in the Digital Age

In every design process, changes in the constraints or the design variables resolve in a new set of possible solutions which can be translated back to the design. If the design process is set up with parametric and associative constraints and variables, the design can benefit from an exploration of the design space while maintaining the inherent logic. In general, not the design of a specific shape is generated with parametric and associative modelling, but a set of principles encoded as a sequence of parametric equations and associations. As a result, multiple variations of a design can be presented when parameters and associative connections are varied, which can subsequently be examined on performance, where the performance gives feedback between the design and the systems it is embedded in and focuses on what the process is able to generate, Figure 3.2.

Greg Lynn states in Animate Form that it is important for any parameter-based design that there be both the unfolding of an internal system and the infolding of contextual information fields[81]. If these contextual information fields do not provide the appropriate conditions for a parametric and associative model or tool development, when relations of parameters are not well defined or when a design model contains too little or too many parameters, making use of parametric and associative modelling can lead to an inefficient design process.

To conclude this, during this research project the goals are set out to describe the geometric definition of rationalised surfaces parametrically, which also implies that the context in which these surfaces are generated need to be defined parametrically as well. In the tool development presented later in this research, the control elements as described in Section 2.2.3 form the conditional parameters and so are the surface properties, such as curvature, continuity and offsetting possibilities. These context parameters should allow for the definition of rationalised surfaces which subsequently can be examined for their architectural performance in relation to the initial design intentions.

Before going into detail concerning the parametric descriptions of the surface properties and surface classes, the next subsection introduces general aspects of parametric descriptions.

Fig. 3.2. Performance based design leading to different designs for different goals [images from various sites]

[81] Lynn, Animate Form

parametric equation forms

Defining geometry of for instance space curves or space surfaces can be based on different descriptions. Space curves are one-dimensional objects in three dimensions which can be described mathematically either in Cartesian form, by two equations, for example for $(x,y,z) \in \mathbb{R}^3$

$$y = x^2, xz = y^2 \qquad \text{(Eq. 3.1)}$$

$$x^2 + y^2 = a^2, \tan\frac{z}{b} = \frac{y}{x} \qquad \text{(Eq. 3.2)}$$

or by parameterisations, for example, for $t \in \mathbb{R}^1$

$$\mathbf{C}(t) = (x(t), y(t), z(t)) = (t, t^2, t^3) \qquad \text{(Eq. 3.3)}$$

$$\mathbf{C}(t) = (x(t), y(t), z(t)) = (a\cos t, a\sin t, bt) \qquad \text{(Eq. 3.4)}$$

where Equation 3.4 describes a spatial helix curve.

The Cartesian description simply gives a shape in space while the parametric description also includes a direction and a rate of evolution as t increases[82].

Obviously, also space surfaces – two dimensional objects in three-dimensional space – can be described mathematically in either Cartesian form or parametric form. In Cartesian form, the surface is described by a single equation in three space variables $(x,y,z) \in \mathbb{R}^3$ in either an explicit form ($z = f(x,y)$) or an implicit form ($f(x,y,z)=0$). In the parametric form, surfaces are described by three equations for the coordinates (x,y,z) in terms of two parameters (u,v).

As an example, in parametric form, a standard helicoid is represented by

$$\mathbf{S}(u,v) = (x(u,v), y(u,v), z(u,v)) = (v\cos u, v\sin u, bu) \qquad \text{(Eq. 3.5)}$$

[82] Davies, An introduction to computational geometry for curves and surfaces

The $x(u,v)$ and $y(u,v)$ coordinate functions describe the planar circular curve, where the domain of v denotes the width of the strip that is swept with a helical motion*. In other words, the lower and upper boundary of the domain of v determine the radii of the inner and outer helix. For every point between these boundary curves, there is a helix contained in the helicoid which passes through that point.

Figure 3.3 shows a helicoid generated with the following parametric domains:

u = [0,4π]
v = [5,20]
b = 2.5

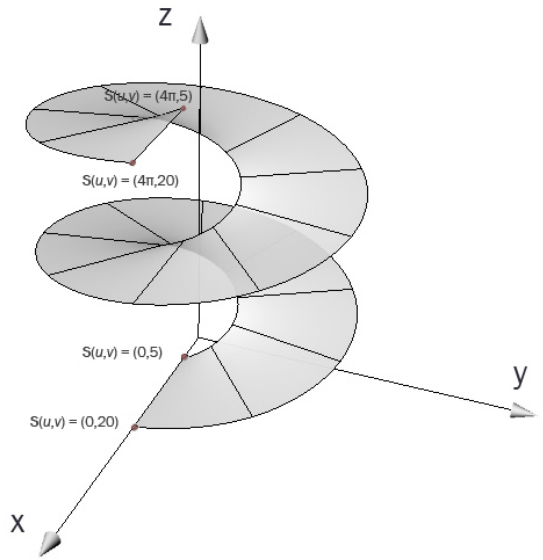


Fig. 3.3. Parametrically defined helicoid

parametric curve and surface descriptions

Following the description in the example of the helicoid, the following representation and definitions for curves and surfaces are used throughout the thesis:

parametric curves

$\mathbf{C}(t) = (x(t),y(t),z(t))$	parametric curve
$\mathbf{C}(t)$	parameterisation of **C**
$x(t)$, $y(t)$ and $z(t)$	coordinate functions
t	curve parameter (or variable)
$\mathbf{N_C}(t)$	normal vector to **C**
$\mathbf{T_C}(t)$	tangent vector to **C**

parametric surfaces

$\mathbf{S}(u,v) = (x(u,v),y(u,v),z(u,v))$	parametric surface
$\mathbf{S}(u,v)$	parameterisation of **S**
$x(u,v),y(u,v),z(u,v)$	coordinate functions
u, v	surface parameters (or variables)
$\mathbf{N_S}(u,v)$	normal vector to **S**
$\mathbf{T_S}(u,v)$	tangent vector to **S**

*For more exiting shapes, the parametric representation of what is called the generating curve $\mathbf{C}(t) = (x(t),y(t),z(t)) = (t,0,0)$ for the general form can be changed to for instance a circular shape with $\mathbf{C}(t) = (x(t),y(t),z(t)) = (\cos t,0,\sin t)$ for generating tube like screw surfaces.

3.2 parametric description of surface properties

3.2.1 parametric equations of surface properties

One advantage of parametrically defined geometry is that local properties, such as direction and normals can be derived as long as their mathematical formulations can be derived. Based on differential geometry, which is concerned with those properties of surfaces which depend on their behaviour in a neighbourhood of a point, the parametric descriptions of the surface normal, tangent plane and other surface properties at a point can be given. These properties are generally very useful for geometric analysis and modelling purposes.

surface normal and tangent plane

First, consider a planar curve $\mathbf{C}(t) = (x(t),y(t))$ and suppose P and Q are points with coordinates $(x(t),y(t))$ and $(x(t+dt),y(t+dt))$[83], Figure 3.4. When $dt \to 0$, the following equation will give the tangent vector of the curve; $\mathbf{T}(t) = \mathbf{C}'(t) = (x'(t), y'(t))$. The unit tangent vector is defined as

$$\hat{\mathbf{T}}(t) = \left(\frac{x'(t)}{\sqrt{x'(t)^2 + y'(t)^2}}, \frac{y'(t)}{\sqrt{x'(t)^2 + y'(t)^2}} \right) \qquad \text{(Eq. 3.6)}$$

If the tangent vector is rotated over an angle of π/2 (in an anticlockwise direction in Figure 3.4), then the normal vector $\mathbf{N}(t) = (-y'(t), x'(t))$. The unit normal vector for a curve is defined as

$$\hat{\mathbf{N}}(t) = \left(\frac{-y'(t)}{\sqrt{x'(t)^2 + y'(t)^2}}, \frac{x'(t)}{\sqrt{x'(t)^2 + y'(t)^2}} \right) \qquad \text{(Eq. 3.7)}$$

A tangent line to a curve upon a surface is called a tangent line to a regular surface at the point of contact*. It is evident that there are an infinite number of tangent lines to a surface at a point. All of these lines lie in a plane, which is called the tangent plane of the surface at the point[84]. The tangent plane can be defined by the parametric tangent vectors in *u* and *v* direction at the point under consideration, Figure 3.5. These

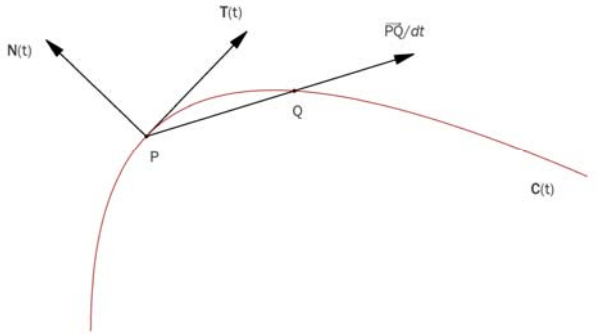


Fig. 3.4. Tangent and normal to a parametric curve

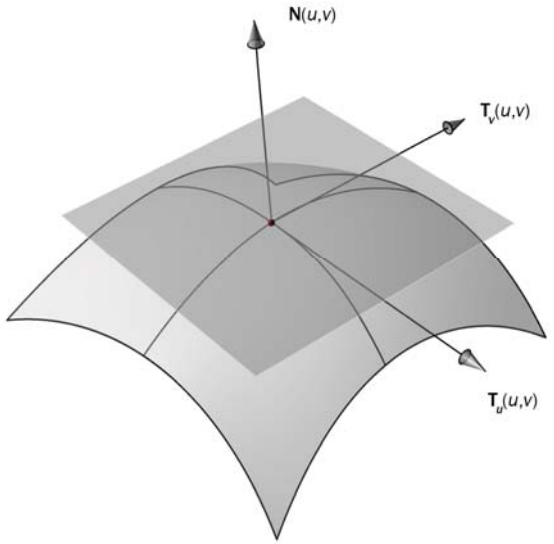


Fig. 3.5. Tangent plane and normal to a parametric surface

[83] Marsh, Applied geometry for computer graphics and CAD

*Roughly speaking, a regular surface is obtained by taking pieces of a plane, deforming them, and arranging them in such a way that the resulting figure has no sharp points, edges or self-intersections and so that it makes sense to speak of a tangent plane at points of the figure [Do Carmo, Differential geometry of curves and surfaces].

[84] Eisenhart, A treatise on the differential geometry of curves and surfaces

tangent vectors can be derived by taking the first partial derivative of the surface definition for respectively (u,0) and 0,v) where the cross product of these vectors is a normal vector to the tangent plane.

$$\mathbf{N}(u,v) = \mathbf{T}_u(u,0) \times \mathbf{T}_v(0,v) \qquad \text{(Eq. 3.8)}$$

As will be discussed in detail later, tangent planes and surface normal vectors play an important role in the modelling and analysis of developable surfaces. Related to that, also the curvature description plays an important role in the analysis of developable surfaces.

surface curvature

In literature, the definition of the curvature of surfaces is frequently explained by introducing the definition of the curvature of planar curves. Multiple methods to derive and approximate the curvature of 2D curves can be adopted. Deriving the tangent angle at specific points and determining the radius of the osculating circle are two of them. A third method follows from calculating the derivatives of the curve. When a parametric equation of a curve is given with curve $\mathbf{C}(t) = (x(t),y(t))$, the signed curvature is

$$\kappa(t) = \frac{x'(t)y''(t) - y'(t)x''(t)}{\left(x'(t)^2 + y'(t)^2\right)^{3/2}} \qquad \text{(Eq. 3.9)}$$

For a space curve, the curvature is described with the following equation

$$\kappa(t) = \frac{\sqrt{\left(y'(t)z''(t) - z'(t)y''(t)\right)^2 + \left(z'(t)x''(t) - x'(t)z''(t)\right)^2 + \left(x'(t)y''(t) - y'(t)z''(t)\right)^2}}{\left(x'(t)^2 + y'(t)^2 + z'(t)^2\right)^{3/2}} \qquad \text{(Eq. 3.10)}$$

When a space curve lies on a surface embedded in three dimensions, different measures of the surface curvature, taking into account the surface's normal vector, are available. These are the geodesic curvature, geodesic torsion and normal curvature. The geodesic curvature, κ_g, is the curvature of the curve projected onto the surface's tangent plane. The geodesic torsion (or relative torsion), τ_r, measures the rate of change of the surface normal around the curve's tangent[85] or in other words it describes how a surface twists about a curve on the surface. However, when talking about the curvature of a surface, usually the normal curvature is meant. To explain the normal curvature of a surface, consider a point on the surface. The tangent vector to the surface in an arbitrary direction, i.e. $\mathbf{T}(u,v)$, together with the normal, i.e. $\mathbf{N}(u,v)$,

[85] Wikipedia, Curvature

at that point of the surface define a normal plane of the surface at that same point. The normal curvature in the direction of the tangent vector, κ_n, is the curvature at the point under consideration of the plane curve defined by the intersection of the normal plane and the surface. Regarding all possible tangent vectors, then the maximum and minimum values of the normal curvature at a point are called the principal curvatures, κ_1 and κ_2[86], Figure 3.6. Based on these principal curvatures, the Gaussian curvature can be determined since it is equal to the product of the principal curvatures, $\kappa_1\kappa_2$. Determining the direction and value of the principal curvatures is primarily based on derivatives of surface curves and the surface itself.

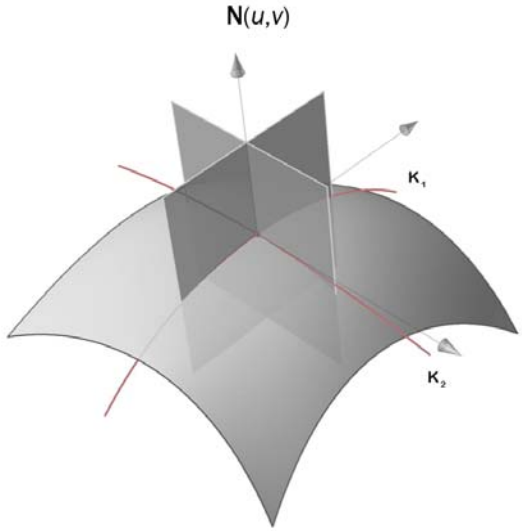


Fig. 3.6. Normal planes containing the curve tangents in the principal directions and the normal to the surface at a specific point

example

Consider a hyperbolic paraboloid. The surface is generically described by the following parametric equation

$$\mathbf{S}(u,v) = (u,v,uv) \qquad \text{(Eq. 3.11)}$$

When additional variables for shape distortion are added, the parameterisation becomes

$$\mathbf{S}(u,v) = (au,bv,cuv) \qquad \text{(Eq. 3.12)}$$

Following from this equation, the partial derivatives of equation can be derived

$$\frac{\partial \mathbf{S}}{\partial u} = \begin{pmatrix} a \\ 0 \\ cv \end{pmatrix} \qquad \frac{\partial \mathbf{S}}{\partial v} = \begin{pmatrix} 0 \\ b \\ cu \end{pmatrix}$$

$$\frac{\partial^2 \mathbf{S}}{\partial u^2} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \qquad \frac{\partial^2 \mathbf{S}}{\partial v^2} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\frac{\partial^2 \mathbf{S}}{\partial u \partial v} = \begin{pmatrix} 0 \\ 0 \\ c \end{pmatrix}$$

[86] Wikipedia, Curvature

In order to derive the curvature of the surface the first fundamental form of the surface needs to be divided by the second. These expressions are respectively

$$ds^2 = E(du)^2 + 2Fdudv + G(dv)^2 \quad \text{(Eq. 3.13)}$$

$$\kappa \cos\varphi ds^2 = L(du)^2 + 2Mdudv + N(dv)^2 \quad \text{(Eq. 3.14)}$$

from which follows, with $\varphi = 0$, the angle between the principal normal a curve on the surface and the normal of the surface

$$\kappa = \frac{L(du)^2 + 2Mdudv + N(dv)^2}{E(du)^2 + 2Fdudv + G(dv)^2} \quad \text{(Eq. 3.15)}$$

To solve this equation, the coefficients of the first and second fundamental forms need to be derived. For a hyperbolic paraboloid, the coefficients of the first fundamental form are given by

$$E(u,v) = \frac{\partial \mathbf{S}}{\partial u} \cdot \frac{\partial \mathbf{S}}{\partial u} = a^2 + 0 + (cv)^2 \quad \text{(Eq. 3.16)}$$

$$F(u,v) = \frac{\partial \mathbf{S}}{\partial u} \cdot \frac{\partial \mathbf{S}}{\partial v} = 0 + 0 + c^2uv \quad \text{(Eq. 3.17)}$$

$$G(u,v) = \frac{\partial \mathbf{S}}{\partial v} \cdot \frac{\partial \mathbf{S}}{\partial v} = 0 + b^2 + (cu)^2 \quad \text{(Eq. 3.18)}$$

The coefficients of the second fundamental form implement the unit normal vector, which can be written as

$$\mathbf{N}(u,v) = \frac{\partial \mathbf{S}}{\partial u} \times \frac{\partial \mathbf{S}}{\partial v} = \begin{pmatrix} -(cb)v \\ -(ac)u \\ ab \end{pmatrix} \quad \text{(Eq. 3.19)}$$

$$\hat{\mathbf{N}}(u,v) = \frac{1}{\sqrt{(cb)^2v^2 + (ac)^2u^2 + (ab)^2}} \begin{pmatrix} -(cb)v \\ -(ac)u \\ ab \end{pmatrix} \quad \text{(Eq. 3.20)}$$

The coefficients of the second fundamental form for the hyperbolic paraboloid are then given by

$$L(u,v) = \hat{\mathbf{N}}(u,v) \cdot \frac{\partial^2 \mathbf{S}}{\partial u^2} = 0 \qquad \text{(Eq. 3.21)}$$

$$M(u,v) = \hat{\mathbf{N}}(u,v) \cdot \frac{\partial^2 \mathbf{S}}{\partial u \partial v} = \frac{abc}{\sqrt{(cb)^2 v^2 + (ac)^2 u^2 + (ab)^2}} \qquad \text{(Eq. 3.22)}$$

$$N(u,v) = \hat{\mathbf{N}}(u,v) \cdot \frac{\partial^2 \mathbf{S}}{\partial v^2} = 0 \qquad \text{(Eq. 3.23)}$$

The principal curvatures, following from Equation 3.15, are the roots of

$$\det \begin{pmatrix} \kappa E - L & \kappa F - M \\ \kappa F - M & \kappa G - N \end{pmatrix} = 0 \qquad \text{(Eq. 3.24)}$$

When κ is a polynomial of degree 2, then

$$(\kappa - \kappa_1)(\kappa - \kappa_2) = 0 \qquad \text{(Eq. 3.25)}$$

can be used to compute the values of the principal curvatures, from which follows the equation to calculate the Gaussian curvature

$$\kappa_1 \kappa_2 = \frac{LN - M^2}{EG - F^2} \qquad \text{(Eq. 3.26)}$$

For a hyperbolic paraboloid without variable values, this reads

$$\kappa_1 \kappa_2 = \frac{-1}{(v^2 + u^2 + 1)^2} \qquad \text{(Eq. 3.27)}$$

For the hyperbolic paraboloid with variable values, it follows that

$$\kappa_1\kappa_2 = \frac{LN - M^2}{EG - F^2} = \frac{0 - \dfrac{(abc)^2}{(cb)^2v^2 + (ac)^2u^2 + (ab)^2}}{(a^2 + (cv)^2)x(b^2 + (cu)^2) - (c^2uv)^2}$$

$$\kappa_1\kappa_2 = \frac{-\dfrac{(abc)^2}{(cb)^2v^2 + (ac)^2u^2 + (ab)^2}}{(cb)^2v^2 + (ac)^2u^2 + (ab)^2}$$

$$\kappa_1\kappa_2 = \frac{-(abc)^2}{((cb)^2v^2 + (ac)^2u^2 + (ab)^2)^2}$$

The equations are based on the descriptions of Hardy and Steeb[87].

Given the shape variables $\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} 1.1 \\ 1.6 \\ 0.9 \end{pmatrix}$, the hyperbolic paraboloid has a Gaussian curvature at (u,v) = (0.5,0.25) of -0.208. The same result is given by Rhino's Gaussian curvature analysis, Figure 3.7.

Over the whole surface, the Gaussian curvature has a negative sign – an intrinsic property of a ruled surface –, indicating that κ_1 and κ_2 are of opposite sign. A point on the surface is then called a hyperbolic point of the surface, also known as a saddle point. When the principal curvatures of a surface are both positive and negative, then points on the surface are elliptic points. For surfaces with vanishing Gaussian curvature, i.e. one of the principal curvatures is zero, points are called parabolic points. At a parabolic point, the Gaussian curvature is zero, but one of the principal curvatures is not zero. In planar points all principal curvatures are zero[88].

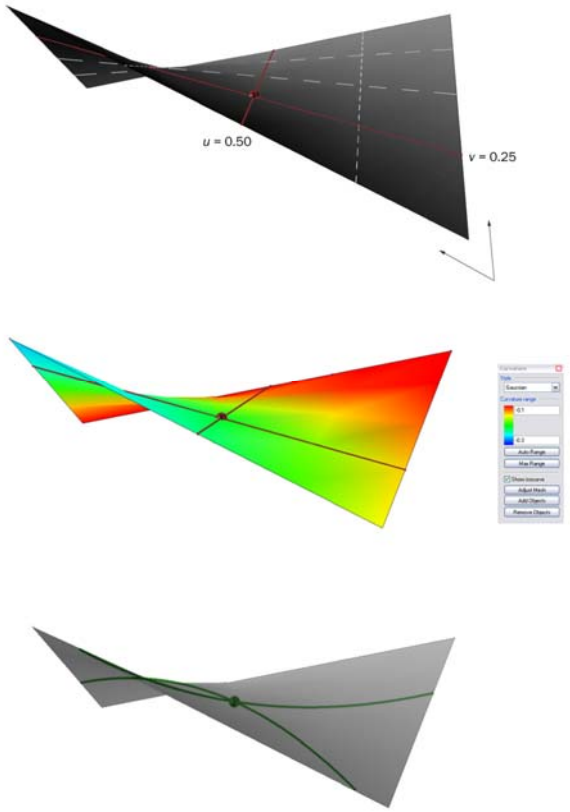


Fig. 3.7. Curvature analysis of a hyperbolic paraboloid; a double ruled surface. Following from mathematical equations the Gaussian curvature at (u,v) = (0.5,0.25) is -0.21. Below, the principal curvature directions are given.

[87] Hardy, Mathematical tools in computer graphics with C# implementations

[88] Do Carmo, Differential geometry of curves and surfaces

Most surfaces are not composed entirely of one type of Gaussian curvature[89]. For example, a torus contains three types of Gaussian curvature. The parabolic points are on two circles in two horizontal planes that touch the top and the bottom of the torus. The elliptic points are on the outside part of the torus with normal facing outward, delimited by the two parabolic circles. The hyperbolic points are on the inside part of the torus with normal facing inward[90], Figure 3.8.

As mentioned before, the distinction between surfaces of zero and non-zero Gaussian curvature is of importance, especially in relation to designing with developable surfaces. In Section 4.2, it is shown how the Gaussian map can be used to denote the curvature of surfaces.

continuity between surfaces

As discussed before, surface continuity denotes how surfaces meet at a shared edge. In order to maintain both positional and tangential continuity while deformed, the surfaces should have matching parameterisation along the shared edge. So the issue is how to modify the edge where two surfaces meet without breaking their continuity. If one selects a minimum of four vertices between the two surfaces, specifically the two points where they touch, and one vertex in from the edge on each surface, as long as those four points are transformed as a unit, the edge will not lose its tangential continuity at the location of the connection[91]. In other words, between two surfaces whose edges meet, there exists G^0 continuity by definition and this can be extrapolated by moving inward from the shared edge to the surfaces. If the control points on the second control lattice away from the ones that just met are co-planar, then there exists G^1 continuity, etcetera. As mentioned before, since the scale of buildings is relatively large, usually G^1 continuity is found to be adequate in architectural design.

Analytically deriving the geometric continuity at points where parametrically defined surfaces meet is difficult. However, the following applies based on the text above; geometrically checking for G^1 continuity between surfaces in a 3D modelling environment at a point **P**, they have to have the same normal directions and hence the same tangent planes. The converse is also true: if two surfaces meet at a point **P** and have common surface normals at **P**, then the surfaces meet with G^1 continuity[92].

Due to the complexity in surface continuity, especially in relation to ruled and developable surfaces, issues may be resolved geometrically. This will be exemplified in Chapter 6.

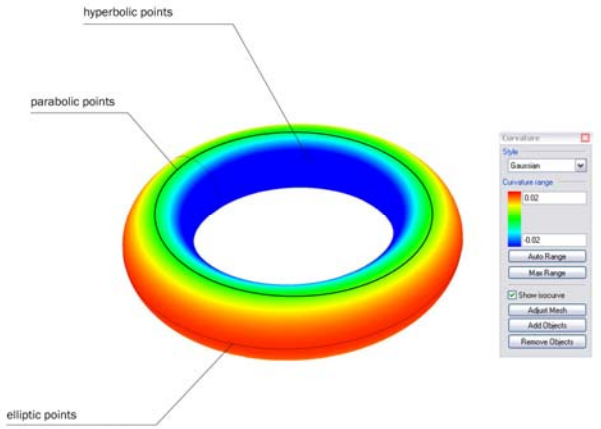


Fig. 3.8. Curvature analysis of a torus, showing the three types of Gaussian curvature classes

[89] Farin, The essentials of CAGD

[90] Penn Engineering, *CIS700* – emerging technologies

[91] The Gnomon Workshop, Surface continuity tip

[92] Marsh, Applied geometry for computer graphics and CAD

offsetting properties

To illustrate the definition of offset surfaces, an analogy to the definition of an offset curve is usually presented. For every planar curve $\mathbf{C}(t)$ there are two dual curves at a distance d from this curve. Mathematically, these offset curves are computed as follows. Let $\mathbf{C}(t) = (x(t),y(t))$ be a parametric representation of a planar curve. The unit normal vector field $\mathbf{N}(t)$ and its offset curve $\mathbf{C}_d(t)$ are given by

$$\mathbf{N}(t) = \left(\frac{y'(t)}{\sqrt{x'(t)^2 + y'(t)^2}}, \frac{-x'(t)}{\sqrt{x'(t)^2 + y'(t)^2}}\right) \qquad \text{(Eq. 3.28)}$$

and

$$\mathbf{C}_d(t) = \mathbf{C}(t) \pm d \cdot \mathbf{N}(t) \qquad \text{(Eq. 3.29)}$$

Because of the square root in the denominator of Equation 3.28, a polynomial or rational parameterisation of $\mathbf{C}(t)$ will in general not yield to a polynomial or rational parameterisation of the offset. Consider the parabolic curve $\mathbf{C}(t) = (x(t),y(t)) = (t,t^2)$. The derivatives, needed to calculate the normal vector field $\mathbf{N}(t)$ are $(x'(t),y'(t)) = (1,2t)$. Hence, $\mathbf{N}(t) = \left(\frac{2t}{\sqrt{4t^2+1}}, \frac{-1}{\sqrt{4t^2+1}}\right)$, and the offset at a distance d is

$$\mathbf{C}_d(t) = \left(t + d\frac{2t}{\sqrt{4t^2+1}}, t^2 + d\frac{-1}{\sqrt{4t^2+1}}\right) \qquad \text{(Eq. 3.30)}$$

which shows that offset curve is not a parabola, although the original curve is. This is visualised in Figure 3.9 where two offsets are given, d and $-2d$.

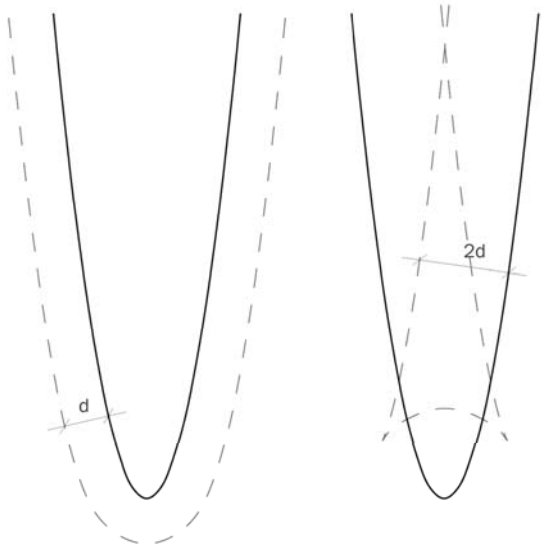


Fig. 3.9. Offsets of a parabola at offset distance d = d (left) and d = -2d (right)

Pottmann defines an offset of a smooth surface as follows; on each surface normal, two points are marked that are at a constant distance d from the surface $\mathbf{S}(u,v)$. The set of all these points forms the offset surface $\mathbf{S}_d$. The original surface and the offset surface share their surface normals and their tangent planes in corresponding points are parallel[93]. Analogous to the planar curve case, the offset surface can be calculated by

$$\mathbf{S}_d(t) = \mathbf{S}(u,v) \pm d \cdot \mathbf{N}(u,v) \qquad \text{(Eq. 3.31)}$$

[93] Pottmann, Architectural geometry

3.2.2 isometric description of surfaces

As mentioned before, a curve is a mapping from an interval – usually t = [0,1] – to its image in three dimensional space. The parameter t is frequently considered analogous to time. The first derivatives of the mapping functions at particular values of t may then be considered at the velocity vector at t. Obviously, geometrically, this derivative with respect to t is the tangent of the space curve. The length or magnitude of the velocity vector denotes the curvature. To explain this, the parameter t of curves – the equivalent of the u and v parameter for surfaces – is considered*.

Firstly, reference is made to the work of Pierre Bézier who started with the principle that any point **P**(t) on a curve segment must be given by a parametric function of the following form

$$\mathbf{P}(t) = \sum_{i=0}^{n} \mathbf{P}_i \mathbf{f}_i(t) \qquad t \in [0,1] \qquad \text{(Eq. 3.32)}$$

where the vectors $\mathbf{P}_i$ represent the n+1 vertices of a characteristic polygon, where n is the degree of the Bézier curve. These vertices are the control points. $\mathbf{f}_i$ are the basis functions describing the Bézier curve.

$$\mathbf{f}_i(t) = B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i} \qquad \text{(Eq. 3.33)}$$

with the binomial coefficient function

$$\binom{n}{i} = \frac{n!}{i!(n-i)!} \qquad \text{(Eq. 3.34)}$$

The open polygon formed by the control points in Figure 3.10 serves two functions: it establishes the initial shape of the curve and then furnishes a framework for altering the curve. For any Bézier curve, each control point $\mathbf{P}_i$ is 'weighted' by its associated basis function. When t = 0, $\mathbf{P}_0$ is given a weight of 1.0, and $\mathbf{P}_1$ through $\mathbf{P}_n$ a weight of zero. Less weight is given to $\mathbf{P}_0$ and more to each succeeding $\mathbf{P}_i$ as t increases, reaching a maximum weight for each $\mathbf{P}_i$ when t becomes i/n. Then all other weights decay gradually to 0 as the weight of $\mathbf{P}_n$ reaches 1 when t = 1. In other words, a shift occurs in the influence of each point (each

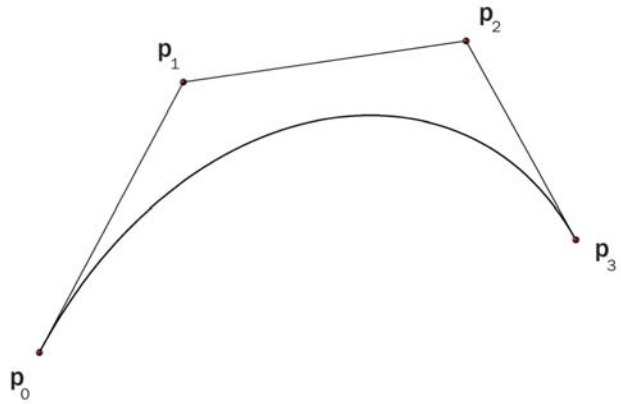


Fig. 3.10. A Bézier curve with four control points

* This text is partially copied from the final report 'architectural designing based on analytical NURBS modelling' – part01 by the author for the AR0055 Mediated Discourse course under the supervision of dr. ir. R.M.F. Stouffs. For more information on geometric modelling with Bézier, B-Splines and NURBS curves, reference is made to this report.

polygon vertex) as the parametric variable moves through its range from 0 to 1. This implies that changing the position of control point $\mathbf{P}_i$ has the greatest influence on the curve's shape at and near the parameter value i/n. It is therefore important to consider the distinction between the parametric, functionally derived qualities of a curve on one hand (i.e. for example the time and speed analogy) and the purely spatial properties on the other hand[94]. With this in mind, it must be noted that by way of progression to discussing the purely geometric properties of a curve, it needs to be considered that specific parameterisation of the curve does not uniquely describe the curve in space. Rather, there are a potentially unlimited number of parameterisations which can describe the same path through space. Exemplifying this with the time analogy, a given path in space may be traversed by any number of speeds and variations of speeds along the path. In other words, the specific choice of a parametric space and mapping function has a large implication on the 'experience' of the curve.

The influence of the control points is best supported by the geometric construction of a Bézier curve. The curve can be created as the locus of points produced by a simple recursive geometric construction derived from the work by De Casteljau, which basically corresponds to the parametric function of the Bézier curve.

The construction of a second degree Bézier curve is as follows[95]; Given any three points A, B, and C, the sides *AB* and *BC* of an open polygon can be drawn. For successive values of *t* in the closed interval [0,1] on these lines, points *D* and *E* can be constructed so that *AD/AB* = *BE/BC* = *t*. Finally, on *DE* point *F* can be constructed so that *DF/DE* = *t*. *F* is then the point that describes the Bézier curve at the *t* value – see Figure 3.11, top. This process can be applied to find Bézier curves of other degrees (see Figure 3.11, middle and bottom for n=3 and n=4 respectively) and to curves of which control points are not constrained to a plane.

In relation to the above, an isoparametric curve (isocurve) can be described as a curve of constant *u*- or *v*-value on a surface[96]. When two parametric curves with the same parametric domain $t \in [0,1]$ are lofted, the direction of one family of isoparametric curves is determined by the vector between the points on the curves with the same parameter *t*. These isoparametric curves define the rulings of the ruled surface. For surfaces, usually the parameter *t* is related to the *u* parameter. Therefore, in the case of the surface in Figure 3.12, the rulings of the lofted surface are formed by isoparametric curves of constant *u* value.

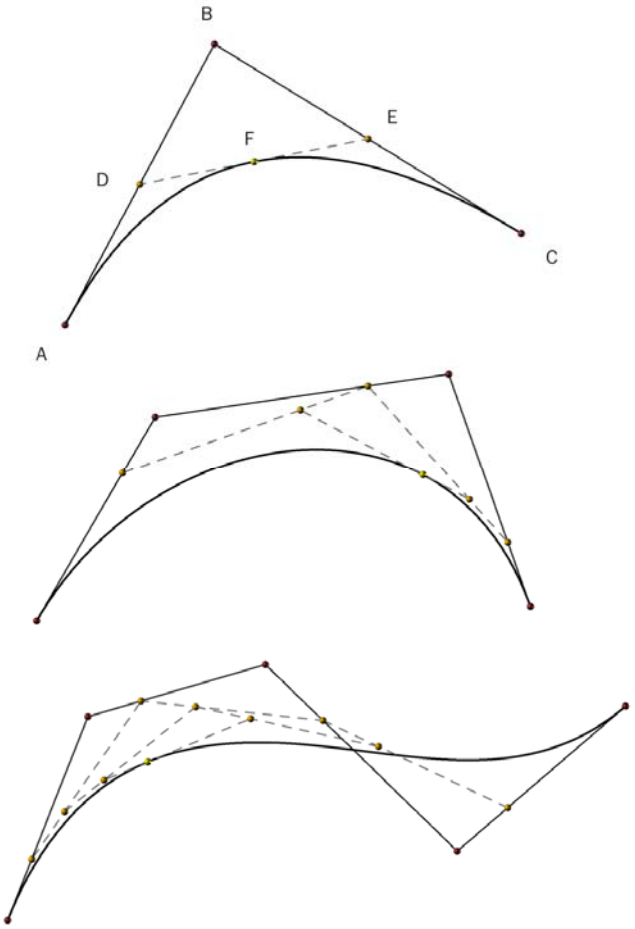


Fig. 3.11. Quadratic, cubic, and quartic Bézier curves with the geometrically constructed interpolation points for respectively $t = 0.5$, $t = 0.725$ and $t = 0.3$

[94] Shelden, Digital surface representation and the constructability of Gehry's architecture
[95] Mortenson, Geometric Modeling
[96] Rhino3D, Glossary

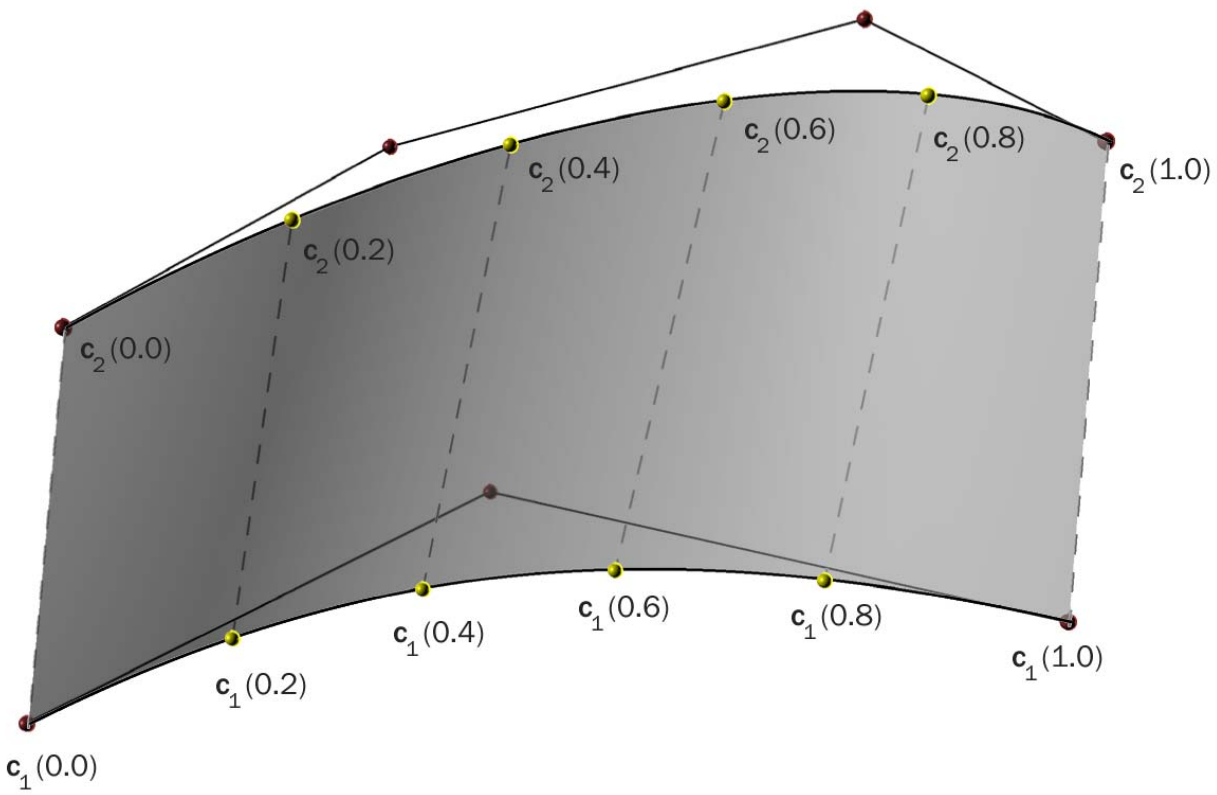


Fig. 3.12. Lofted surface over two Bézier curves of different degree, but with a corresponding parametric domain

Subsequently, a surface in three dimensional space is defined through a mapping from $\mathbb{R}^2$ coordinate parameters to $\mathbb{R}^3$ coordinates. By convention, this is usually from *u* and *v* to *x, y, z*. The $\mathbb{R}^2$ specification is referred to as its parametric space definition, and its mapping into $\mathbb{R}^3$ as a surface in world space.

3.3 epilogue: theoretic parametric surfaces

The intrinsic equations and properties of a surface are subject of differential geometry are relatively complex; fundamental forms and derivations are generally left for theory books dealing with the topic of differential geometry of curves and surfaces. However, from the given parameterisation, surface properties, such as the mean and Gaussian curvature, can be computed. Therefore, in their general form, surfaces are presented in a parametric form. Also, based on parametric equations, theoretical surfaces that are self-intersecting or non-oriented can be generated, whereas this might not be possible with implicit non-parametric descriptions. Take for instance the Klein bottle, which can be describe parametrically as

$$
\begin{aligned}
x &= \cos u\left[\cos\left(\frac{1}{2}u\right)\left(\sqrt{2}+\cos v\right)+\sin\left(\frac{1}{2}u\right)\sin v\cos v\right] \\
y &= \sin u\left[\cos\left(\frac{1}{2}u\right)\left(\sqrt{2}+\cos v\right)+\sin\left(\frac{1}{2}u\right)\sin v\cos v\right] \\
z &= -\sin\left(\frac{1}{2}u\right)\left(\sqrt{2}+\cos v\right)+\cos\left(\frac{1}{2}u\right)\sin v\cos v
\end{aligned}
\qquad \text{(Eq. 3.35)}
$$

but physically can only be realised in four dimensions, since it must pass through itself without the presence of a hole[97].

This chapter provided a brief onset to the description of parametric properties of surfaces by introducing a general theoretical discourse in surface properties. The next chapter deals with parametric description of developable surfaces, also to provide a common base to start working from. After Chapter 4, the switch is made towards a more practical approach by means of discrete differential geometry, introducing vector mathematics in order to compute rulings used to generate developable surfaces.

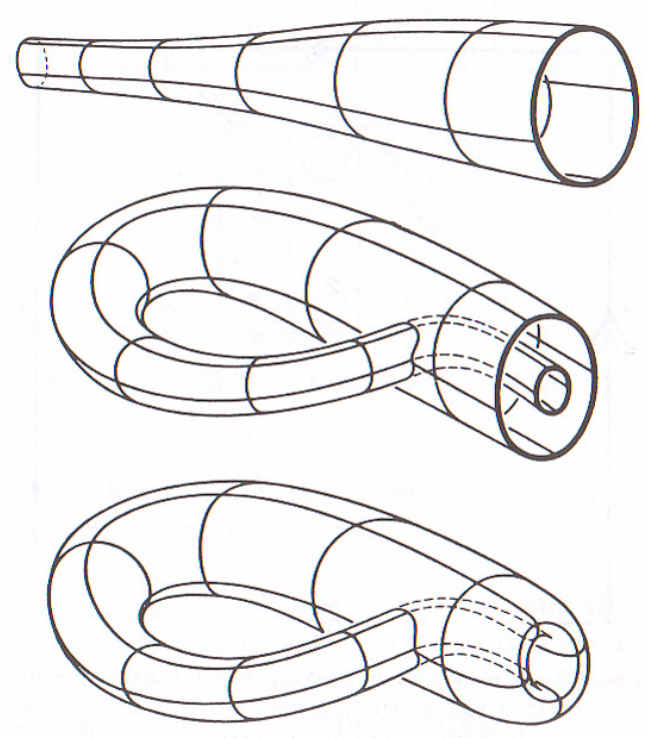


Fig. 3.13. Gluing up a Klein bottle [http://www.arch.columbia.edu, January 2011]

[97] Wolfram Mathworld, Klein bottle

part02 – digital developables

4 analytical characteristics of developable surfaces

As with all the rational surface classes, developable surfaces embed specific geometric characteristics. The possibilities in architectural design to employ these characteristics are closely related to the restrictive boundary conditions to which developable surfaces have to comply.
Chapter 3 presented the elementary differential geometric description of surfaces in general. Before addressing the discrete differential geometry of developable surfaces and their relation with vector mathematics, this chapter will focus on the elementary differential geometry in relation to a parametric description of developable surfaces and presents and in-depth research on the surface properties of developable surfaces.

4.1 introduction: theoretical discourse of developable surfaces

generic theory on ruled and developable surfaces

As noted in Chapter 2, ruled surfaces are both simple and fundamental to surface design and are of considerable importance in particular for the design of rationalised surfaces in architecture and structural design[98]. Generally, ruled surface follow the problem of defining a surface between two given space curves and a family of straight line segments. The input curves $\mathbf{C}_1(t)$ and $\mathbf{C}_2(t)$ have ideally the same degree and are parameterised over the same interval, usually the interval [0,1], Figure 4.1. If all pairs of points on the input curves are joined at the parameter value t by a straight line segments, a ruled surface strip connecting the curves is obtained.

In mathematical form this reads;

$$\mathbf{S}(u,0) = \mathbf{C}_1(t)$$
$$\mathbf{S}(u,1) = \mathbf{C}_2(t) \qquad \text{where } u \text{ is equal to } t \qquad \text{(Eq. 4.1)}$$

There are other methods to generate ruled surfaces as will be presented in parametric form in the following section, but in general, ruled surfaces follow the notion of linear interpolation; every isoparametric curve – a curve over the surface with a constant u value – is a straight line segment. As denoted before, generally, the tangent plane at a point on a ruling varies as the point moves over the ruling and as such the family of tangent planes define a double curved surface. Because ruled surfaces appear to be straight in one

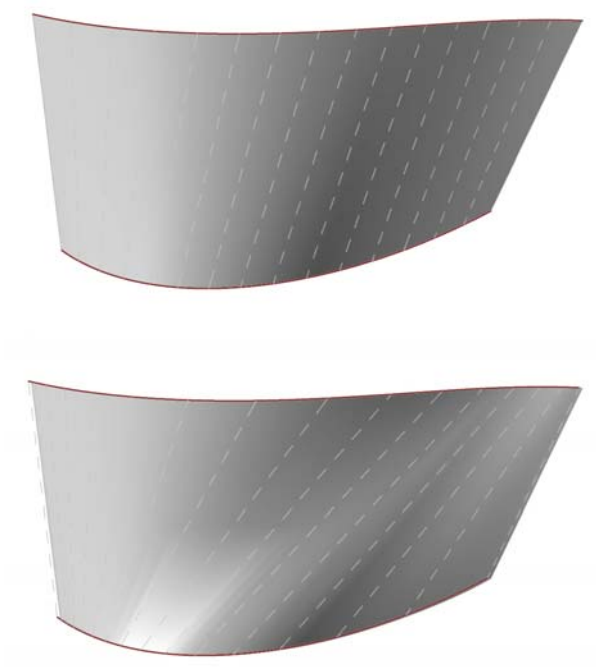


Fig. 4.1. Two ruled surfaces based on the same boundary curves; based on equal parametric length segments (top) and on redistributed t values (bottom)

[98] Farin, Curves and surfaces for computer aided design

'direction' they are easily mistaken for developable surfaces when they are in fact double curved. The double curvature is always expressed as a negative Gaussian curvature in all their surface points, excluding the subset of developable surfaces. This is proven by the fact that the normal curvature of any curve in a surface at a point **P** is a weighted average of the two principal curvatures of the surface at **P**[99]:

$$\kappa_N = \kappa_1 \cos^2\theta + \kappa_2 \sin^2\theta \qquad \text{(Eq. 4.2)}$$

where θ is the angle with the first principal direction, $\cos^2\theta$ and $\sin^2\theta$ have the domain [0,1] and $\cos^2\theta + \sin^2\theta = 1$. Since through every point on a ruled surface runs a curve that has normal curvature of 0 (the generator), this implies the two principal curvatures do not have the same sign at **P** and the Gaussian curvature is not positive there.

Only for a surface of zero Gaussian curvature it holds that it characterises developability. However, practical developability is obstructed by self-intersections of a surface. It is possible to show that the only regular geodesically complete developable surfaces are planes and cylinders, where geodesic completeness means that geodesics can be extended arbitrarily[100]. If, for example, for the definition of a tangent surface (see below) tangents of the input curve in both directions are taken, the surface is self-intersecting at this curve which is called the curve of regression, see Section 4.3.2. Obviously, also a cone shows a self-intersecting point at the apex when rulings are extended. Basically, if the generators do not intersect anywhere, but in the curve of regression or a cone's apex, then the surface is developable.

typologies of developable surfaces

Above, without explicitly addressing them, three basic typologies of the developable surface class are presented; cylinders, cones and tangential developables, Figure 4.2. If the definition of developable surfaces is not restricted to a sufficient smoothness, the class of these surfaces is too large to be useful. It then includes all possible ways of arranging crumpled paper in space[101].

cylindrical and conical developables

A cylinder and a cone are ruled surfaces where the generators are respectively parallel to a fixed direction or passing through a single point. As such, methods using conic and cylindrical surfaces are relatively

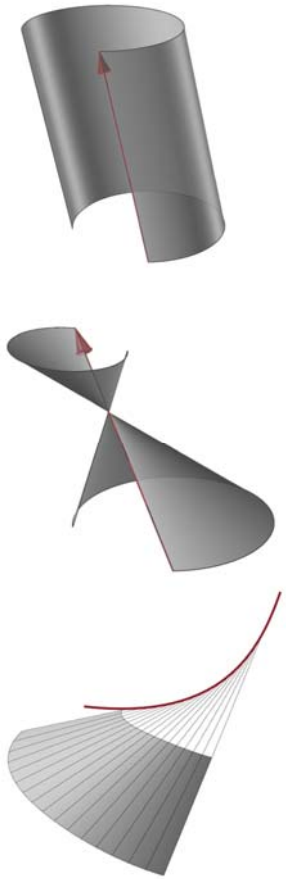


Fig. 4.2. Three basic types of developable surfaces. From top to bottom: cylindrical, conical and tangent surface

[99] Wolfram Mathworld, Euler curvature formula
[100] Pottmann, Computational line geometry
[101] ibid.

restricting in varying between various shapes and generally more difficult to use[102]. However, one of their main advantage is that they can be easily subdivided in flat panels; rectangular shapes for cylindrical surfaces and triangular shapes for conical surfaces.

tangent surfaces or tangential developables

A tangent surface is a surface spanning between rulings which are the tangent lines of a space curve, which is then called the curve or edge of regression. The correspondence between space curves and their tangent planes is one-to-one. And as such, the entire tangent surface is determined by means of its edge of regression[103]. A result of Euler states that most developable surfaces can be obtained as a tangent developable with the exceptions of generalised cones and cylinders and the plane[104].

4.2 parametric description of developable surfaces

4.2.1 parametric equations of ruled surfaces

Before discussing the parametric description of developable surfaces, first parametric equations of ruled surfaces are presented.

As mentioned before, Pottmann et al. describe two types of ruled surfaces:

- ruled surfaces by moving a straight line along a directrix curve
- ruled surfaces by connecting corresponding points of two generating curves

The parametric representation of the first type is

$$\mathbf{S}(u,v) = \mathbf{C}(u) + v\mathbf{a}(u) \qquad \text{(Eq. 4.3)}$$

where $\mathbf{C}(u)$ is the parametric representation of the directrix curve, also called the base curve, and the generator $\mathbf{a}(u)$ describes the continuously changing unit direction vector of the moving straight line segment, also called the ruling[105]. A given point on the directrix curve with the parameter value u thus denotes the position of the moving line segment and the value v gives the particular point along this line.

[102] Nolan, Computer-aided design of developable hull surfaces
[103] Glaeser, Developable surfaces in contemporary architecture
[104] Wikipedia, Tangent developable
[105] Pottmann, Architectural geometry

For the second type, the ruled surface is formed from two spatial curves, $\mathbf{C}_A(u)$ and $\mathbf{C}_B(u)$, where two points on each curve, generally corresponding to the same parameter value u, are joined by a line. Depending on the parameterisation of the two directrices, different ruled surfaces can then be generated.

The equation describing this type of ruled surface is

$$\mathbf{S}(u,v) = (1-v)\mathbf{C}_A(u) + v\mathbf{C}_B(u) \qquad \text{(Eq. 4.4)}$$

where $\mathbf{C}_A(u)$ and $\mathbf{C}_B(u)$ are the parametric representations of the directrices. This equation is closely related to Equation 4.1. Obviously, the equations of the two representations for ruled surfaces are identical when $\mathbf{C}(u) = \mathbf{C}_A(u)$ and $\mathbf{a}(u) = \mathbf{C}_B(u) - \mathbf{C}_A(u)$. Also note the resemblance with Equation 2.7. In general, there is virtually no restriction to the input curves other than having to be defined over the same parameter interval, usually [0,1][106].

The simplest examples of ruled surfaces are planar surfaces, cylindrical surfaces and conical surfaces. When the direction vector is constant, $\mathbf{a}(u) = \mathbf{a}$, the rulings are all parallel and a cylindrical ruled surface is generated. A conical ruled surface results from choosing a constant position vector, $\mathbf{c}(u) = \mathbf{c}$.

example

In the example of a Möbius strip, there are no two generating curves to generate a ruled surface with. A circle $\mathbf{C}(u)$ is used as the directrix and a set of rulings is swept over this curve with a constant rotation about $\mathbf{C}(u)$. When the generator arrives again at the starting point and the rotation about the circle is one half turn a standard Möbius strip is generated. Therefore, a Möbius strip is an example of the type; ruled surfaces by moving a straight line along a directrix curve.

The parametric representation of the directrix and generator are respectively

$$\mathbf{C}(u) = (r\cos(u), r\sin(u), 0) \qquad \text{(Eq. 4.5)}$$

$$\mathbf{a}(u) = (\cos(u/2)\cos(u), \cos(u/2)\sin(u), \sin(u/2)) \qquad \text{(Eq. 4.6)}$$

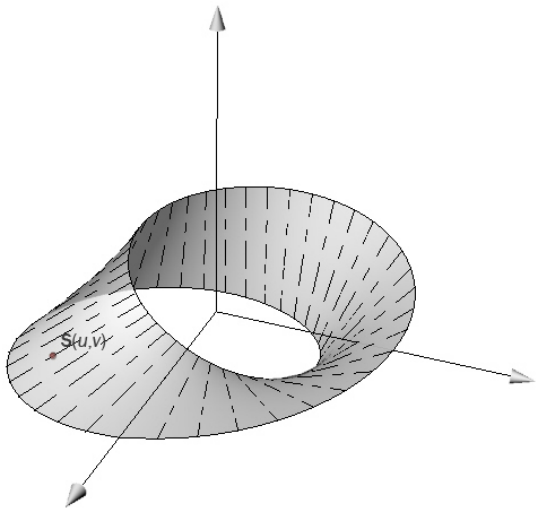


Fig. 4.3. Parametrically defined Möbius strip

[106] Farin, Curves and surfaces for CAGD

Combining these equations and multiplying the generator with the desired width of the strip the parametric representation of a Möbius strip is

$$x(u,v) = r\cos(u) + v\cos(u/2)\cos(u)$$
$$y(u,v) = r\sin(u) + v\cos(u/2)\sin(u) \qquad \text{(Eq. 4.7)}$$
$$z(u,v) = v\sin(u/2)$$

with the domains u = [0,2π] and v = [-½width of strip, ½width of strip]. The point in Figure 4.3 shows the location of $\mathbf{S}(u,v) = \mathbf{S}$ (1.75π,-3.5) = (16.4,-16.4,-1.3).

Essentially, a ruled surface is completely determined by a base curve and the direction of the generators at their points of meeting with the curve[107] which by definitions are geodesics[*].

4.2.2 parametric equations of developable surfaces

Since developable surfaces are a subclass of ruled surfaces, they can be described by the mathematical equations of ruled surfaces. However, they are subject to some restrictions. To find the condition for a ruled surface to be developable, consider the general Equation 4.2 with parameter values u and u + du which denote two adjacent rulings[108]

$$\mathbf{S}(u,v) = \mathbf{C}(u) + v\mathbf{a}(u) \qquad S(u+\mathrm{du},v) = \mathbf{C}(u+\mathrm{du}) + v\mathbf{a}(u+\mathrm{du})$$

Consider the triple scalar product

$$[\mathbf{C}(u+\mathrm{du}) - \mathbf{C}(u)] \cdot [\mathbf{a}(u) \times \mathbf{a}(u+\mathrm{du})] \qquad \text{(Eq. 4.8)}$$

By definition, the three vectors $\mathbf{C}(u + \mathrm{d}u) - \mathbf{C}(u)$, $\mathbf{a}(u)$ and $\mathbf{a}(u + \mathrm{d}u)$ are all in the plane of the two rulings, Figure 4.4. If the rulings intersect, such as in Figure 4.4. as with a cone, then the vector $\mathbf{a}(u) \times \mathbf{a}(u + \mathrm{d}u)$ is orthogonal to the plane of the two rulings and so the triple scalar product of Equation 4.8 has a zero value. If the rulings are parallel as with a cylinder, then $\mathbf{a}(u) \times \mathbf{a}(u + \mathrm{d}u) = 0$ and again the triple scalar product has a zero value.

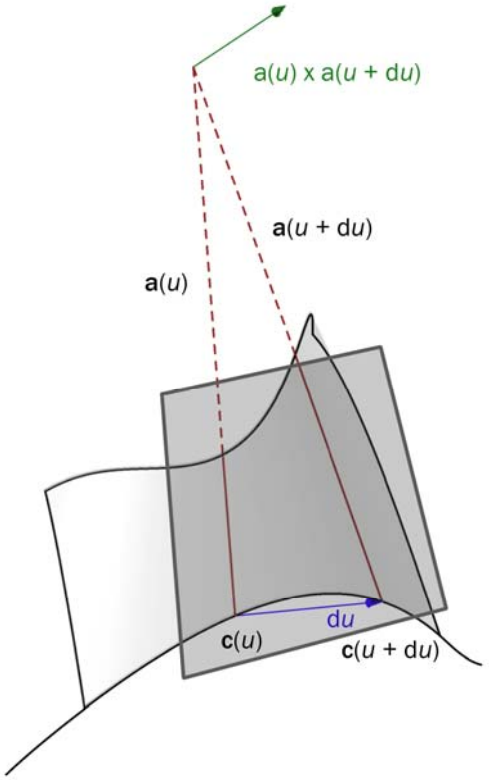


Fig. 4.4. Vectors for determining the developable condition of a ruled surface

[107] Eisenhart, A treatise on the differential geometry of curves and surfaces

[*] On a sphere, the geodesics are great circles, like the equator. In a plane, the geodesics are straight lines. As such, the generators of developable surfaces are straight lines as well. Basically, geodesics preserve a direction on a surface.

[108] Davies, An introduction to computational geometry for curves and surfaces

Since the coplanar rulings either intersect or are parallel, it follows that the condition for the two rulings to be coplanar is that

$$[\mathbf{C}(u + \mathrm{d}u) - \mathbf{C}(u)] \cdot [\mathbf{a}(u) \times \mathbf{a}(u + \mathrm{d}u)] = 0$$

If the limit value for du is taken, the required condition for a ruled surface to be developable is obtained

$$\dot{\mathbf{C}}(u) \cdot (\mathbf{a}(u) \times \dot{\mathbf{a}}(u)) = 0 \qquad \text{(Eq. 4.9)}$$

4.3 generic geometric properties and analysis of developable surfaces

4.3.1 curvature, offset and continuity analysis for developable surfaces

Gaussian curvature properties

As mentioned before, the Gaussian curvature of developable surfaces is zero. Carl Friedrich Gauss developed the approach of spherical mapping to determine curvature of surfaces. The Gaussian spherical mapping, or normal mapping, uses the mapping from a surface onto a unit sphere in a way that the surface unit normal vectors are seen as coordinate vectors of points from the centre of the sphere, thus representing a point on the unit sphere. The Gaussian spherical map can be defined globally if and only if the surface is orientable, i.e. for instance Möbius strips cannot be mapped. The area of the image of the Gauss map is called the total curvature and is equivalent to the surface integral of the Gaussian curvature[109]. In other words, the local area distortion of the from the original surface to the Gaussian spherical map is a measure for the Gaussian curvature.

Following from the fact that the Gaussian curvature of developable surfaces is zero, their Gaussian spherical map is one-dimensional; the area of the image is zero. In other words, the Gaussian spherical map of a developable surface is a curve (or a point in case of a planar surface), because all points of the same ruling have the same tangent plane and therefore have parallel normals and the same Gaussian mapping point[110], Figure 4.5. Reversely, if the normal map is a single curve, then the directrix of the surface is a single continuous curve[111].

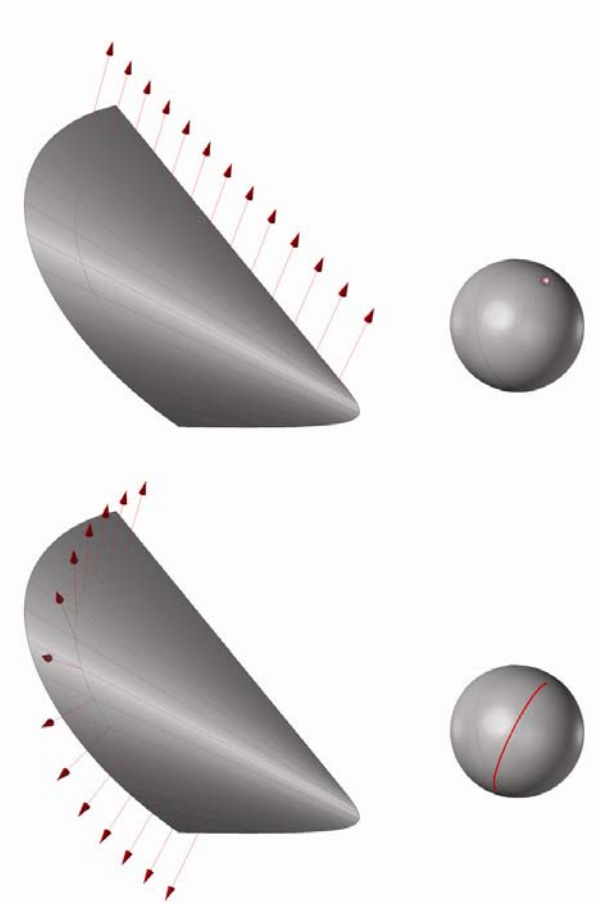


Fig. 4.5. Gaussian spherical image of a tangent developable surface. Top: the Gaussian mapping point of the parallel normals along a ruling. Bottom: the Gaussian mapping curve of the normals along an isocurve in the direction of the directrix

[109] Wikipedia, Gauss map

[110] Pottmann, Architectural geometry

[111] Rose, Developable surfaces from arbitrary sketched boundaries

offsetting properties

For surfaces, consider a curve $\mathbf{C}(t)$ contained in the surface and the spheres of radius d whose centres are contained by the curve. The envelope of these spheres is the pipe with axis $\mathbf{C}(t)$ and radius d. If the original surface is a ruled surface with rulings $\mathbf{C}(u)$, then its offset is the envelope of the one-parameter family of cylinders with radius d and axis $\mathbf{C}(u)$, which again is a ruled surface[112].

For the three basic types of developable surfaces the following holds true; the offsets of cylinders are cylinders again. For a cone, consider a sphere with radius d whose centre coincides with the cone's vertex. The two-sided offset surface of a cone at distance d is in general composed of a developable surface of two component which touch the sphere. The complete offset of a tangent surface of a space curve consists of a developable surface[113].

In general it holds that because all points on a ruling of a developable surface have the same tangent plane, the corresponding points of an offset surface at distance d lie on a straight line at this distance. The tangent plane of the offset surface is parallel to the original tangent plane. Thus, the offset surface is also a developable surface as its rulings and tangent planes are at distance d to the corresponding rulings and tangent planes of the original surface[114].

continuity properties

Due to the fact that there always is a single tangent plane along each generator, in some cases it is possible to connect developable surfaces with each other[115]. To properly connect the two adjacent patches, a shared surface generator should be correctly aligned and the surface normals of each patch at that boundary should be parallel, Figure 4.6. This connection then has a G^1 continuity.
If developable surface patches are joined along an edge which is not parallel to the surface generators then only a G^0 continuous connection is possible.

[112] Pottmann, Computational line geometry

[113] ibid.

[114] Pottmann, Architectural geometry

[115] Glaeser, Developable surfaces in contemporary architecture

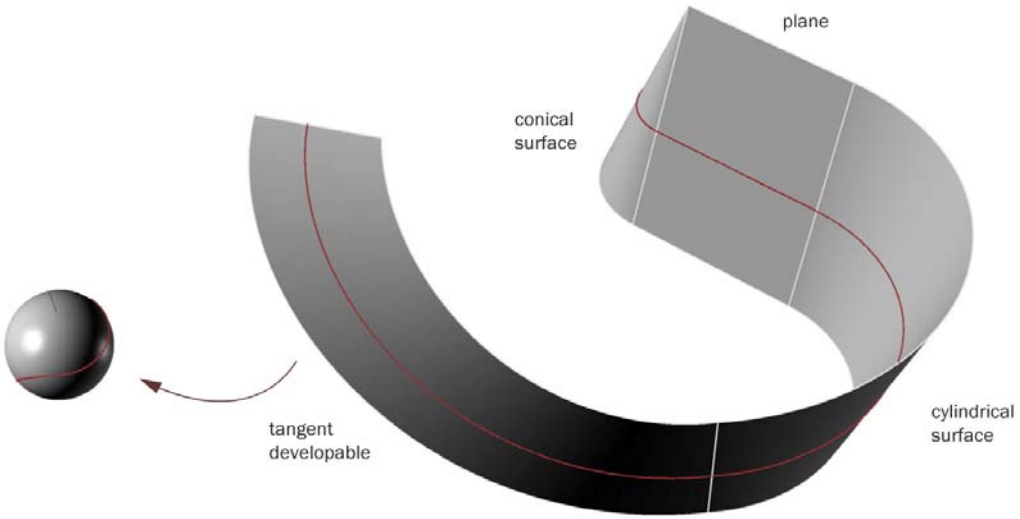


Fig. 4.6. G^1 continuous connections between four types of developable surfaces and their Gaussian spherical map

4.3.2 geometric properties of the curve of regression

As mentioned before, with the exception of the plane, cylinder, and cone, every developable surface in three-dimensional Euclidean space is the tangential developable of a certain curve. Intuitively, it is a curve along which the surface needs to be folded during the process of developing into the plane[116]. This curve is called the curve of regression; along it, the tangent developable consists of two sheets corresponding respectively to the positive and negative tangent vectors. These tangents are obviously tangent to one another as well along the curve, and thus form a sharp edge[117], Figure 4.7. The curve of regression and similar infeasible conditions on developable surfaces frequently crop up, both in modelling activities and numerical solutions to the generation of these surfaces[118]. This for instance in disallowing the developable surface to be extended smoothly beyond this boundary.

[116] Wikipedia, Tangential developable

[117] Eisenhart, A treatise on the differential geometry of curves and surfaces

[118] Shelden, Digital surface representation and the constructability of Gehry's architecture

Therefore, it is of interest to investigate the nature of the developable near this curve, in particular, since the surface is shown to be singular here which cannot be overcome by a change of surface parameters[119]. In Section 6.2.1, the curve of regression is discussed in more detail and it is explained how it can be generated from a given developable surface.

4.4 epilogue: utilisation of parametric differentials

Parametric equations for ruled and developable surfaces are not frequently employed in architectural design. Especially, since most designers work with NURBS based geometry in 3D software which do not present the intrinsic differential methods for generation and analysis of curves and surfaces. Nonetheless, deriving these equations essentially leads to a better understanding of the intrinsic surface structure, also in relation with the parametric equations of surface properties. It is shown in this chapter that the complexity of the parametric equations of developable surfaces are corresponding to the relative straightforward geometric properties of this type of surfaces; presenting the parametric description of a developable surface as a restricted equation of the general class of ruled surfaces.
This chapter also introduced the Gaussian mapping method; a method to visualise a surface property – the Gaussian curvature – by means of geometrical operations. Essentially, this method is still based on differentially derived properties, such as deriving the surface normals. Subsequently, however, the Gaussian spherical map is generated by geometrically mapping a discrete number of surface normal vectors on to the unit sphere, after which the resulting points on the sphere are interpolated.

Combining the general intrinsic differential methods – and knowing how these methods work – with discrete geometrical operations forms the base of the digital generation of developable surfaces and component development as described in the subsequent chapters. Firstly, the step from differential geometry to discrete differential geometry and vector mathematics is described. Secondly, the combined methods are employed in the parametric modelling of architectural developables.

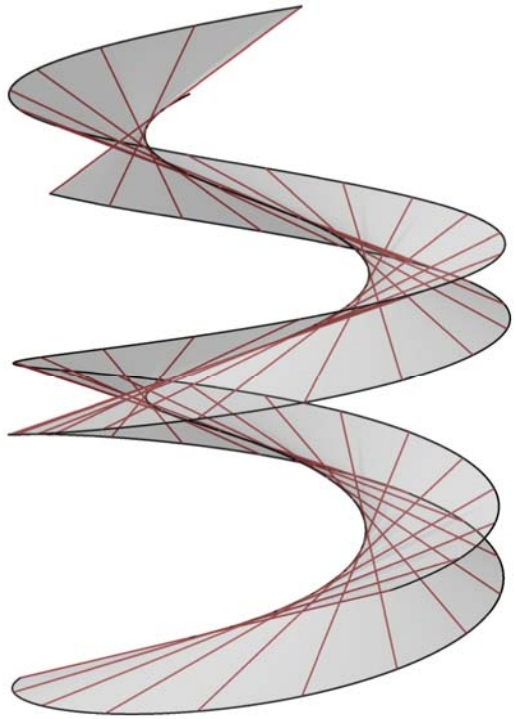


Fig. 4.7. Two sheets of the tangent developable surface of a circular helical curve. This curve is the cuspidal edge of the two sheets

[119] Stoker, Differential geometry

5 digital generation and modelling of developable surfaces

In Section 3.2, it was stated that one advantage of parametrically defined curves and surfaces is that their local properties, such as normal direction and curvature can be derived relatively easily as long as their mathematical formulations can be derived. However, since double curved surfaces in architectural design are generally not directly based on parametric surface descriptions – with various exceptions such as hyperbolic paraboloids –, the presented formulas in the previous chapters are not well applicable in design.

The following chapters focus on discrete differential geometry in order to generate discrete elements for the development of surfaces. This chapter is firstly introduced with presenting three general typologies of methods for modelling developable surfaces. Subsequently, definitions for discrete differential geometry are given related to the research and a developability condition in relation to the discrete description of rulings is given based on vector mathematics.

5.1 introduction: approach of design with developable surfaces

Various examples exist of methods to generate developable surfaces either via approximation or direct modelling. Rose et al. provide a review of some of these methods for modelling developable surfaces. Without going into detail of specific techniques or algorithms, below a coarse overview of these methods is given.

developable approximation

Given an existing nondevelopable surface, a large number of methods originating from different theoretical and practical fields aim at approximating it with one or more developable surface patches based on for instance combining conical surfaces, the generation of (triangular) strips or deforming double curved surfaces to approximate developable surfaces.

Generally speaking, the approximation approach is highly restricted, as the methods can only succeed if the original input surfaces already have fairly small Gaussian curvature. Moreover, in most cases the final result is not analytically developable. This can be problematic in manufacturing setups, where the surfaces need to be realised from actual planar patterns. In these setups the distortion caused by using unfolded patterns from approximate developables can be quite significant[120].

[120] Rose, Developable surfaces from arbitrary sketched boundaries

direct modelling

Most existing methods for modelling developable surfaces require the user to clearly specify the ruling direction for the surface. In a discrete setup for modelling developable surfaces, the input is based on polyline directrices and the output is a developable strip where each interior edge approximates a ruling connecting the two directrices, Figure 5.1.
Another direct modelling method is the highly time consuming alternative presented by some of the commercial modelling tools which first design a planar pattern for the surface and then deform it into the desired shape using bending and physical simulation. An example of this is given in Section 6.3.2 where a digital paper strip is modelled, which allows for instance for rolling up the strip.

The methods used in this research and are as well categorised in methods for modelling developables either via developable approximation or direct modelling. Rose et al. present one other method as well; sketch-based modelling.

sketch-based modelling

In their paper *Developable surfaces from arbitrary sketched boundaries*, Rose et al. describe a sketch-based system for modelling general developable surfaces by using an interface from which users can sketch the boundaries of each surface patch as a 3D polyline. They adopt a sketching framework for modelling of developable surfaces and use it to obtain the 3D boundaries of the modelled surfaces[121]. For the functionality of this method, reference is made to the paper.

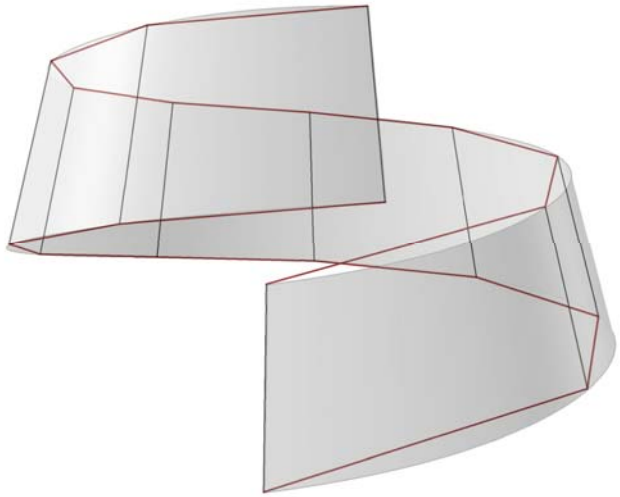


Fig. 5.1. Approximated developable surface based on polyline directrices

5.2 discrete differential geometry of developables

Both in approximation methods and direct modelling, generating developables surfaces based on discrete differential geometry will reduce complexity in designing these surfaces in relation to methods based on differential geometry.
In order to understand the distinction between differential geometry and discrete differential geometry, consider the following. Roughly speaking, classical differential geometry is the study of local properties of curves and surfaces; properties which depend only on the behaviour of the curve or surface in the neighbourhood of a point. Global differential geometry studies the influence of the local properties on the behaviour of the entire curve or surface[122]. Based on mathematical equations, in an intrinsic, explicit and implicit or parametric manner, derivatives are used to calculate these local properties, such as normal and tangent directions and curvature.

[121] Rose, Developable surfaces from arbitrary sketched boundaries

[122] Do Carmo, Differential geometry of curves and surfaces

Discrete differential geometry is an active mathematical terrain where differential geometry and discrete geometry meet and interact. It provides discrete equivalents of the geometric notions and methods of differential geometry. Generally, current interest in this field derives not only from its importance in pure mathematics, but also from its relevance for other fields such as computer graphics and design[123].
Discrete differential geometry initially arose from the observation that when a notion from smooth geometry – such as that of a developable surface – is discretised, the discrete objects may not merely be approximations of the smooth ones, but preserve or approximate the fundamental properties of the smooth equivalent and have special properties of their own, which makes them form a coherent entity by themselves[124]. Often such a discretisation clarifies the structures of the smooth theory and as such it might be claimed that the discrete theory is in a sense the more fundamental one: the smooth theory can always be recovered as a limit, while it is a nontrivial problem to find which discretisation has the desired properties.

In the research of parametric modelling of architectural developables, discrete differential geometry plays an important role. It provides for the generation of discretely defined rulings. Subsequently, the rulings are used as input information to generate smooth developable surfaces, which are in essence approximations of surfaces based on differential geometry. Where the limit goes to a zero distance between subsequent discrete rulings, the approximation is equivalent to the surface based on differential geometry, as will also be pointed out in the next section.

5.3 ruling vector mathematics for developable surfaces

As rulings are straight elements with a direction, the relation with vector mathematics is evident. Additionally, the expression for developable surfaces lead to complex equations when reduced to a function of one variable, but the definition of developables is relatively simple in terms of vectors[125]. For these two reasons and in relation with the advantages mentioned above in using discrete differential geometry, rulings will be defined as vector at discrete locations.

5.3.1 ruling vector description between two curves

The choice for defining rulings as vectors is supported by the geometrical definition described by Nolan which is based on the determination of rulings between two curves where a plane is tangent to the surface along the full length of the ruling, Figure 5.2.

[123] Bobenko, Discrete differential geometry
[124] ibid.
[125] Nolan, Computer-aided design of developable hull surfaces

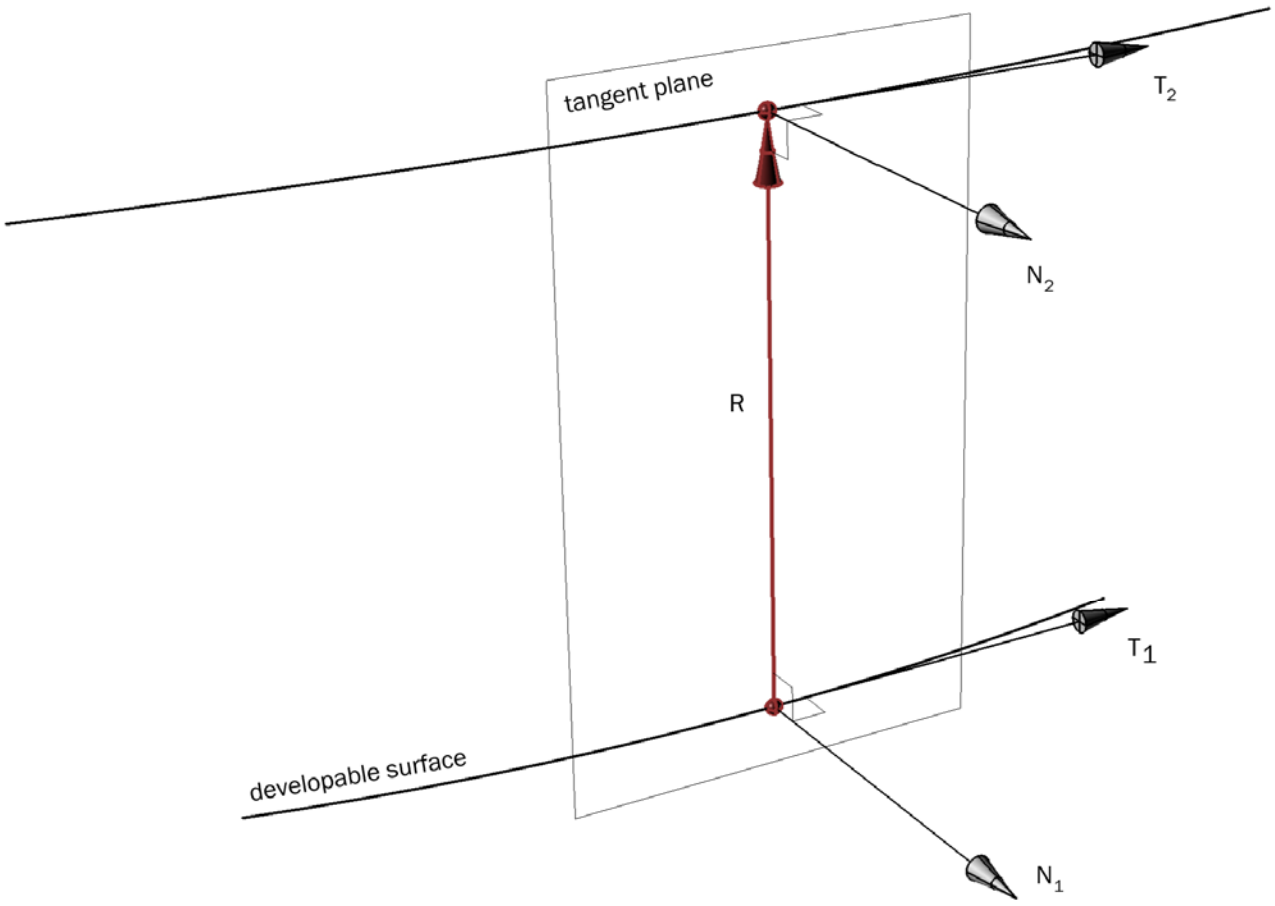


Fig. 5.2. The ruling vector description and the tangent plane along this ruling

For a ruled surface, it holds that if two points, each one lying on a curve in the surface, can be found such that the tangents to the curves at these points lie in the same plane, they define the position and direction of a ruling **R**.
With t_1 and t_2 being the points on respectively the first and second curve, $\mathbf{T}_1$ and $\mathbf{T}_2$ their tangent vectors at these points and **R** denoting the ruling vector, the normal vectors can be found for the plane determined by the ruling and the tangent vector as follows:

$$\mathbf{N}_1 = \mathbf{R} \times \mathbf{T}_1$$
$$\mathbf{N}_2 = \mathbf{R} \times \mathbf{T}_2 \qquad \text{(Eq. 5.1)}$$

R will be a ruling of a developable surface if $\mathbf{N}_1$ and $\mathbf{N}_2$ are parallel; that is, if these vectors are normal to the same plane. The condition for developability can then be described mathematically as

$$\mathbf{N}_1 \times \mathbf{N}_2 = 0 \qquad \text{(Eq. 5.2)}$$

It will be clear that this condition is met for a plane, a cylinder and a cone, Figure 5.3.

5.3.2 ruling vector directions for general developable surfaces

Also for other ruling definitions, vector mathematics are well applicable in order to generate rulings that comply with the conditions for developability. Tangent vectors are used by default to generate a tangent developable surface from an input space curve. The vectors denote the start point of the semi-infinite rulings in the curve tangent direction and its inverse direction. Also for the definition of cylindrical and conical surface patches, the ruling directions serve as an input which are respectively parallel or intersecting in one point.

Fig. 5.3. Parallel normals of a planar, cylindrical and conical surface

5.4 epilogue: combining differential geometry with discrete vector mathematics

In general, generating developables based on parametric equations hardly lends itself for architectural design. Nonetheless, the mathematical formulas of the previous chapters presented focal points for intrinsic surface properties which form the basis for a discrete differential geometry approach. This approach aims to preserve a selected structure when going from a continuous abstraction to a finite representation for computational purposes. For example, for a piecewise linear approximation of a surface, i.e. a mesh, one may define Gaussian curvature in such a way that important theorems are preserved in the discrete setting[126].

Essentially, a differential geometry approach focuses on the intrinsically perspective of a surface, reflecting the properties determined solely by the distance within the surface as measured along curves on the surface[127]. The discrete approach of, for instance defining rulings based on vector mathematics, relates to an extrinsically perspective, regarding the embedding in Euclidian space. But how does the combination of these approaches allow for a parametric design method for digital developables? One way is to utilise them in the development of custom tools that either directly model developable surfaces or approximate them based on nondevelopable input surfaces.

[126] Cal Tech, Discrete differential geometry

[127] Wikipedia, differential geometry of surfaces

part03 – tool development and implementation

6 tool development for the parametric developable design approach

Part03 of this thesis focuses on the development and the description of the functionality of a design approach which encapsulates methods for parametric modelling of architectural developables. The presentation of this approach serves as an example of the general parametric rational design approach and it is shown how it adopts the previously discussed methods of direct modelling and approximation of developables as well as methods for the analysis of surfaces in relation to the properties of developables surfaces. The conceptual outline of the general approach is discussed in Section 6.1. Functional components containing the logic for the parametric analysis and modelling of developable surfaces are presented in Section 6.2. The final section of this chapter discusses the algorithms of the coded objects and the functionality on which these components are build.

6.1 introduction: conceptual outline of the parametric rational design approach

6.1.1 problem approach of parametric modelling of rational surfaces

Implementing a design approach for a parametric environment allows for maintaining an intrinsic structure and logic of the construction of rational surfaces where the input parameters are, within the aforementioned boundaries, providing design freedom. From an architectural design perspective, the approach should be able to follow practical digital ways of designing; that is 1) from a design sketch from which parametric input is extracted to a rationalised digital 3D model, Figure 6.1, and 2) from a digital design model to an approximation of this design based on rationalisation principles, both followed by the processing of the rationalised model to manufacturable elements.

The proposed design approach, as visualised in Figure 6.1, is primarily set out to define rational surface geometry based on parametric digital geometry extracted from design ideas. The approach incorporates and makes use of predefined logic defining boundary conditions of rational surface classes, amongst which developable surfaces, and ideally also contains definitions for principles for manufacturing and construction, for instance related to material properties. As such, it is envisioned that the approach is a parametric modelling primitive which provides for interactive rational modelling methods based on the generation and deformation of rational surfaces for both architectural purposes as well as manufacturing and construction purposes, and thus supporting the design process. Alternatively, the approach allows for incorporation of approximating rational surface geometry based on non-rational defined or for instance double curved surface geometry.

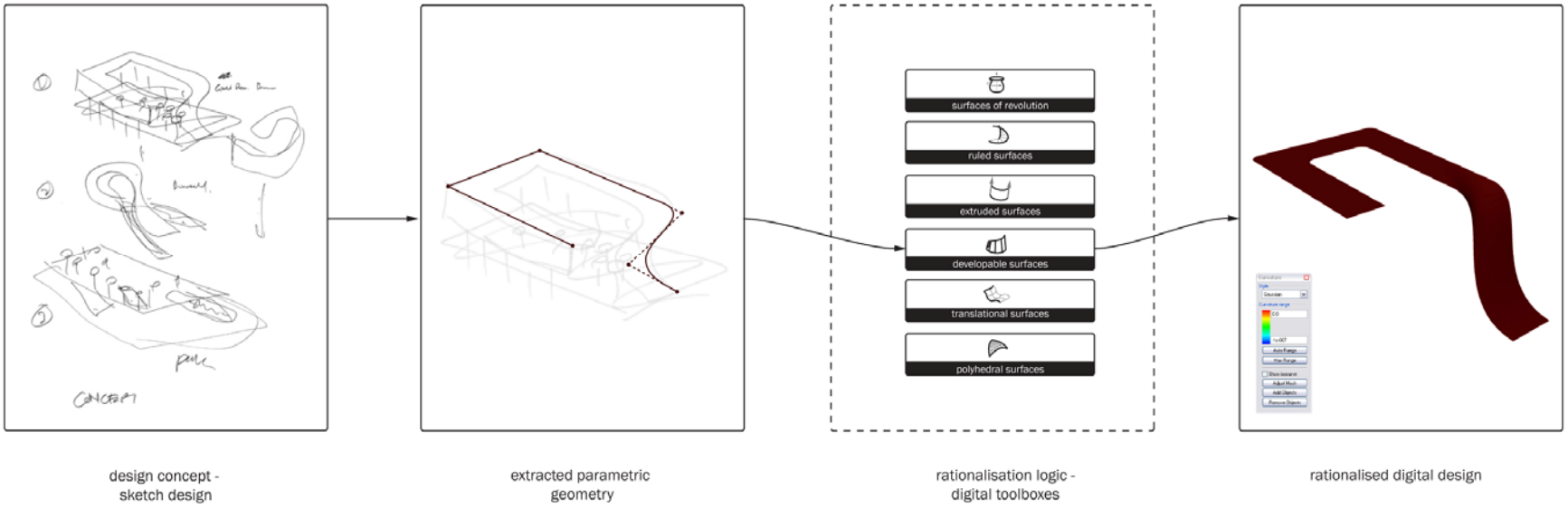


Fig. 6.1. Abstract visualisation of the parametric rational design approach [design sketch, left: http://plusmood.com/, March 2011]

Concerning developable surfaces, their intrinsic structure provides for the advantageous properties of single-curve geometric objects, but as such are restricted to specific design constraints. And although developables are used frequently in various fields of industry, design taking these constraints into account proves to be difficult, especially if a sculptural appearance is pursued. In general it holds that the problem of defining rational complex geometry is not that of the surface description itself, but the problem relates to answering to boundary conditions. In other words, of most of the rationalisation methods, the input parameter definitions and the surface description are intrinsically not complex, but conforming to the boundary conditions which follow from the surface type determines whether a rationalised surface structure can follow the design intent. Also for developable surfaces, the surface description is relatively simple, but the main question relates to how the restrictive design conditions of developable surfaces can be taken into account whilst allowing for a full descriptive design language and how the logic on which design with developable surfaces is based can be adopted by the designer.

6.1.2 functionality of the parametric developable design approach

The proposed design approach for developable surfaces is deployed via a digital toolbox containing various algorithmic methods categorised in three toolbox 'compartments' which are based on the theory of discrete differential geometry and vector mathematics, Figure 6.2. The algorithmic methods, or tools, allow for surface analysis and modelling and support the designer in the search for geometrically rationalised surfaces to find an answer to the initial ideas of the design. As such, the toolbox provides 'hints' to guide the generation of surfaces towards the constructible geometry of developable surfaces. This holds that the tools may need alterations of the design input for them to generate surfaces based on the input, however they cannot be categorised as design tools. This in a sense that the toolbox provides the outcome of a predefined logic within a restricted workflow and therefore lacks the essentials of a set of design tools, which allow for a more creative process.

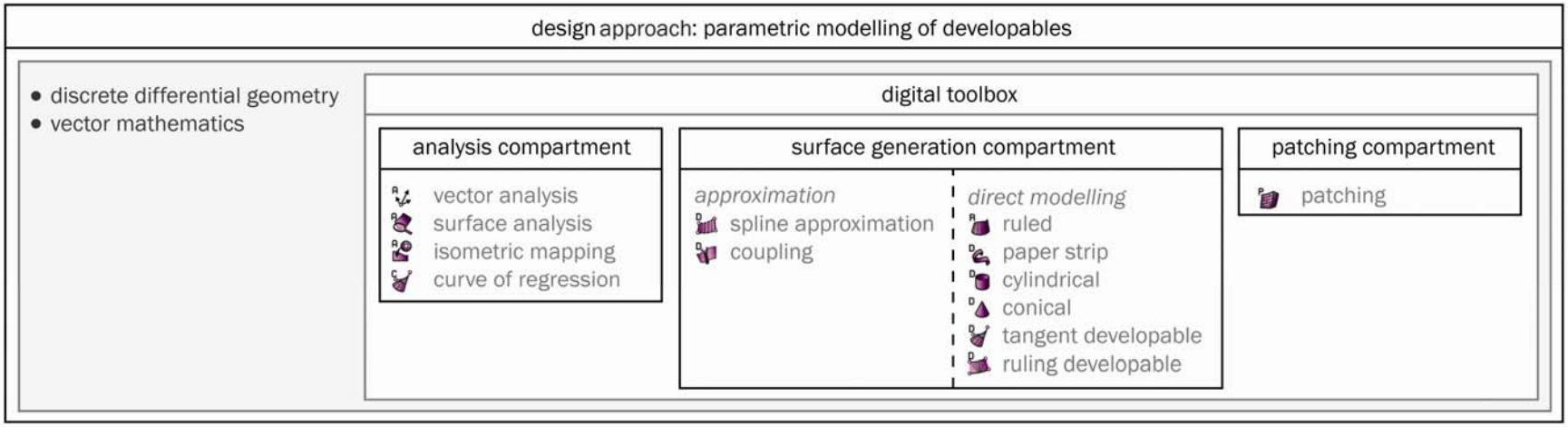


Fig. 6.2. Conceptual outline of the parametric developable design methodology

6.2 development of parametric components for analysis and generation

In this section, the general functionality of the components is described. In Subsection 6.2.1, the components are presented individually and Subsection 6.2.2 shows a number of validation test cases.

6.2.1 component description and functionality

As presented in Figure 6.2, the digital toolbox contains three compartments. The tools in these compartments focus on 1) the analysis of surfaces, 2) the generation of ruled and developable surfaces – subdivided in approximation and direct modelling sub-compartments – and 3) the patching of surfaces in manufacturable elements, Figure 6.3. Below, the functionality of the compartments are presented.

The icons of the Grasshopper components display a letter indicating the type of outcome of the component:

A: analysis data
C: curve geometry
D: developable surface
R: ruled surface
P: surface patches

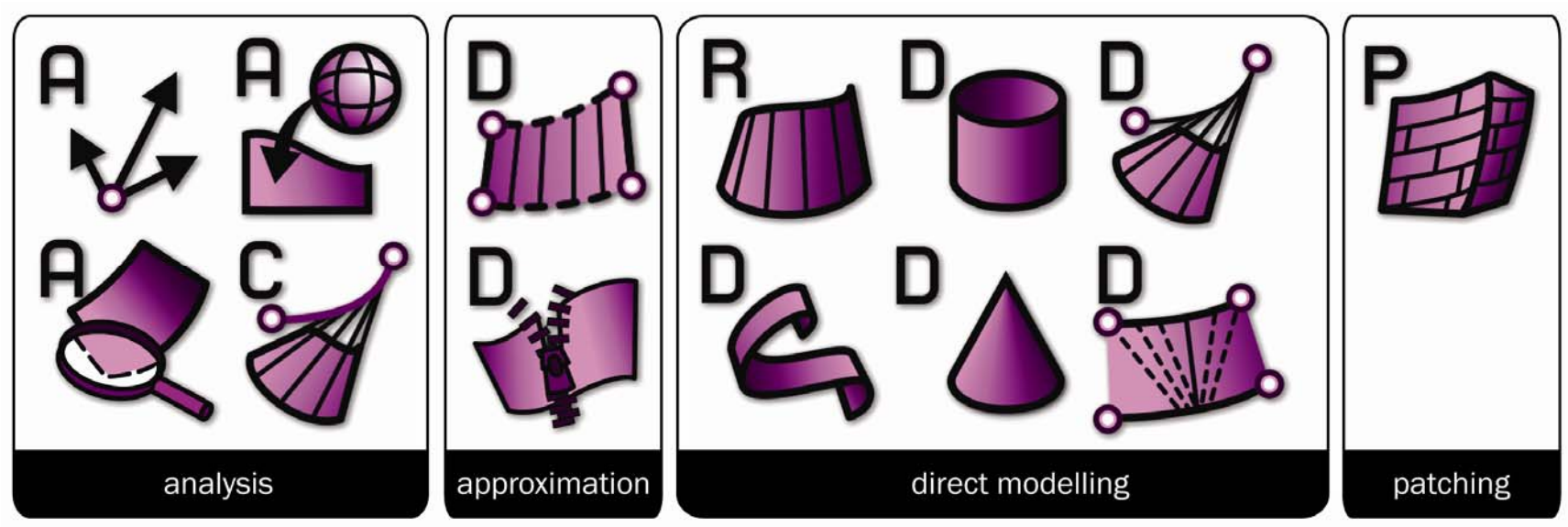


Fig. 6.3. The Grasshopper component icons of the toolbox

analysis components

The main goal of the analysis tools is to provide insight in the structure and geometry of surfaces in relation to the properties of developable surfaces, i.e. primarily the presence of straight rulings and the single curvature property.

vector analysis

In order to check the local conformity with the condition of developability, i.e. two normal vectors on a ruling need to be parallel, basic vector mathematical operations can be used. Whether the normals are parallel can be checked by taking the ruling and the tangent directions at the intersection points with the input curve under consideration. The normal vector can be distilled by calculating the cross products of these two vectors.

Figure 6.4 shows six rulings from one point on the bottom curve to six points on the top curve. With the direction of these rulings and the tangent vectors at each point, the local normal vectors are distilled via the cross product and it can be checked whether the warp angle over the rulings is equal to zero.

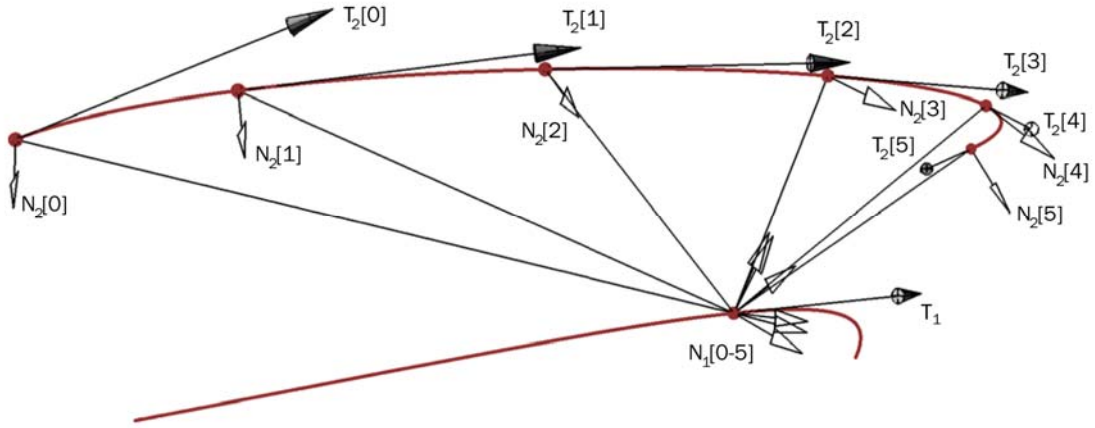


Fig. 6.4. Derivation of the normal vectors based on the tangent and ruling direction a specific locations

Although it may be presumed that the definition of the cross product is known, it is noted here that the cross product vector can be calculated with:

$$\mathbf{a} = \mathbf{b} \times \mathbf{c} \tag{Eq. 6.1}$$

where **a**, **b** and **c** are vectors:

$$\mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}, \mathbf{b} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}, \mathbf{c} = \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix} \tag{Eq. 6.2}$$

so that:

$$\begin{aligned} a_x &= b_y c_z - b_z c_y \\ a_y &= b_z c_x - b_x c_z \\ a_z &= b_x c_y - b_y c_x \end{aligned} \tag{Eq. 6.3}$$

The vector angle is defined by the dot product and the length of the vectors under consideration

$$\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\| \|\mathbf{b}\| \cos\theta \tag{Eq. 6.4}$$

As such:

$$\theta = \arccos\left(\frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|} \right) \tag{Eq. 6.5}$$

where the scalar value of the dot product of two vectors is defined as:

$$\mathbf{a} \cdot \mathbf{b} = a_x b_x + a_y b_y + a_z b_z \tag{Eq. 6.6}$$

When the vector angle θ is zero, the vectors are parallel and denote a ruling that complies with the developability condition.

The vector analysis component allows for the calculation of the cross product, dot product and vector angle of two sets of vectors and may, therefore, also provide a check of the outcome of other components which will be discussed below.

surface analysis

Recall the subsection in Section 3.2.1 on surface curvature. Here it is explained that the normal curvature, κ_n, is the curvature of the curve projected onto a plane which is defined by the curve's tangent $\mathbf{T}(u,v)$ and the surface normal $\mathbf{N}(u,v)$ at the point $\mathbf{P}(u,v)$. Regarding all possible tangent vectors at a point on the surface, then the maximum and minimum values of the normal curvature at a point are called the principal curvatures, κ_1 and κ_2[128], Figure 6.5.

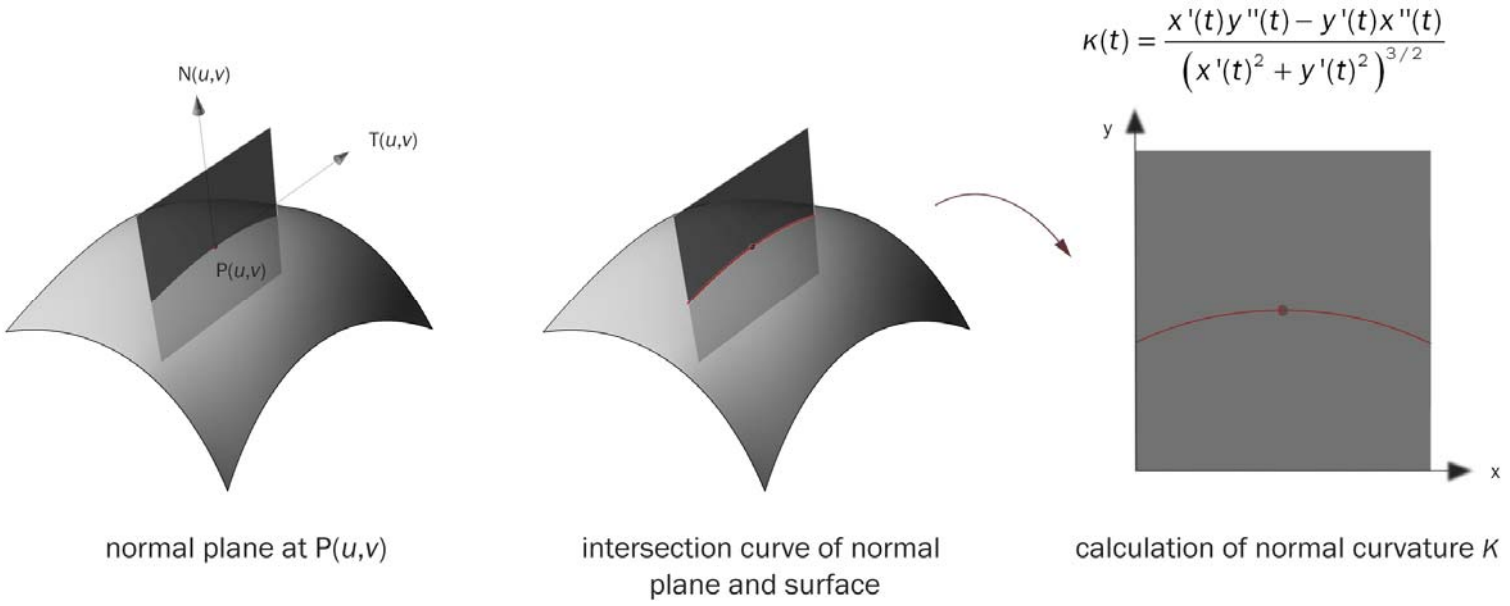


Fig. 6.5. Derivation of the normal curvature in an arbitrary tangent direction

[128] Wikipedia, Curvature

For a developable surface, the minimum value of the principal curvatures is zero, hence the Gaussian curvature, calculated with $\kappa_1\kappa_2$, is equal to zero as well. The surface analysis component determines the direction and value of the principal curvatures, primarily based on methods in the RhinoCommon SDK which calculates the derivatives of surface curves and the surface itself.

The values of the principal curvatures are presented as lines at the point on the surface under consideration, where the length of the lines indicates the value of the curvature. Also, the surface is coloured – from dark purple to bright pink – where the minimal principal curvature lies under a certain tolerance value, giving a hint where the surface is locally almost single curved and near-developable patches can be recognised. When the surface is single curved within a tolerance along a straight line over the surface, a ruling is generated, visualising the structure of the developable patch.

Figure 6.6 shows a distorted ruled surface, left, and the visual output of the surface analysis component, right. The latter gives an indication where the surface can be considered as developable within a certain range. Related to this, three rulings are generated perpendicular to the locally defined strongest curvature as indicated by the network of curvature lines.

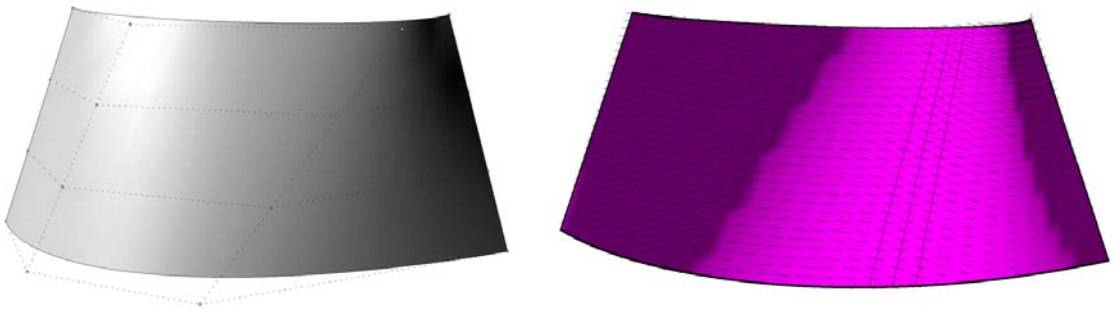


Fig. 6.6. Visualisation of the output of the surface analysis component

isometric mapping

All surfaces with zero Gaussian curvature are rightfully called developable surface since they can be developed to the plane. This means that there is a one-to-one isometrically mapping of the surface to the plane in such a way that all curves that correspond in the mapping have equal lengths and the area of the development is equal to that of the developable surface. For surfaces which are nondevelopable there are multiple methods of developing them to the plane, however, each of these methods all have a discrepancy in the development related to the original surface. For instance, an isometric mapping of a double curved ruled surface, a nondevelopable surface, will preserve the length of the rulings, but the other curves and the area are not corresponding; also recall the difference in the mapping of the earth surface to the plane with the Mercator projection and the Gall-Petersen projection.

The isometric mapping component projects rulings of the input surface to the plane, providing that the surface has rulings, or else the user will be notified that no development can be generated. Based on persevering the length of the rulings and the mutual angle and distances between subsequent rulings by vector mathematical operation, the development is generated, Figure 6.7. Also, the area of both the input surface and the development are provided by the component and compared numerically.

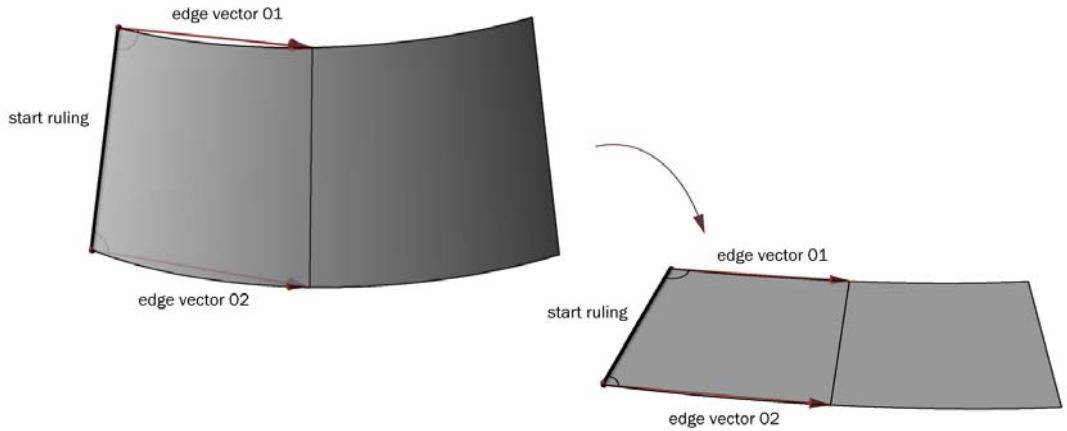


Fig. 6.7. Visualisation of the output of the surface analysis component

curve of regression

A ruled surface follows the notion of linear interpolation; every isoparametric curve – a curve over the surface with a constant u value – is a straight line segment. This follows directly from the parametric representation of a ruled surface

$$\mathbf{S}(u,v) = \mathbf{C}(u) + v\mathbf{a}(u) \qquad \text{(Eq. 6.7)}$$

where $\mathbf{a}(u)$ is the direction vector of the rulings.

If a space curve $\mathbf{C}(u)$ is a curve of regression of a ruled surface, then the rulings of the surface are tangent to this curve. As noted in Section 4.3.2, with the exception of the plane, cylinder and cone, every developable surface has a curve of regression. A possible way to obtain the curve of regression it to firstly develop the surface into the plane. Secondly, the image in the plane of the extended generators of the rulings of the surface needs to be considered, Figure 6.8, top. The pattern generated by these lines marks an additional curve which is called the envelope of the family of lines. Generally, this envelope of a one-parameter family of lines is a curve that touches every member of the family tangentially[129]. Its inverse image under the development is the curve of regression[130]. Inversely, computing the development of a surface can also be based on generating the curve of regression and integrating the differential equation of curve $\mathbf{C}(u)$, parameterised over its arc length[131].

In order to surpass the relatively complex generation of the inverse image, the curve of regression can also be modelled directly by deriving the envelope of the semi-infinite rulings of the developable surface. A point on the envelope of the rulings of a developable surface is generated by the intersection of adjacent rulings with respect to the limit of the distance between these rulings going to zero.

Taking double curved ruled surfaces into account, these intersection points are only valid as approximations of interpolating points of a curve of regression when the distance between the rulings and the warping of the surface patch between the rulings is limited to certain tolerances. In other words, when a ruled surface is nondevelopable, the rulings do not intersect by definition and a theoretical curve of regression based on ruling intersections cannot not be distilled.

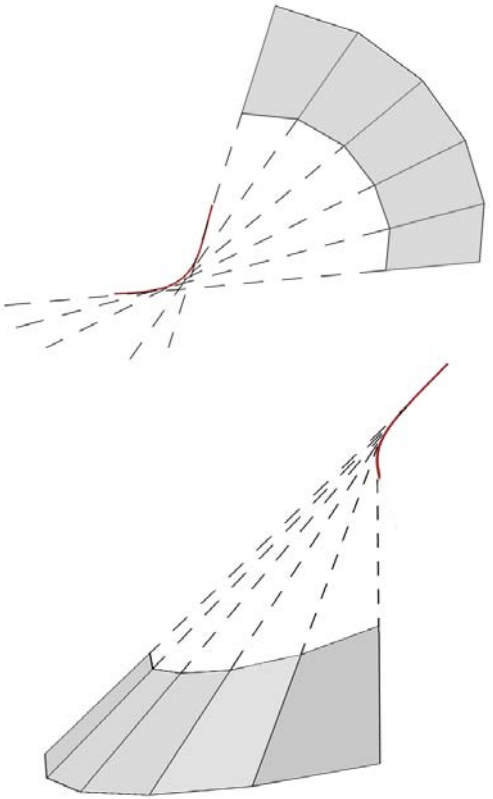


Fig. 6.8. Top: The planar envelope curve of the set of rulings of the development.
Bottom: The curve of regression of the discrete developable patch

[129] Wolfram Mathworld, Envelope
[130] Wikipedia, Tangential developable
[131] Pottmann, Computational line geometry

As the component limits the number of rulings by taking a discrete set of rulings, the theoretical ruling intersections are not found. Therefore, the intersection points are approximated by the points where the semi-infinite rulings are the closest to each other. As such, an approximation of the curve of regression is distilled, Figure 6.8, bottom. That this set of points denotes the curve of regression is based on the fact that in the directions diverging from the curve of regression – the positive and negative tangent direction of the curve –, the distance between the rulings increases, except for planar and cylindrical surfaces.

Figure 6.9 clarifies how the component generates the curve of regression. Firstly, for all rulings, it calculates the two closest points, each on one of the adjacent rulings under consideration which denote where the rulings are closest to each other – when the distance between the rulings is zero, these points coincide. For the first and last ruling, these points are added to the set of intersection points; presented in Figure 6.9 with a circle around the point. Secondly, the intermediate intersection points are defined by taking the mid points between the closest points. The curve of regression then spans between the two points on the first and last ruling and is interpolating the intersection points.

Subsequent intersection points denote a direction of the curve of regression if they are conform the definition of developability; when this direction is, within certain tolerances, equal to the related ruling direction, the two points are part of the valid set of interpolation points of the curve of regression.

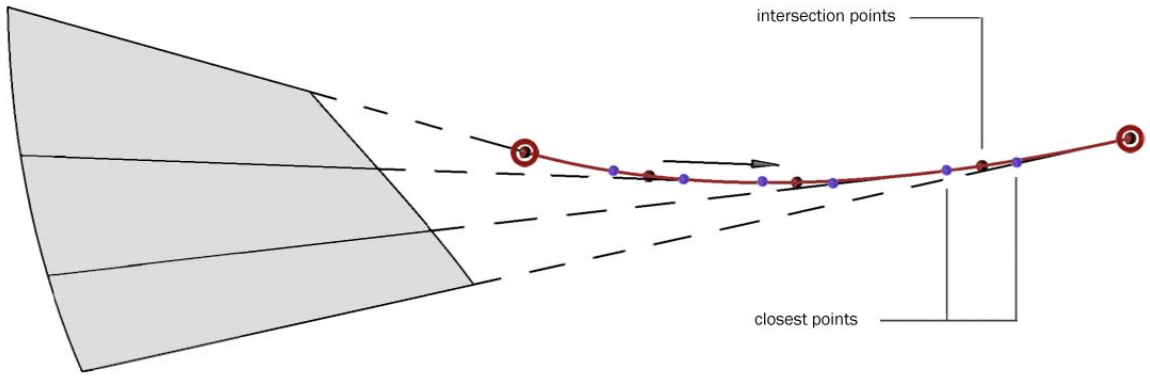


Fig. 6.9. Visualisation of the output of the surface analysis component

surface generation components

The set of surface generation toolbox compartment contain two approximation methods and six direct modelling methods to generate ruled and/or developable surfaces. The main goal of these components is to allow for the generation of various types of these surfaces based on different input parameters and boundary conditions.

spline approximation

When a double curved surface is remodelled to approximate a developable surface, there are various possibilities to base the approximation on. The search for an approximation can be based on for instance maintaining a global ruling direction, persevering a global shape or fix local definitions or conditions. Concerning the latter, the spline approximation component takes one of the curved edges of a double curved ruled surface and extracts the surface rulings at subdivision points of this curve. Subsequently, the rulings are rotated individually around the curve tangent directions at these points and as such define a new lofted surface, Figure 6.10. In order to find a set of rotated rulings which define an approximated developable surface, an optimisation algorithm can be applied in order to validate all rulings for their compliance to the conditions for developable surfaces, e.g. having a minimal warp angle between the top and bottom normal vector along the ruling.

In Grasshopper, an evolutionary solver called Galapagos can be accessed. This component takes a number of numeric input values via sliders and requires a numeric parameter which serves as a fitness value. In the case of optimising the values of every single ruling rotation angle, the fitness value is equal to the summed warp angle of every ruling. Generally, when the initial rotation angles are set to zero for all rulings, the optimal solution will maintain a global ruling direction which does not divert much from the ruling directions of the input surface. As such, it is prevented that the surface definitions converges to a cylindrical surface. When the ruling directions of the first and last rulings are not added to the set of input values, and are therefore not part of the set of sliders for which an optimal value is searched, their direction remains equal to the direction of the straight edges of the input surface and the approximation has three coincidental edges with the input surface. Generally, disallowing rotation of the edge vectors does not yield to a developable surface, however results are better when only one edge direction is fixed to the edge direction of the input curve.

To limit the genome size, the number of rulings needs to be limited and the step sizes in the allowable rotation domain are set to 0.01 rad. Increasing the number of rulings and decreasing the step size would generate a more accurately modelled developable surface, but will result in longer calculation time as well. Due to the fact that the component will only approximate an developable surface, it is advised to distil the edge curve of the approximated surface and use this approximated spline and the fixed curve on the other side of the surface as input curves for the ruling developable surface component, see below.

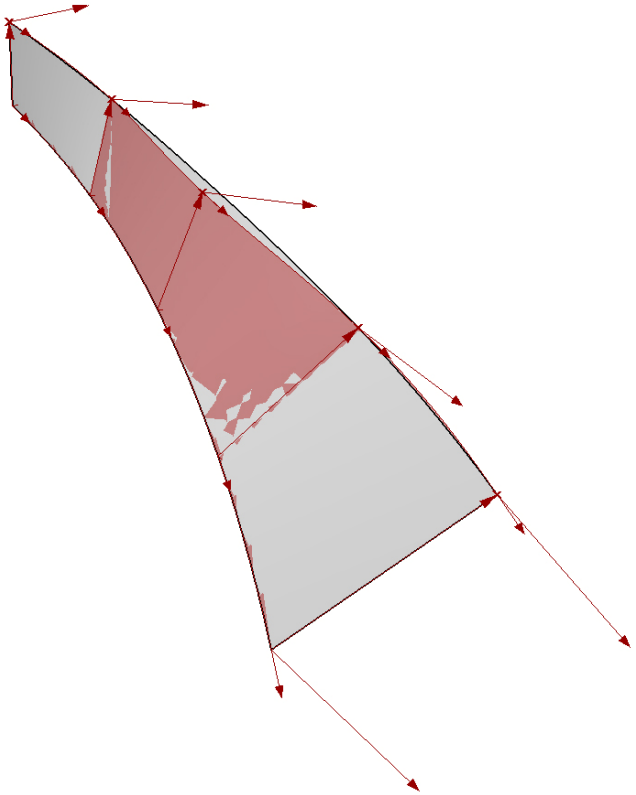


Fig. 6.10. The figure shows how one ruling is rotated around the tangent vector at the starting point of the ruling vector on the fixed curved surface edge

connecting developables

As mentioned before, each developable surface has a single tangent plane along each entire generating line making it possible to connect developable surfaces with each other with a G1 continuity at this line. To properly connect two adjacent developable surfaces, a shared surface generator should be correctly aligned and the surface normals of each patch at that boundary need to be parallel. When two separate patches, not necessarily with parallel edges, need to be connected by a third developable patch, both input patches should have straight edges on the side where they are connected and these edges need to comply with the rules of developability, i.e. are surface generators.

Based on the tangent directions at the edges of the input surfaces, two space curves can be defined which generally allow for the definition of a developable surface between them. The tool, however, does allow for connection of any ruled surface, but double curved ruled surface patches will generally not yield in a fully defined connecting surface since the rulings at the end do not follow the conditions of developability.
Figure 6.11 shows two tangent lines for both surface patches which define a plane which is equal to the tangent plane of the edge ruling. The two planes intersect each other in the dashed intersection line. The two mid control points of the space curves are then defined by creating points on a line from the edge ruling end points to the intersection points at a certain *t* value. If the direction of these lines is opposite to the tangent vectors, then these points are mirrored over the edge ruling in the plane. Together with the two starting points of the tangent lines, these points define the space curve based on four control points defining another edge curve of the connecting developable.

ruled surface

In architectural design, ruled surfaces are generally modelled by connecting corresponding points of two space curves with straight lines. Generating ruled surfaces by transposing a ruling over a directrix curve based on parametrical input is applied less frequently. Nonetheless, this type of surface generation may serve as a starting point of designing with mathematical surfaces.

By denoting the functions for the directrix curve $\mathbf{C}(u)$ and the direction vector $\mathbf{a}(u)$, the rulings of the ruled surface can be generated based on a set of input parameters *u* and *v*. The functions can easily be given by importing the parametric definitions for the (x,y,z) values of the directrix and the direction vector via the inbuilt Grasshopper panel component to an expression component with a single variable *u*. The parametric ruled surface component then combines these inputs together with the *v* value, generating the rulings and the lofted ruled surface, as exemplified in Figure 6.12 for a helicoid.

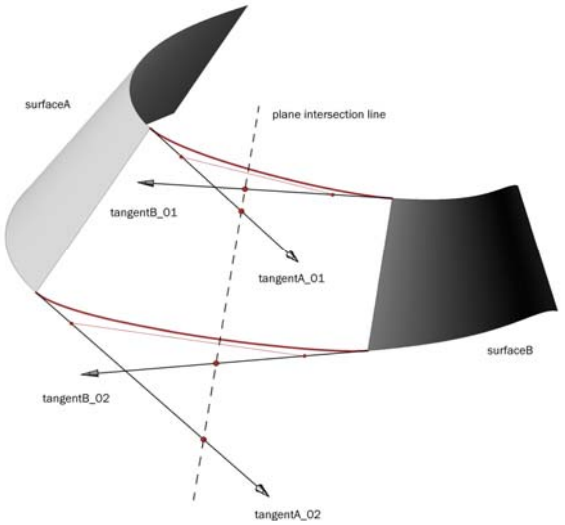


Fig. 6.11. Construction of two space curves defining a developable surface patch connecting two input surface patches

paper strip

As mentioned before, a developable surface is a special kind of surfaces that behave like paper if it is bent or twisted without tearing or stretching. The paper strip component generates a digital paper strip, based on an algorithm by Lorenz Lachauer as posted on his blog*.

The algorithm takes a starting direction for the first ruling which is perpendicular to the initial curvature direction. A third vector, being the cross product of these vectors, is denoted as the ruling rotation axis, Figure 6.13, left.

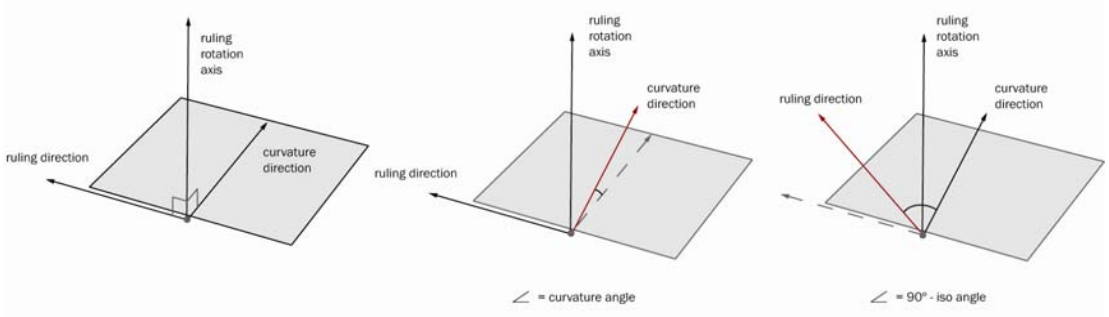


Fig. 6.13. Visualisation of the algorithm of the vector mathematical definition of the ruling generation of a paper strip

The right two diagrams in Figure 6.13 show how every next ruling direction is distilled by rotation operations over two axes; 1) the rotation of the curvature direction vector over the initial ruling direction by the related curvature angle and 2) the subsequent rotation over the ruling rotation axis over 90° - iso angle. For every segment this algorithm is recursively executed, resulting in a set of rulings defining the developable paper strip surface.

By varying the width and length of the paper strip as well as the local curvature angle and iso angle, various paper strip configurations are possible. Obviously, this component is difficult to use as a design tool, but it serves its purpose by presenting a developable surface as a digital equivalent of a paper strip.

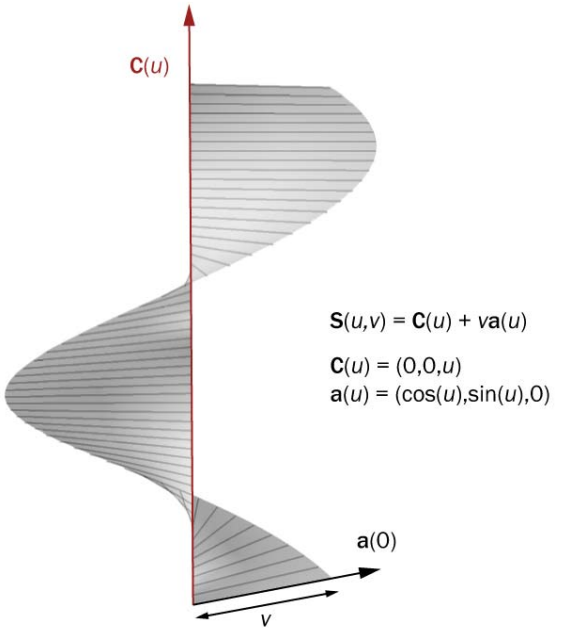


Fig. 6.12. Parametrically defined helicoid based on the parametric ruled surface component

* http://eat-a-bug.blogspot.com – d.d Monday 16 August 2010

cylindrical surface

Cylindrical surfaces can be generated by various operations; parallel extrusion and rotation of a generator around a parallel axis are two possible methods of defining this type of developable surfaces. Respectively, the extrusion direction and the direction of the generator define the orientation of the surface, where the shape of the sectional input – usually a closed circle – defines the shape of the surface.
The standard Cylinder component in Grasshopper only allows for closed circular surfaces to be generated where the surface orientation is perpendicular to a predefined base plane. With the custom developed cylindrical surface tool it is possible to have any curve, planar and nonplanar, as input. Based on a user defined direction and height, a cylindrical surface is generated.

conical surface

Similar to a cylindrical surface, a conical surface can be generated by various operations and also for this developable surface type, the functionality of the custom developed component is enhanced in relation to the standard Grasshopper component. The conical surface component also allows for noncircular and nonplanar input of the sectional base curve. However, in lieu of the direction and height as input parameters, the conical surface takes a 3D point as input parameter, denoting the cone's apex.

tangent developable component

Besides the plane, the cylindrical and conical surface, a tangent developable surface is one of the basic types of developable surfaces, mathematically described as

$$\mathbf{S}(u,v) = \mathbf{C}(u) + v\dot{\mathbf{C}}(u) \qquad \text{(Eq. 6.8)}$$

Geometrically, this implies that the vector direction at a value u of the space curve $\mathbf{C}(u)$ is equal to the tangent of $\mathbf{C}(u)$. Recall Section 3.2.1 where it was noted that equation of the tangent vector of a space curve is given by

$$\mathbf{T}(t) = \mathbf{C}'(t) = (x'(t), y'(t), z'(t)) \qquad \text{(Eq. 6.9)}$$

Since generally the input curve is not defined by a parametric equation, the functionality of the tangent developable component is based on the TangentAt() method of the RhinoCommon SDK which calculates the unit tangent vector $\hat{\mathbf{T}}(u)$ of the input curve at a given curve parameter. Similar to the conical surface component, the user denotes the length of the rulings for every discrete u value, giving a set of rulings which define the tangent developable surface, Figure 6.14. As such, this component is essentially the inverse of the curve of regression component.

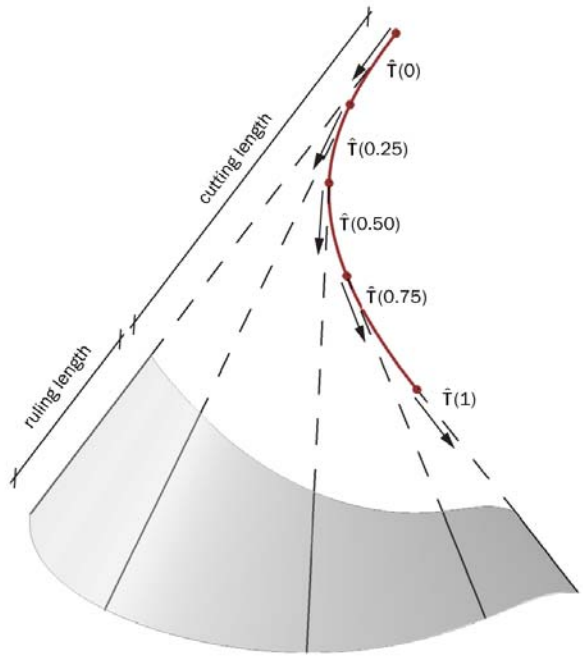


Fig. 6.14. Discrete definition of a tangent developable surface

ruled developable surface component

According to Sun and Fiume, in developable surface modelling, the classic problem of constructing a continuous developable surface is in connecting two given space curves[132]. To ensure that rulings between space curves are also generators of a developable surface, the tangents at the intersection points of the rulings with the space curves need to be aligned so that the cross products between the ruling and the two tangents are co-linear[133]. When no such set of two points can be distilled, the curves will not define a developable surface patch. In this case, either one curve or both curves must be altered to make the pair compatible in the sense of developability[134].

From a general perspective, it is not possible to construct a single developable surface patch from two arbitrary curves in space, nor are there a set of patches with tangent continuity which can. These patches will in general have a discontinuity in surface tangency at their joined ruling, resulting in a G^0 continuous connection. However, when it is enforced to use singular and subsequent steps to find intersection points of a ruling with the input curves, a single developable surface patch can be generated at the cost of deviating from the original input curves between these points. When the step sizes are taken sufficiently small, the deviation can be limited to allowable values.

There are several CAD applications which support the generation of developable surfaces between input edge curves also taking this deviation into account. Rather than developing the surface from defined cylindrical, conical or planar elements, usually algorithms are used approximating a developable surface based on discrete steps[135]. For instance, the Rhino Loft command allows for a developable loft which creates a developable surface or polysurface from each pair of curves. However, since the functionality of this command is limited, also a DevSrf plug-in has been developed, which is available via the Rhino Labs Tools website[136]. This plug-in displays the developable ruling lines between a pair of rails where the tangents at the ends of the ruling lines are co-planar within a twist tolerance. The approach used for the DevSrf plug-in follows the same routine as the mathematical vector approach by Nolan.

[132] Sun, A technique for constructing developable surfaces

[133] Konesky, Computer aided design of developable surfaces

[134] Nolan, Computer-aided design of developable hull surfaces

[135] Shelden, Digital surface representation and the constructability of Gehry's architecture

[136] McNeel Wiki, Rhino 4.0 labs tools

Rather than dealing with the complexities of multiple roots and domains resulting from expressing the ruling condition in one equation, Nolan adopted an approach with a search technique. Firstly, the boundaries of the surface are expressed by spline-approximating polynomials, given a set of points for each boundary. Secondly, rulings are computed at small intervals over the surface by an iterative process which searches the boundary curves for the line which satisfies the ruling definitions in terms of normal vectors[137] - Equation 6.10, which equals the definitions denoted in Section 5.3.1.

$$\mathbf{N}_1 \times \mathbf{N}_2 = |\mathbf{N}_1||\mathbf{N}_2| \sin\theta \qquad \text{(Eq. 6.10)}$$

The approach was mainly successful for simple curve definitions and therefore for the generation of simple surfaces. The major problem was that rulings might either cross or fan out, yielding an unrealistic or non-usable surface.

The ruled developable surface component is able to overcome these problems. The algorithms used for this tool are similar to those used by Nolan as they generate the rulings of a developable surface by searching for a ruling direction which satisfies the vector approach such that the tangent and normal vectors at the start and end point of the rulings lie in a plane. However, the boundary curves are not remodelled as polynomials, but the curves are directly subdivided in discrete points based on a parametric subdivision method used in Grasshopper. As such, the algorithm shows the possibility of generating developable surfaces by deriving developable rulings.

The following steps define the functionality of the tool: two space curves, defining the edges of the surface, serve as the input for the component, based on which a set of rulings is constructed under the constraint of developability. The algorithms follow two subsequent steps. Firstly, the base input curve is subdivided based on a user-defined number and the secondary curve is subdivided with a much higher subdivision density. Secondly, the algorithms loop through the first set of points, searching for points on the second curve where the ruling $\mathbf{C}_B(u)$ - $\mathbf{C}_A(u)$ complies with the developability condition, Figure 6.15.
The tool does not allow for a ruling to be found with an end point on the second curve with a smaller index than the index of the last point on the second curve. As such, the generated rulings are a result of a directed search algorithm, stepping along both edge curves based on their initial directions. Therefore, it makes a difference which curve is taken as the base input curve, which is the second curve and what the direction of the curves is.

[137] Nolan, Computer-aided design of developable hull surfaces

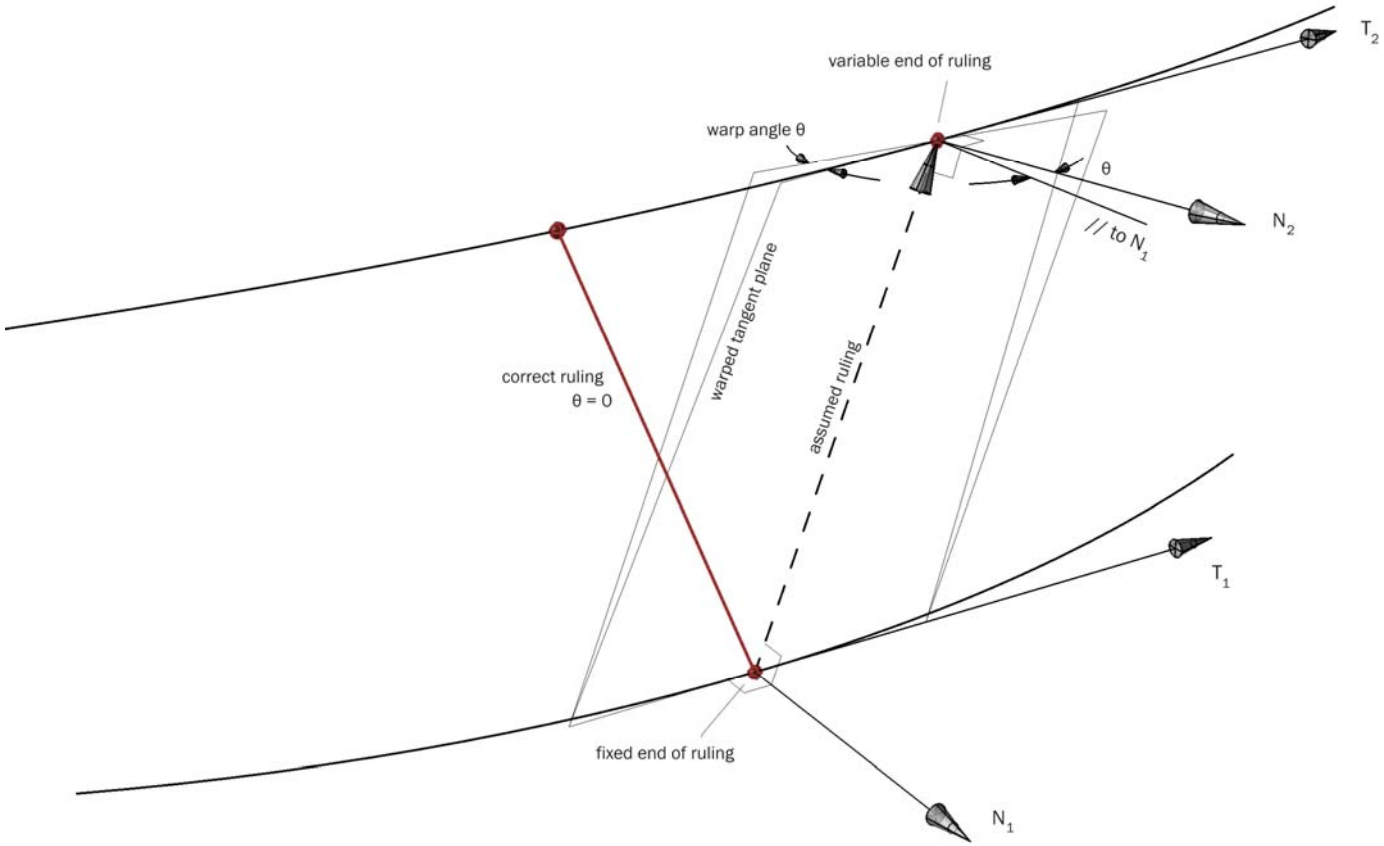


Fig. 6.15. The search strategy of the method adopted by Nolan; a warped tangent plane is modified to a flat tangent plane by fixing one end of a ruling and finding the point of intersection with the other boundary curve that satisfies the ruling condition

surface patching component

surface patching

Surface panels naturally introduce opportunities for introducing overlaps between individual sheets, slight discontinuities in the smoothness of the overall shape or the possibility of slight warping of the material sheet. Generally, these effects are not identifiable at the building scale level, see also Section 7.2.2. This potential relaxation of constraint conditions as design heads toward actual fabrication is of course beneficial to the design process and provides in essence a factor of safety in the schematic design of the project form[138].

The surface patching component takes the development of a developable surface as an input and maps a predefined pattern on to it. Figure 6.16 shows, a Guggenheim-like pattern with a shift of individual panels of half the width of the panel size based on the highlighted set of lines. This set of lines is firstly bounded by a bounding rectangle which denotes the translation of each patch based on the width and height of the rectangle. Multiplied by a scaling factor, the pattern is translated over another bounding rectangle defining the rectangular shape around the development. As such, a pattern of flat surface panels is defined which could be sent to a CNC machine for cutting the individual panels.

Subsequently, the pattern on the development is reversely mapped to the initial developable surface, creating a surface pattern following the orientation of the pattern of the development. By rotating the development in the XY plane, the orientation of the pattern on the developable surface follows dynamically. As such, the tool enables to develop representations of sheet materials, which reflect fabrication constraints without imposing additional constraints resulting purely from limitations of the geometric representation itself.

As the pattern potentially could be based on a large number of parameters and since desired output factors such as visual appearance or cutting loss during fabrication of the panels are largely influenced by the pattern definition, it is omitted to investigate other surface patching patterns further in this research as this might be a research on its own. Related to this, it might be of interest to research the possibilities to optimise panelisation of developables surfaces, for instance in relation to CNC operations, minimising cutting losses and material properties.

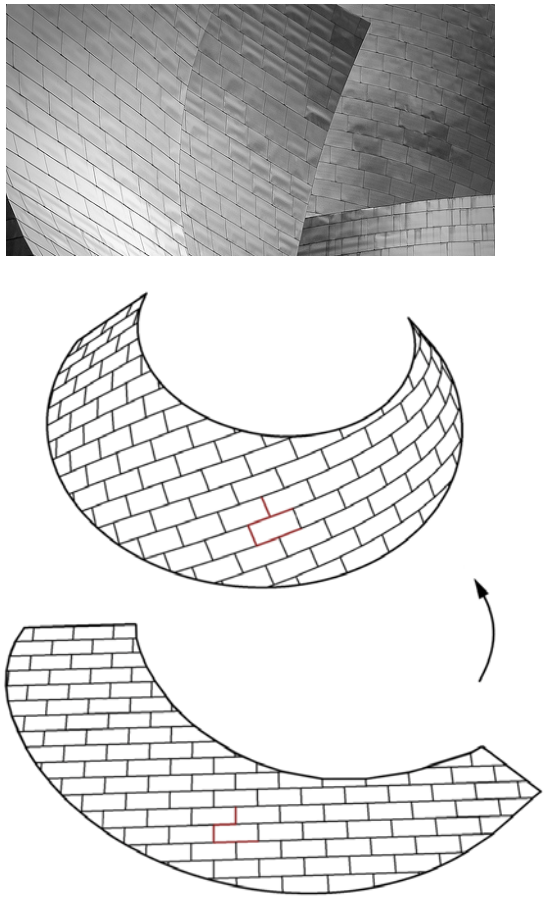


Fig. 6.16. Mapping of a pattern from the development to a developable surface in R^3. Top image: Façade detail of the Guggenheim Museum in Bilbao. Photograph by Jorge Estevez Garcia

[138] Shelden, Digital surface representation and the constructability of Gehry's architecture

6.2.2 component test cases

Although the components serve as a proof of concept, below, a number of basic tests are presented to verify the functionality of methods used by the components. Generally, the functionality and the derivation of the outcome of the components are dependent on vector mathematics and RhinoCommon SDK methods, see Section 6.3. Therefore, firstly, definitions related to vector mathematics are given. Secondly, if applicable, the outcome of the components is compared to the outcome of Rhino commands.

Since basic vector operations are essential for the functionality of the tools, Figure 6.17 and Figure 6.18 present Grasshopper definitions comparing the numeric output of the standard Grasshopper cross product, dot product and vector angle components with the custom developed VectorAnalysis component and show that their outcome is identical.

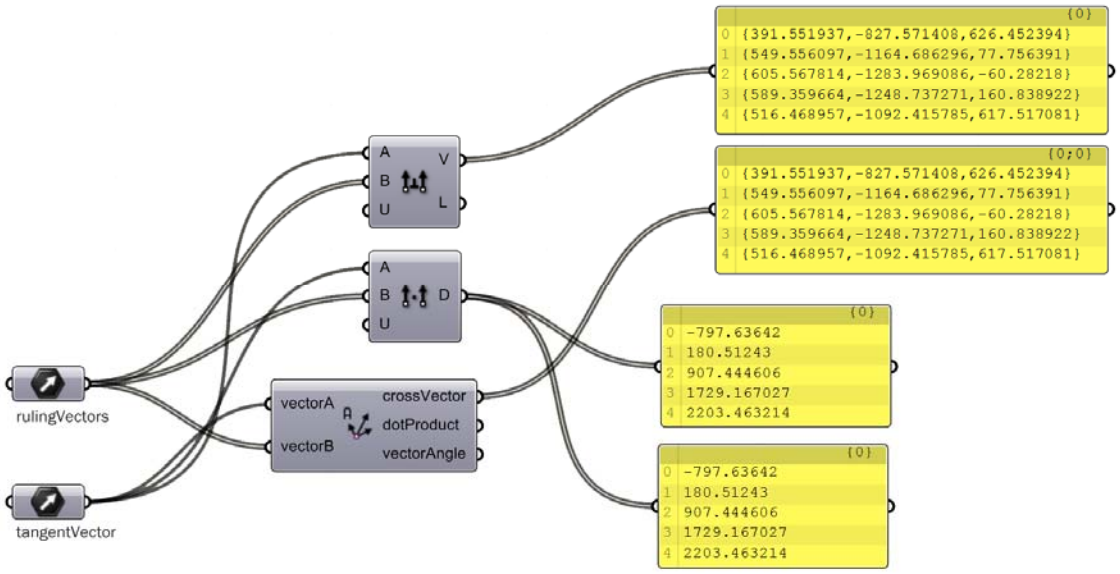


Fig. 6.17. Comparison of the numeric output of the cross product and dot product of the standard Grasshopper component and the VectorAnalysis component

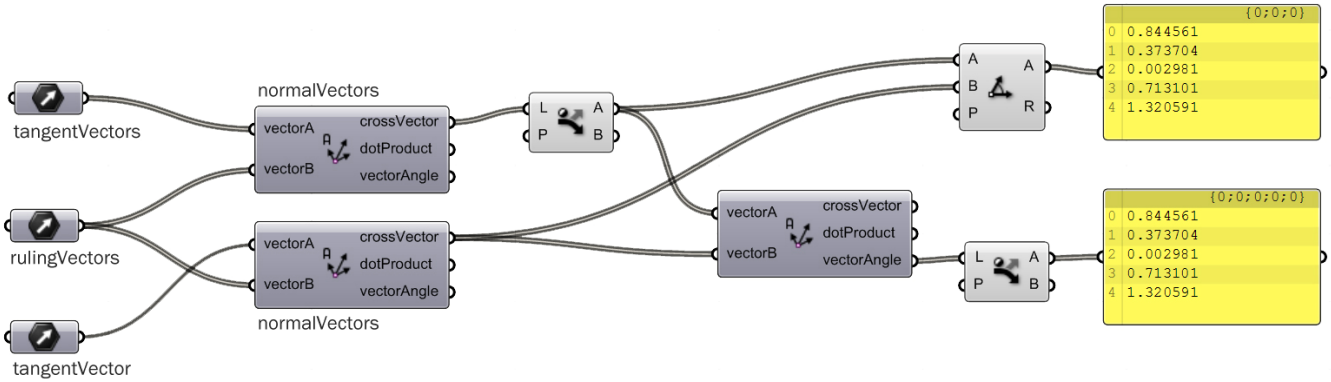


Fig. 6.18. Comparison of the numeric output of the vector angle of the standard Grasshopper component and the VectorAnalysis component

The cross product, dot product and vector angle methods are used by a number components for which normal vectors of surface rulings are generated and/or checked, such as the SurfaceAnalysis and RulingDevelopable component. Related to this, these components take into account the value of the warp factor of the rulings, see also Section 7.2.2, in order to check local compliance with conditions of developability. The warp factor is defined as the warp angle between two normals at the end of the rulings divided by the ruling length. In the aforementioned components, this warp factor is set against a tolerance value in order to determine the level of developability.

Figure 6.19 shows the outcome of a validation test for the results of the RulingDevelopable component. The curves at the top of the figure indicate which input curves are used for this test. From left to right; arbitrary curves (2x), straight lines under an angle, edge curves extracted from a developable surface and circular curves extracted from a straight cylinder. The test aimed at graphically showing where the warp angle of a ruling has its minimal value. The rulings under consideration span from a point on the bottom curve at a t value of 0.5 to a discrete but high number of points on the top curve over a domain [0,1]. As such, it is obvious that for a cylinder, containing straight rulings, the warp angle is minimal at the t = 0.5 value, as shown in the right graph of Figure 6.19.

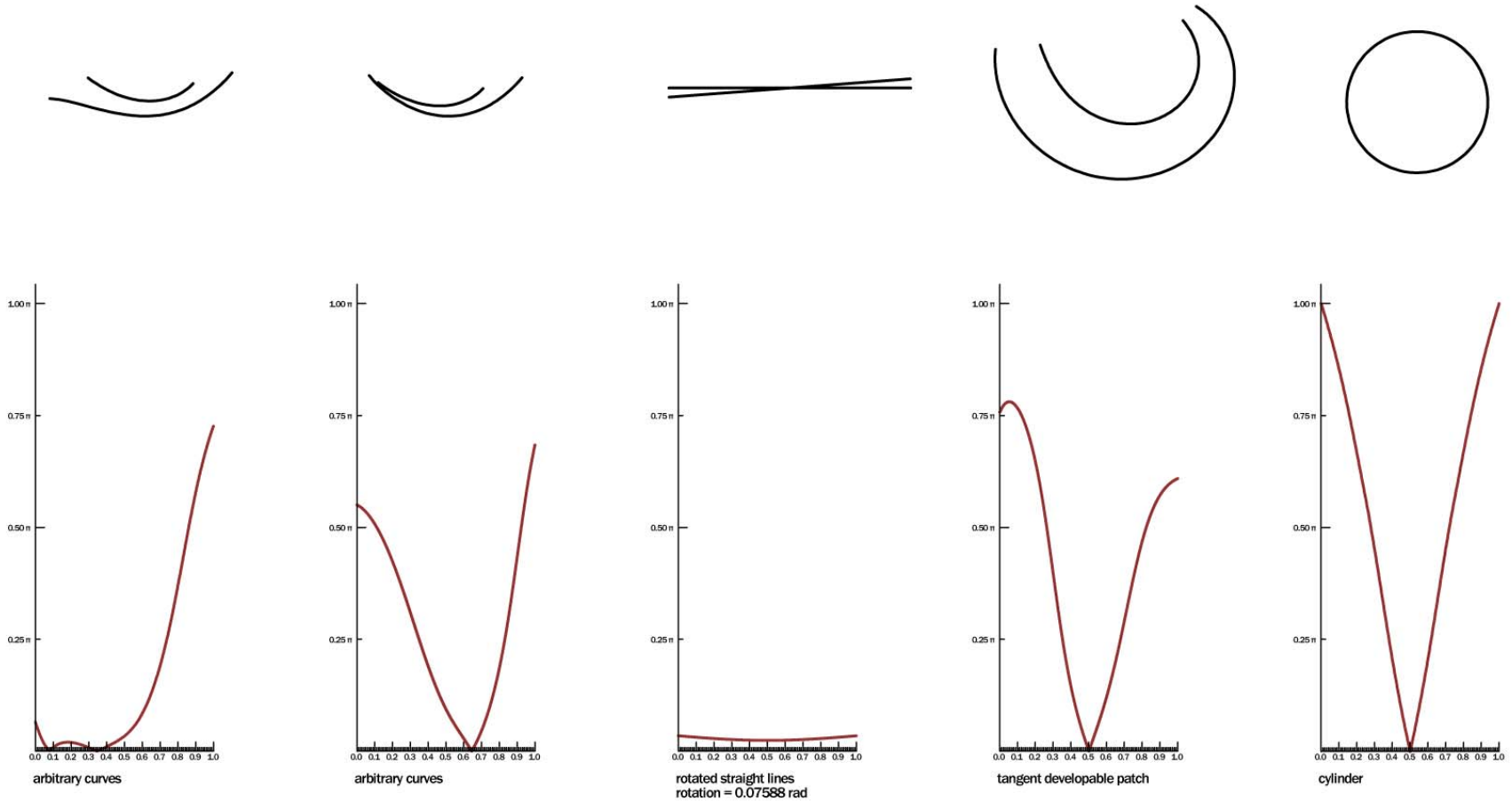


Fig. 6.19. Graphical representation of the location of the minimum warp factor value of a ruling between a set of two curves directing from a point on the bottom curve with a *t* value of 0.5

Figure 6.19 also shows that it might be possible that two values comply with a local minimal value, as indicated by the left graph. The RulingDevelopable component deals with this by generating a ruling to the first *t* value which is related to the minimal value, unless previously defined rulings are directed to a point on the top curve with a higher *t* value. Additionally, the graph related to the straight lines under an angle indicates that the minimum value is also at $t = 0.5$, but that this warp value is equal to the rotation angle between the curves. As such, although the ruling definitions are based on a set of discrete starting points, finding rulings conform to the global minimum warp appears to be valid.

Figure 6.20 shows the geometry of the aforementioned developable surface patches as the outcome of the RulingDevelopable component related to the output of the Rhino curvature analysis command. The monochrome output is a result of the input parameters of the curvature range of the Rhino command and indicates that the Gaussian curvature is indeed zero, or at least approximates this value.

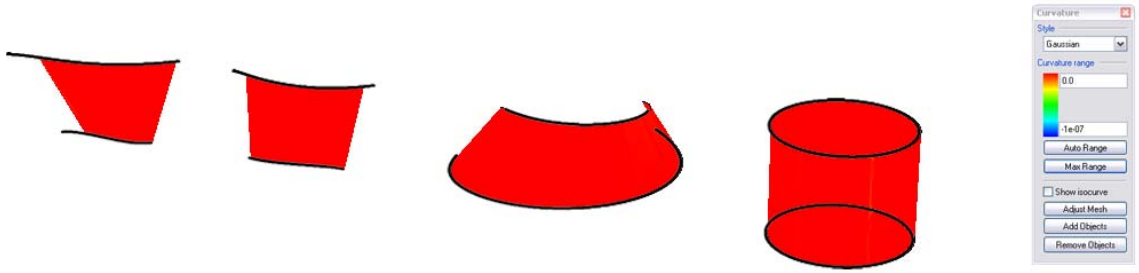


Fig. 6.20. Curvature analysis of the developable surfaces generated by the RulingDevelopable component

Also the SplineApproximation component takes the warp angles of rulings into account in defining a developable surface based on a double curved ruled surface as they provide for the definition of the fitness function which drives the Galapagos evolutionary solver, Figure 6.21. Checking the output of this component shows that the number of segments, i.e. the number of rulings minus one, is of great influence to the level of developability. In this test case the input surface is subdivided by twenty rulings, individually rotated by parametrically defined angles which are the input parameters for the Galapagos component. As such, locally rules of developability are complied with, however, in between the rulings, the loft method generally creates a surface which is slightly off to that of a developable. Therefore, it is stressed out that the SplineApproximation component primarily serves its value in approximating the free curved edge. This curve and the initial fixed curve can then be used as input curves for the RulingDevelopable component to generated a better approximation of a theoretical developable surface.

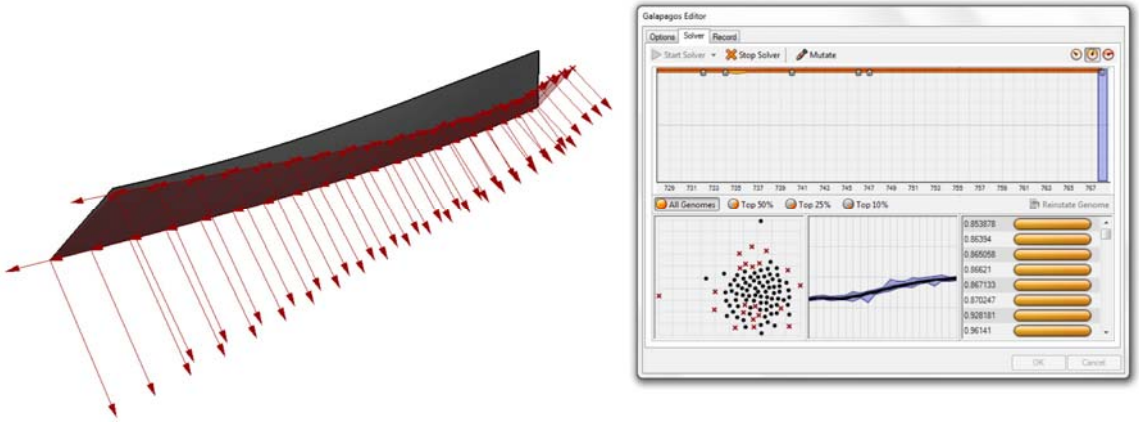


Fig. 6.21. Screenshot of an evolutionary solver run with Galapagos based on the fitness function provided by the SplineApproximation component

As rulings are mapped based on preserving angles and direction between adjacent ruling vectors, the IsoMapping component basically has its own validation output parameter as it numerically compares the area of the input surface with that of the development. Therefore, this component, as is its counterpart, the SurfacePatching component, is also primarily based on vector mathematics.

Other components also do rely on vector operations, but incorporate RhinoCommon SDK functionality as well. For instance, in order to surpass to code the definitions of differential tangents of a NURBS input curve, the TangentAt() method is used at discrete locations along an input curve in order to define a tangent developable surface by the TangentDevelopable component.

Related to the CurveOfRegression component it can be stated that the curve of regression is theoretically equal to the input curve of the TangentDevelopable component. Logically, the curve definitions are not identical, as the input curve generally is a modelled NURBS curve based on control points, whereas the curve of regression output is a curve interpolating a number of points which denote where the deviation from the warp angle of the input surface is less than a user defined allowable deviation angle, Figure 6.22. However, local coordinates and directions are, within this tolerance, in accordance with each other.

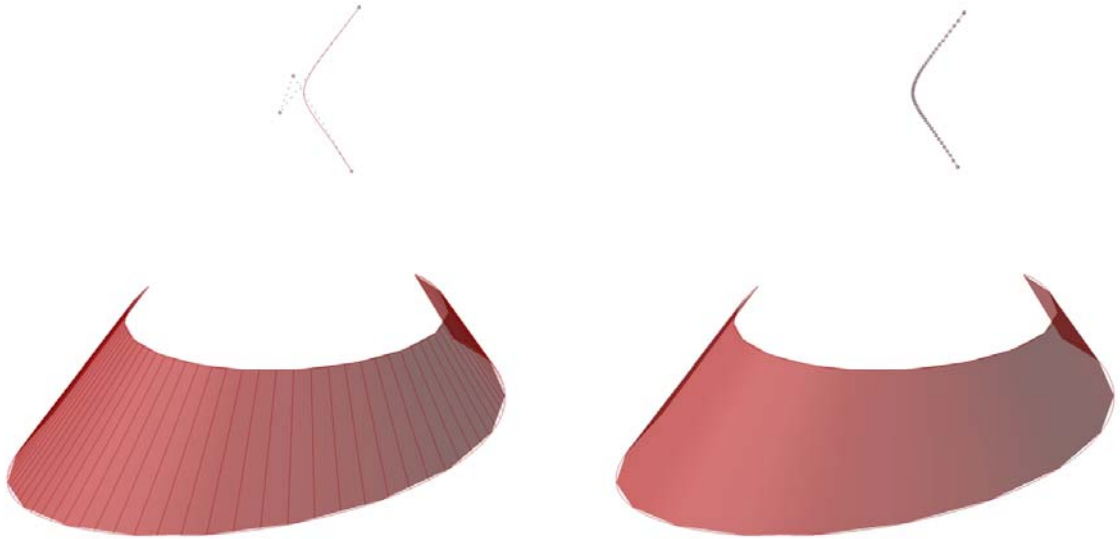


Fig. 6.22. Left: the input curve and its tangent developable surface. Right: the copied tangent developable surface and its curve of regression

Similar to the CurveOfRegression component, the SurfaceCoupling component relies on RhinoCommon SDK methods related to intersections, e.g. PlanePlaneIntersection and LineLineIntersection. Ideally, these methods are coded by the developer, which would allow for better insight in their functionality.

For the SurfaceCoupling component, it is of primary interest to define space curves, which function as input curves for the RulingDevelopable component, with tangent vectors at their end points which are identical to the tangents at the end points of the curved edge curves of the input surfaces. Figure 6.23 displays the tangent vectors and shows the numerical validation of their interrelated direction as the angle between them is equal to pi rad.

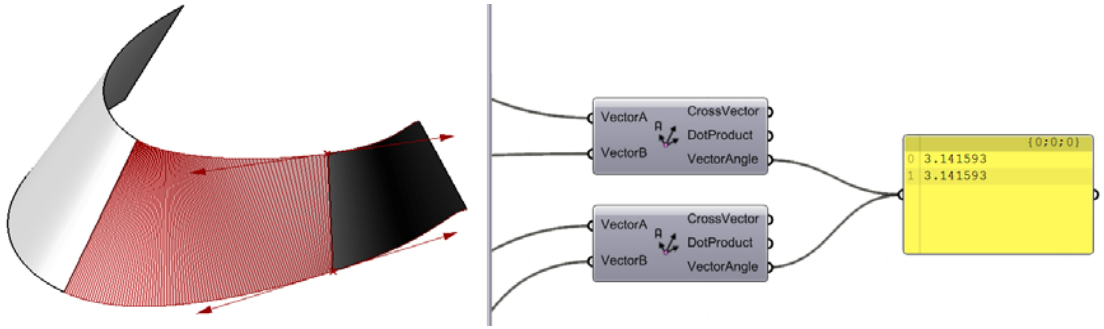


Fig. 6.23. Visualisation of the tangent vectors at the common ruling between an input developable surface and a developable surface generated by the RulingDevelopable component

Ultimately, Figure 6.24 shows definitions of three parametrically defined ruled surfaces of which only the cylinder is a developable surface. To the right of every surface, the parametric definition of the directrix curve and the vector in *v* direction is given. After the surfaces are reparameterised, the *u* and *v* domains are [0,1;0,1]. The points on the surfaces denote the location on the surface at (*u*,*v*) = (0,0.7) and serve as definitions for the validation of this component.

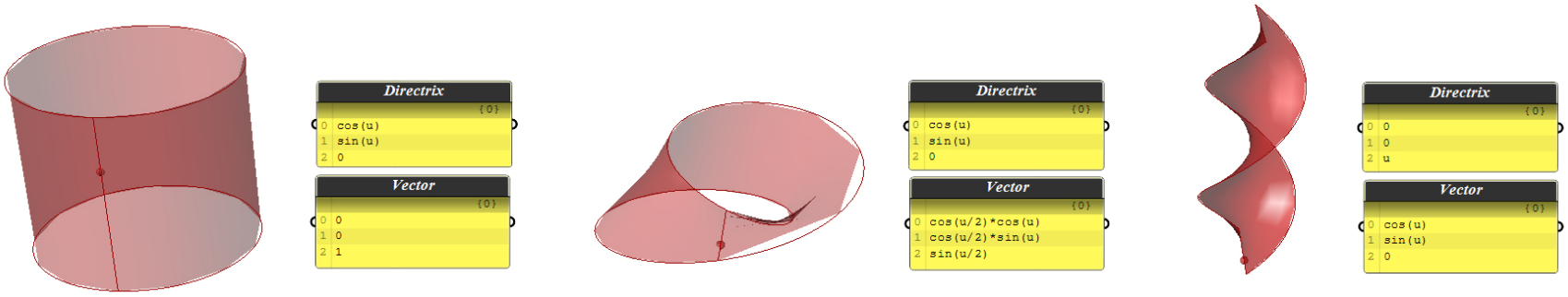


Fig. 6.24. The surface visualisation and the parametric definition of a cylinder, Möbius strip and helicoid

6.3 component-oriented programming of the toolbox

The components of the toolbox which support the design approach have been developed in the .NET plug-in RhinoCommon SDK in the C# language. The development code structure is based on inheritance of the Grasshopper.Kernel.GH_Component which is accessible by using the Grasshopper reference. In order to clarify the structure of code definitions, this section presents the development of the toolbox from a programming perspective, describing the sequence of methods of the code and its functionality.

6.3.1 software specifications

The tools have been developed as custom components for Grasshopper version 0.8.0004 in Rhinoceros 4.0 SR8 based on the RhinoCommon SDK. The code has been generated in the Visual Studio 2008 Professional Edition software development environment in the C# language*.

6.3.2 programming specifications of the toolbox

In using the RhinoCommon SDK assembly reference, a large array of intrinsic Rhino objects and functions is available as a set of respectively object types and method members of the assembly classes. Although this allows the developer to build code based on predefined efficient algorithms, the functionality of these algorithms is not always directly clear, possibly resulting in unexpected output. Also, refactoring the base code in order to implement functionality to other software or a stand-alone executable becomes more difficult. On one hand, ideally, geometric operations as defined in the SDK methods are coded by the developer allowing for direct implementation in other software or a stand-alone version of the toolbox. On the other hand however, when the functionality is understood and results are validated, calling these functions may shorten development time to a great extend.

In certain occasions, intrinsic functionality of the toolbox has been rewritten or made part of an individual class in order to allow for other (types of) input and/or output, such as is the case for the basic vector mathematical operations. However, the code for toolbox components also still contains a number of Rhino methods, for instance in order to surpass having to define NURBS curve and surface definitions; e.g. a Loft method is used, taking rulings as input and generating a NURBS defined surface.

* http://www.rhino3d.com/
http://www.grasshopper3d.com/
http://www.rhino3d.com/5/rhinocommon/
http://www.microsoft.com/visualstudio/en-gb

Other SDK methods which have been implemented in the code include:

- the Extract Isocurve method and ProjectCurve method
- intersection methods, such as PointAt and LineLine intersections or PlanePlane intersections
- geometry checking methods, such as ClosestPoint and IsLinear and
- geometry characteristic definition methods, such as TangentAt and CurvatureAt

In order to indicate their general functionality, most of the RhinoCommon SDK methods have been defined as methods in individual classes, categorised in a curve utility set of classes, a surface utility set of classes and a vector mathematics set of classes, see next section.

6.3.3 coding structure of the toolbox components

The RhinoCommon SDK allows the developer to define the custom components in an object oriented programming (OOP) environment in individual classes which inherit from the GH_Component class. This class defines four methods which respectively 1) register the input parameters, 2) register the output parameters, 3) solve the component logic and 4) return the unique Guid of the component. Additionally, in order to create a custom component, the user needs to provide a public, empty constructor which calls the base class constructor and may provide a location for the component icon to be displayed in Grasshopper.

Figure 6.25 graphically represents the structure of the component in pseudo code and shows the three sets of the curve and surface utility classes and the vector mathematics classes which can be accessed from the component classes.

For the full code definitions and related algorithms, reference is made to the ParametricDevelopable Solution file which is sent with this report.

```
Component class : GH_Component
{
	Constructor
	//initialise the component
	"name and description of the component"
	{
	}

	InputManager method
	{
		register input parameters
	}

	OutputManager method
	{
		register output parameters
	}

	SolveInstance method
	{
		//define global variables
		input variables
		output variables
		general variables

		//define the component logic
		if (input = curve OR surface)
		{
			reparameterise input
		}
		perform logical operations
		call utility classes

		//set GH output data
		SetDataList(rulings)
		SetData(developableSurface)
	}

	Component Guid method
	{
		get guid
	}

	Icon Bitmap method
	{
		get component icon
	}
}
```

```
Curve utilities class
{
	perform curve operations
	return points OR curves
}
```

```
Surface utilities class
{
	perform surface operations
	return points OR curves OR surface
}
```

```
Vector mathematics class
{
	perform vector mathematical operations
	return vector OR double
}
```

Fig. 6.25. Pseudo code definitions of a toolbox component

6.4 epilogue: specific component functionality

Regarding the large set of components in Grasshopper, it becomes clear that they individually serve a small but specific goal; e.g. to calculate one number, to generate one geometric object, etc. As such, they allow to be used in different definitions. In other words, paradoxically, their specific nature provides for a general utilisation.

The development of the architectural developable toolbox tried to follow the Grasshopper framework setup in providing relatively simple components which perform basic operations. As such, the user is provided a new set of components which allow for a similar approach in utilisation as the standard Grasshopper components. Firstly, the input parameters are limited in number and generally do not require specific definitions or operations prior to providing data. Secondly, the surface definition components output is primarily focussed on two types of data; 1) the lofted surface defining a developable surface and 2) the rulings on which the developable surface is based on, Figure 6.26. This loft is based on the Loose loft option by which the surface control points are created at the same locations as the control points of the loft input curves, i.e. the end points of the rulings[139]. When rulings join in a single point, which for instance is possible with the conical surface component and the ruling developable component, the Loose loft option allows for the generation of a smooth surface. By taking a sufficient large number of rulings to generate the loft with, the discrepancies between the lofted surface and the theoretical developable intersecting are negligible.

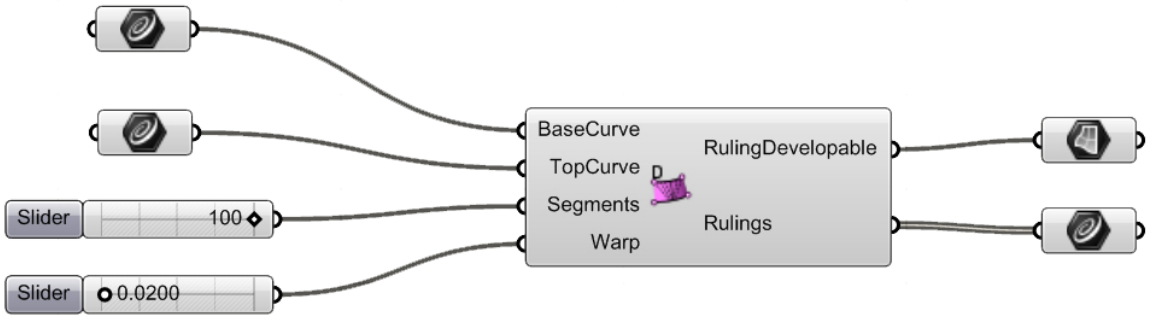


Fig. 6.26. Visualisation of the Grasshopper definition for the generation of a ruling developable surface

[139] Rhino3D, Glossary

7 implementation of parametric developable surfaces

From a digital perspective, the user needs to load the parametric developable assembly in Grasshopper and reference to its libraries in order to use the methods of the different tools. Implementing its functionality in the design process is the subsequent step to the development and validation of the individual components and finalising the installation of the software and deploying the toolbox. This chapter focuses on this next step in utilisation of the functionality of the tools, followed by a set of case studies exemplifying the possibilities of the toolbox.

7.1 introduction: implementation of the toolbox in the design process

One goal of the toolbox is to allow the user to use a set of functionalities to analyse and generate developable surfaces within the proposed design approach. These functionalities are set to provide assistance in analysis and modelling of developable surfaces in different stages of the design process between the initial sketch design and construction, Figure 7.1.

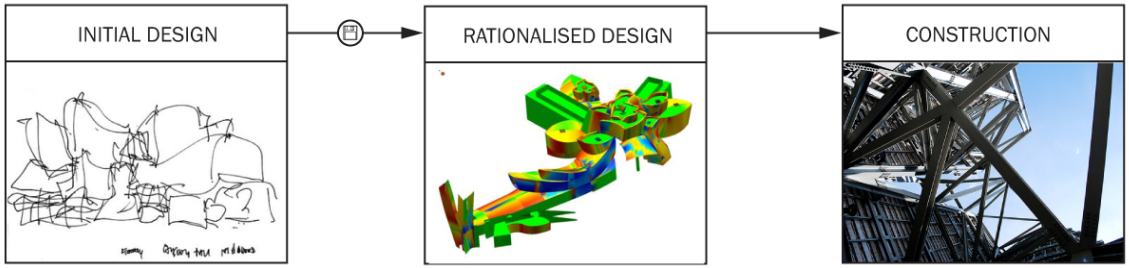


Fig. 7.1. Schematic representation of the location of the digital modelling of developable surfaces in a design process

The analysis tools are best used when digital models have been generated and which need to be analysed for conformance with conditions of developability. The surface modelling components may serve two possible approaches in the design process as a number of tools provide a more theoretical approach in surface definitions, such as the ruled surface component and the tangent development component, whereas other tools allow for a more ‘designed’ surface definition based on specific input. Finally, although underexposed in the whole research, panelisation forms an important step in the transformation from design to product especially related to surface rationalisation. When overall form geometries have been established, the surface patching component provides for a digital link between surface modelling and surface panel definitions.

As such, generally, each individual component can function as a specific tool at a specific moment in the design process. There are, however, components which are dependent on the functionality of other components. One example is the surface patching component which needs a development of a ruled surface as input parameter and therefore needs to be part of a larger Grasshopper definition, linking different components with each other. Either used as a single component or as part of a bigger definition, utilising the tools denote a virtual part of the design process supported by the parametric modelling of developables design approach.

7.2 usability and functionality of the toolbox

This section focuses on the exemplification of the usability and functionality of the toolbox when implemented in a design process. Firstly, the input and output parameters for each component are presented. Secondly, the relation with the practical definitions for developable surfaces are given based on tolerances defined by for instance material properties.

7.2.1 input and output parameters

As mentioned in the epilogue of the previous chapter, the component development aimed at a standardisation of inputs and outputs in order to present and identify the toolbox as a coherent set of tools. As such, the output of the direct modelling surface generating components is always a set of rulings and a lofted developable surface. Inputs are usually related to geometric typologies, such as curves and surfaces and numerical data, such as indices and tolerance parameters. In order to avoid having the user to parameterise the geometrical input, every input curve or surface is reparameterised to respectively a one-dimensional domain [0,1] and a two-dimensional domain [0,1; 0,1] before processing them.

On the next pages, the input and output parameters for each component are defined, including a description of these parameters and their relation with the functionality of the tool. It is indicated that the output of a number of component is dependent on an allowable tolerance. A deviation from the theoretical definition of developable surfaces can be considered when construction methods, dimensions and physical properties of materials allow for flexibility in relation with the condition for developability. In Section 7.2.2, definitions for the tolerance inputs are discussed.

Table 1. Analysis components input and output

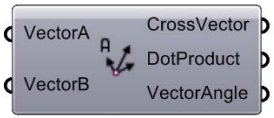
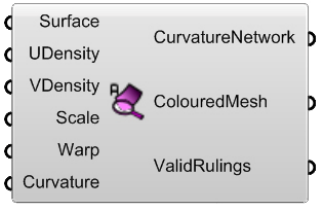
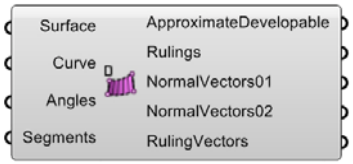
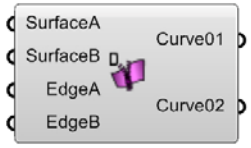
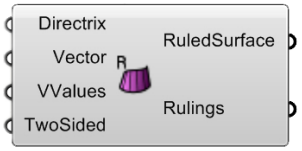
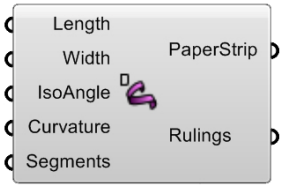
component	*input*	*output*	*description*
	• (list of) vector(s) • (list of) vector(s)	• (list of) vector(s) • (list of) number(s) • (list of) number(s)	The vector analysis component performs three basic vector mathematical operations, defining the cross product vector, the dot product value and the vector angle in radians the between two (sets of) vectors.
	• surface • integer • integer • double • double • double	• list of lines • coloured mesh • list of lines	The surface analysis component shows the network of principal curvature lines, where the lengths of the individual lines indicate the amount of local curvature. For visibility purposes the lines can be scaled by the scale factor. The UDensity and VDensity parameters define the density of the curvature network in *u* and *v* direction. The coloured mesh indicates regions of the smallest of the absolute values of k1 and k2 related to a curvature tolerance and the valid rulings indicate where rulings occur which are conform the condition of developability within the given warp tolerance.
	• surface • integer	• surface • list of lines • string	The isometric mapping component develops a ruled surface to the XY plane. The Segment parameter determines the number of rulings to base the development on. The Area output indicates the numerical difference in area between the input surface and the development.
	• surface • double	• (list of) curve(s)	The curve of regression component generates the curve of regression of an input surface. The input surface type is restricted to ruled surfaces and also excludes planar surfaces, cylinders and hyperbolic paraboloids as no curve of regression can be extracted from these and non-ruled double curved surfaces. The Deviation input parameters sets the allowable local angular deviation between a ruling of the surface and the tangent of the curve of regression at their near intersection point.

Table 2. Surface components input and output

component	*input*	*output*	*description*
	• surface • curve • list of doubles • integer	• surface • list of lines • list of vectors • list of vectors • list of vectors	The spline approximation component generates an approximated developable surface from an input ruled surface. The approximation is driven by one positional fixed curved edge of the input surface and an evolutionary algorithm which searches for the minimal value of the summed angles between the normal vectors. The number of sliders which serve as input for the rotation Angles parameter should be equal to the number of segments. The Segment parameter determines the number of rulings which the surface is based on.
	• surface • surface • integer • integer	• curve • curve	The surface coupling component generates two space curves that span between the edges of two input surfaces. Based on these output curves, a developable surface can be generated with the ruling developable component. The EdgeA and EdgeB parameters indicate the edge number of the input surfaces as denoted in Grasshopper or Rhino.
	• list of doubles • list of doubles • list of doubles • boolean	• surface • list of lines	The parametric ruled surface component generates a ruled surface based on parametric equations of the directrix and the vector direction. Both parameters are best supplied with the standard expression component of Grasshopper which can take as input a string denoting the parametric equations for the ruled surface and a list of *u* values. The VValues indicate the length of the ruling in the vector direction and the boolean TwoSided parameter determines whether the rulings are extended to the negative vector direction as well.
	• double • double • list of doubles • list of doubles • integer	• surface • list of lines	The paper strip component generates a digital paper strip with a user defined length and width. The IsoAngle and Curvature parameters determine respectively the rotation of a ruling over the normal at the centre of the ruling and the local rotation of the strip around the previous ruling. The Segment parameter determines the number of rulings which the surface is based on.

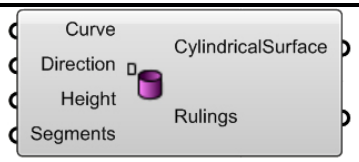
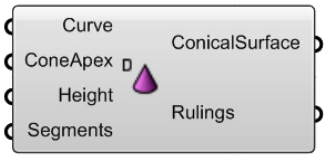
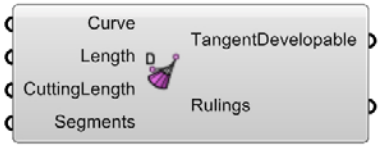
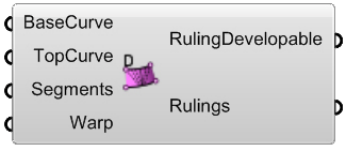
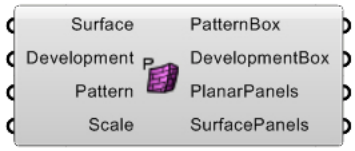
	• curve • vector • double • integer	• surface • list of lines	The cylindrical surface component takes a space curve and extrudes it parallel in the given vector direction with the specified height. The Segment parameter determines the number of rulings which the surface is based on.
	• curve • point • double • integer	• surface • list of lines	The conical surface component takes a space curve and performs a central extrusion in the direction of the ConeApex over the given height. The segment parameter determines the number of rulings which the surface is based on.
	• curve • integer • double • double	• surface • list of lines	The tangent developable component generates a developable surface based on rulings that are tangent to an input space curve. The Length parameter determines the length of the tangent vectors. This length minus the CuttingLength parameter defines the length of the rulings. The Segment parameter determines the number of rulings which the surface is based on.
	• curve • curve • integer • double	• surface • list of lines	The ruling developable component generates a developable surface based on two input curves and a ruling search algorithm. The algorithm steps through a number of possible ruling locations, defined by the Segment parameter and returns a ruling if it complies with the condition of developability within the given warp tolerance.

Table 3. Patching component input and output

component	*input*	*output*	*description*
	• surface • surface • list of curves • double	• curve • curve • list of curves • list of curves	The surface patching component maps a predefined pattern, denoting the panel definition, on a developable surface. The Development surface input can be generated from the isometric mapping component. The Scale input scales the pattern definition. The output shows the bounding box of the pattern definition and the developed surface. The panel definitions are generated both for the development and the original surface.

7.2.2 practical tolerances for developable surfaces

Many materials of construction have analogous constraints that at least qualitatively constrain full scale fabricated building elements to assume sheet like forms. The relationship between the material constrains of modelling materials in scale and the constraints of fabrication may only be approximate. However, for schematic design purposes, this approximate correspondence may be sufficient to guarantee the constructability of designed forms[140].

The class of developable surfaces is clearly quite constrained. While there is a broad class of shapes that may be constructed from a single-curved sheet positioned in space, there is certainly a much larger set of shapes, such as spheres, cars, Pixar's digitally animated characters, that cannot be tightly covered by a developable surface. However, since many sheet materials allow for an elastic deformation, it is possible to deviated from a mathematically correct developable surface without introducing visible distortions. In case of developable surfaces, this deviation is related to the warp angle between two surface normals at the end of each ruling line, Figure 7.2.

The maximum allowable warp angle for a physical sheet of a certain material is amongst others dependent on the distance between the normals, i.e. the length of the ruling, and the out of plane bending capacity. Since out of plane bending is not pure bending, calculating the capacity is much more difficult than for in plane bending. From a research in cold bent glass sheets, it followed that glass sheets display a double curvature for small deformations and a predominantly single curvature for large deformations[141]. Therefore, frequently, the maximum angle is based on a set of physical tests.

Suppose that the maximum warp factor for a certain material of a certain thickness is 0.1rad/m, then this value could be used as an input value for the components which are dependent on the warp factor to determine the feasibility of the physical surface definition. Ideally, properties such as thickness, stiffness, brittleness and forming methods are combined in a tolerance component, feeding the parametric developable components with values for allowable tolerances. Besides the physical restrictions also visual restrictions could then be implemented in order to quantify the effect on the desired smooth appearance, which is an important factor in surface design. Finally, an important factor for defining the feasibility of transforming a digital design to a physical design, is the curvature value perpendicular to the ruling direction, i.e. the maximum curvature value. Fully incorporating such a component is outside the scope of this research, but nonetheless would be an important addition to the set of tools in the toolbox.

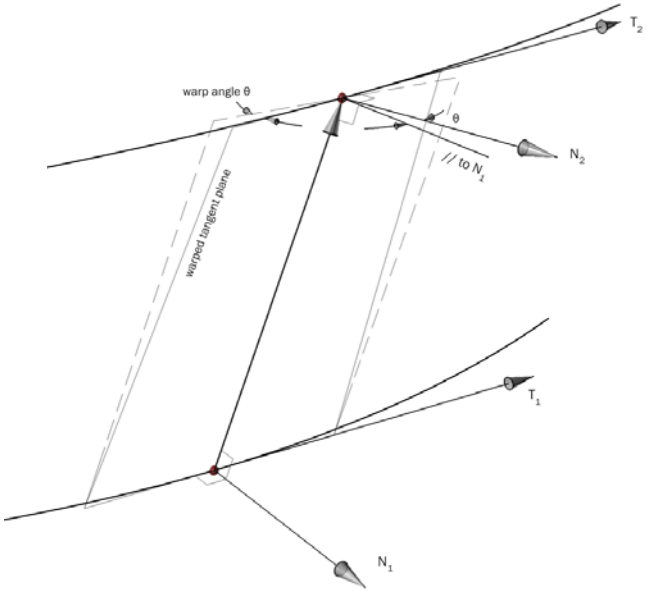


Fig. 7.2. The definition of the warp angle of a ruling

[140] Shelden, Digital surface representation and the constructability of Gehry's architecture

[141] Herwijnen, van, Cold bent glass sheets in façade structures

7.3 use case studies

Next to the test studies discussed in Section 6.2.2, a set of three case studies related to actual built projects are presented below in order to demonstrate the functionality of the rational design approach. As the case studies are based on a given geometry, it might be suggested that the tools are used in a post rationalisation approach; rationalising the geometry in relation to construction principles after initial design stages of the design process. However essentially, the example case studies follow the approach based on extracting parametric geometry from a design – although be it, not a sketch design in the given examples – and the generation of rationalised digital 3D models based on this input, instead of deriving geometric approximations of the initial design.

conical and cylindrical surfaces – modelling the Guggenheim Museum New York

The Salomon R. Guggenheim New York case study exemplifies how the façade surface of the museum can be parametrically generated by means of the ConicalSurface and CylindricalSurface components, Figure 6.3. These components allow for defining the geometry by changing the height of the rings as well as their diameter. Obviously, the geometric design could be based on a Grasshopper definition only taking standard components into account as well. Nonetheless, using the conical and cylindrical surface components allowed for a direct definition of the rings including the cylindrical elevator shaft.

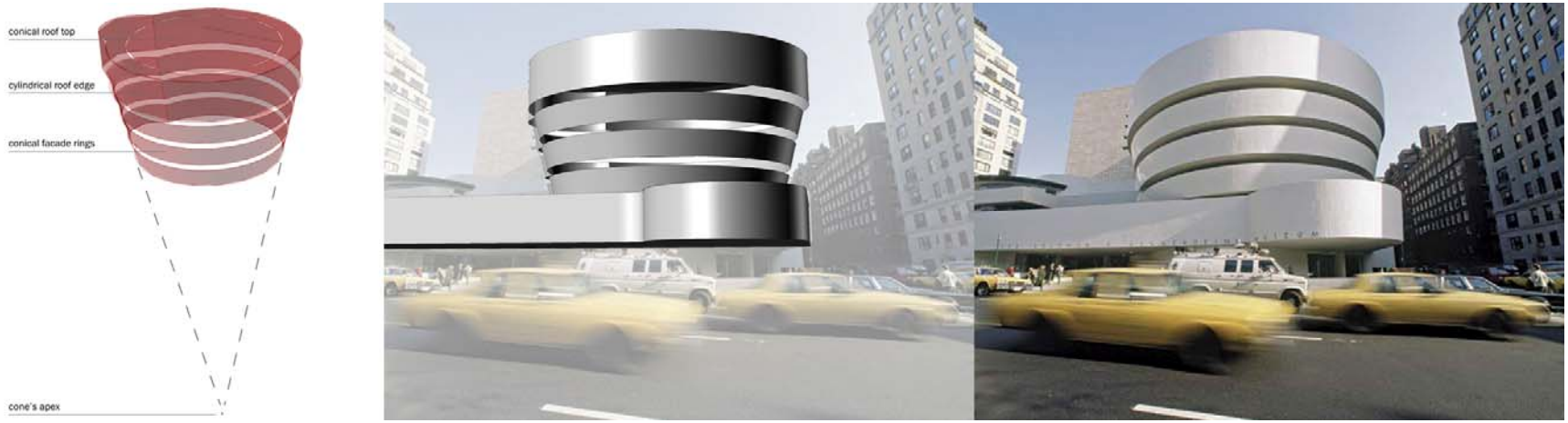


Fig. 7.3. Modelling the Guggenheim museum, New York by Frank Lloyd Wright with conical and cylindrical surfaces. Image left: http://new-york.world-guides.com, February 2011

ruling developable surface – modelling the roof edge of the ZVE

The roof edge of the Zentrum für Virtuelles Engineering (ZVE) Fraunhofer Institute (Centre for Virtual Engineering) is defined by two edge curves of which the bottom curve is in plane with the vertical façade. The top curve is placed inwards in relation to the façade which results in a nondevelopable surface when a simple loft operation between the curves is used to model the roof edge geometry.

The RulingDevelopable component is able to generate a developable surface between these curves, while allowing for parametrically remodelling of the top curve until the desired shape of the roof edge is defined, Figure 7.4. Locally, the mid section of the edge curves consists of two straight lines which are placed under an angle. As such, the only developable definition between the curves in this section is possible by modelling two triangles. Since in this example the bottom curve is defined as the input curve of the toolbox component, rulings are defined by subdivision points at equal parametric lengths along this curve. Therefore, one of the triangles is defined by two rulings, whereas the other is defined by a greater number of triangles. When the top and bottom curve are interchanged, the geometry of the triangles is unaltered, but the ruling definitions between the triangles is swapped. In either way, the component demonstrates that a single developable surface with smooth transitions between the triangles and from the triangles to the two other sections of the roof edge surface is possible.

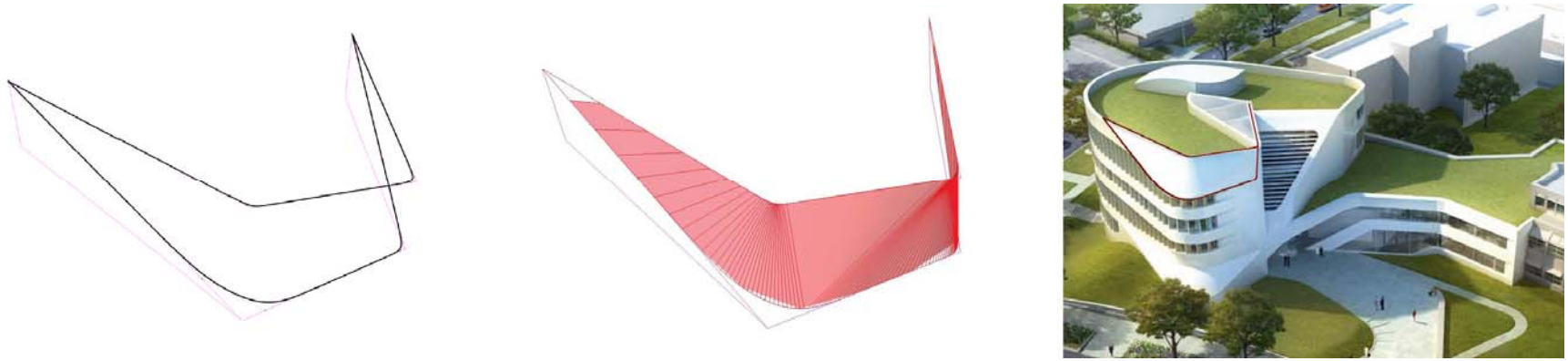


Fig. 7.4. Modelling the roof edge of the ZVE, Stuttgart by UNStudio with a ruling developable surface. Copyright left image: UNStudio

ruling developable surfaces – modelling the developable roof strips of the Kaohsiung Solar Stadium

The Kaohsiung Solar Stadium roof is covered by almost nine thousand solar panels, claiming the record for largest solar-powered stadium in the world. This case study shows another example of using the RulingDevelopable component in the geometric design of developable surfaces.

A Grasshopper definition takes a user defined horseshoe-shaped curve as the basic input. Although not fully following the final design geometry, this base curve is copied vertically and then copied inwards based on a single scaling factor. As such, the base shape of the stadium roof geometry is defined. Subsequently, the three curves are subdivided by a user defined number of points and connecting these points with a curve by shifting the second curve by *n* points and the third curve by 2x*n* curves generates the edges of the strips which serve as input curves for the RulingDevelopable component. As such, the developable strips locally follow a single curvature definition, whereas the global appearance is a double curved surface, Figure 7.5.

Fig. 7.5. Modelling the Kaohsiung Solar Stadium by Toyo Ito with ruling developable surfaces. Image left: http://meydan-city.com, March 2011

7.4 epilogue: tool adopting design approach

Concluding the tool development and implementation section of this thesis, a reference is made to the initial definition of discrepancies between handling of geometric complexity in design and construction. As indicated in the introduction chapter, over the last decades, the advances in digital modelling grew ahead of possibilities to follow the design intention in analysis and construction. Therefore, the relationship between development of computational tools and construction principles is frequently one that is relatively tensed, as computational power allows for extremely complex definitions of geometry, whereas construction principles generally ask for known and simple methods of manufacturing and construction. However, there seems to be a shift in design tool development, especially at engineering firms, where tools are developed indicating possible rationalisation methods related to the design or proposing simplified parametric structural models still relating to visually appealing shapes. One of the goals of this research was to present a number of simple tools which follows the approach related to this second type of tools, decreasing geometric complexity of a design, driving it to a well constructible shape, without desecrating the initial design intentions, Figure 7.6.

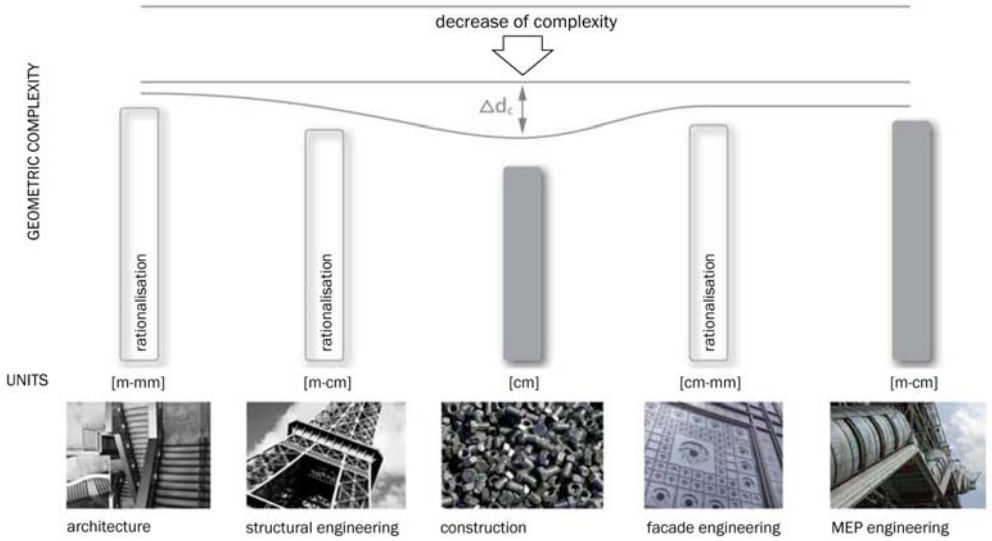


Fig. 7.6. Decreasing complexity, limiting the gap between design and construction by implementing digital design processes

part04 – discussion and conclusion

8 concluding remarks and reflection

This chapter presents the conclusions of the research described in this thesis. The results of the research are discussed and a reflection on the process is presented in the first two sections. Subsequently, the conclusions are drawn up categorised in different subsections. Finally, recommendations are presented, indicating additional research subjects related to this thesis.

8.1 discussion of the results

Rationalising surface geometry may be regarded as fixing problems which may only be overcome by negatively influencing the initial design intention, e.g. diverting from designed surface geometry, subdivision of smooth surfaces, limiting areas of double curvature. Nonetheless, frequently post rationalisation may prove to be the solution to provide for analysis and construction of a complexly shaped project. However, when pre-rationalisation methods are taken into account at a moment in the design process when the geometry is not yet fully defined, embedding of analysis and construction principles may drive the design to geometric definitions which directly follow design intentions.

As a rationalisation logic is embedded in the design process, there are obviously consequences to the descriptive language. However, this does not mean that design freedom is severely limited, especially when digital design parameters allow the designer to modify and alter the design within the logical boundary conditions of the rationalisation methods. As such, the digital rationalisation strongly relies on the derivation of (a limited number of) parameters and therefore it is of importance to identify the driving forces in deriving the parameters and how parameters may be altered when the initial parametric process fails to meet the design goals.

The tools presented in this thesis serve as the blocks of logic translating the input parameters to rationalised surface definitions. Therefore, as the rationalisation logic is linked to a parametric environment, it can be defined and be related to a certain design freedom within the boundary conditions denoted by the logic. The toolbox presented in this thesis serves as a proof of concept of embedding definitions and logic of rationalisation methods and demonstrates how surface geometry can be analysed and generated related to these rationalisation methods. The tools do not form a full range of commercially deployable methods, as their functionality is not generically applicable and validation of their outcome was limited, but they do demonstrate where and how the blocks of logic can be embedded in the design process.

8.2 reflection on the process

As mentioned in the preface, initially, objectives were set out to distil analysis and construction principles from complex geometry. However, these objectives reversely changed when the research focused on a less intangible subject of embedding logic to define rational geometry. Based on the general research in surface rationalisation methods, it became clear that embedding multiple methods in an in-depth research would negatively affect the quality of the research. Therefore a single surface class type was adopted in order to prove the concept of parametric digital modelling of rationalised surface geometry.

Of the discussed rational surface classes, developable surfaces contain a number of characteristics that make them well applicable for architectural and structural purposes, such as the possibility to generate smooth discrete surfaces that are aesthetically appealing and the advantages in relation to structural layouts and manufacturing. In particular, developable surfaces offer definitions of single-curvature geometries with an overall sculptural appearance in $\mathbb{R}^3$ space and are structured for constructability within specific design constraints. However, they remain difficult to model, particularly for non-expert users since using existing tools requires significant geometric expertise and time[142]. Therefore, modelling of developable surfaces in architecture is usually focused on two basic types: cylindrical surfaces and conical surfaces. Another method of modelling developables is demonstrated by the work of Frank Gehry, who uses paper models and a reverse engineering process to digitally define single curved surfaces. These methods are either restrictive in design freedom or complex and hard to control, especially from a structural perspective. Based on this, objectives were set out to define methods for analysing and modelling of developable surfaces based on various definitions of input and as such to present a design approach taking the defined logic of developable surfaces into account.

The research process of the graduation project proved to be difficult and the steps taken and the decisions made during this process may had shown a more fluent path under different circumstances. Also the progress and results in relation to the full development and employment of the toolbox could be better served, although, it is believed that the thesis together with the developed toolbox summarise the research in the parametric modelling of architectural developables into a coherent product.

[142] Rose, Developable surfaces from arbitrary sketched boundaries

8.3 conclusions and recommendations

conclusions on the parametric design approach for developable surfaces

The design approach as proposed in this research illustrates how a toolbox can provide for the analysis and modelling of developable surfaces based on parametrical input. In the parametric approach used in this research developables are analysed and generated as an envelope of one parameter sets of tangent planes. This allowed to adopt discrete differential geometry and vector mathematics as the underlying theoretical base. Related to this approach, a number of conclusions can be drawn on the parametric design approach for developable surfaces:

rationalisation and complexity in design

- The distinction whether a geometric description is complex or not is particularly important from an analysis and construction technology viewpoint in relation with the digital means of modelling this geometry. As such, complex geometry does not directly have to be related to visual geometry, but is more related to surface definition and the relation with construction principles.

- Therefore, along with the digital possibilities in fabrication and construction, focusing on decreasing the complexity of the design by rationalising surface definitions can narrow the gap between digital modelling and fabrication and construction.

- Shaping the geometry based on rational surface classes such as developable surfaces, allows designers to greatly influence the buildability along with affecting the level of complexity in structural analysis. In this sense, rationalisation may allow for full and precise control over the structural dimensions, may avoid having to deal with limitations in CAD/CAM machinery and related software and may force a design to be constructed out of elements from a limited number of moulds.

- In this research, it is suggested that a design approach incorporating pre-rationalised surface definitions, may be well implemented in a (digital) design process when custom developed tools related to these surface definitions are available to the designer. This especially, when these tools allow for the definition of the logic and restrictions defined by the surface definitions on one hand, and for a certain freedom in denoting design parameters on the other hand.

- In general, the problem of defining rational complex geometry is not that of the surface description itself, but the problem of answering to boundary conditions.

definition of the design approach and the digital toolbox

- The design approach offers an approach to the analysis and modelling of rational surfaces, providing the designer with a set of parametric and associative tools which can be used in a design process focussing on generating rational geometry. The proposed design approach is deployed via a digital toolbox containing various algorithmic methods categorised in three toolbox 'compartments' which are based on the theory of discrete differential geometry and vector mathematics.

- This approach aims to preserve a selected structure when going from a continuous abstraction to a finite representation for computational purposes.

- As such, the approach is a parametric modelling primitive which provides for interactive rational modelling methods based on the generation and deformation of developable surfaces for both architectural purposes as well as manufacturing and construction purposes, supporting the design process.

- Implementing a design approach for a parametric environment allows for maintaining an intrinsic structure and logic of the construction of developables surfaces where the input parameters are, within the aforementioned boundaries, providing design freedom.

- The digital toolbox has been developed containing three compartments related to 1) analysis, 2) modelling and 3) subdivision. As such, the toolbox offers the possibility to be utilised during various steps in the design process.

- The tools of the design approach should be utilised at stages in the design process where they are appropriate as the design approach tries to tie together the design intentions with the added value of a constructible definition of the design. This for instance means that a surface approximation method ideally is preceded by a surface analysis method.

- The tools allow for surface analysis and modelling and support the designer in the search for geometrically rationalised surfaces that answer to the initial ideas of the design. As such, the toolbox provides 'hints' to guide the development of surfaces towards the constructible geometry of developable surfaces, however the tools cannot be categorised as design tools. This in a sense that the tools provide the outcome of a predefined logic within a restricted workflow and therefore lack the essentials of a set of design tools, which allow for a more creative process.

- Although it is possible to model free form surfaces and to approximate these with developable surfaces, this may lead to undesired deviations from the design intentions. As such, the tools perform best in a pre-rationalised design process with surface definitions that are related to the structure of developable surfaces.

- The focus of the tools is primarily on the analysis and modelling of single surface patches and as such are generally less practical when implemented in a definition which is set up to generate a full design with.

- The components follow the logic of generating developables whilst taking into account various input parameters, based on numerical data or curve and surface geometries. For most of the components, the number of segments can be indicated, relating to the discrete differential approach and allowing the user to specify the accuracy of the generated surface in relation to theoretical defined developable surfaces.

- The approach can be seen as a parametric modelling primitive which provides for interactive rational modelling methods based on the generation and deformation of developable surfaces for both architectural purposes as well as manufacturing and construction purposes, supporting the design process.

theoretical and practical assets of designing with developable surfaces

- As with all the rational surface classes, developable surfaces embed specific geometric characteristics. The possibilities in architectural design to employ these characteristics are closely related to the restrictive boundary conditions to which developable surfaces have to comply.

- The toolbox contains tools which implement various algorithms based on the practical theory of discrete differential geometry and vector mathematics. By discretising the defining parameters, the resulting surface are numerical approximations of theoretical developable surfaces. As such, a certain simplicity is disclosed in the theoretical geometric complexity.

- As rulings are straight elements with a direction, the relation with vector mathematics is evident. Additionally, the expression for developable surfaces lead to complex equations when reduced to a function of one variable, but the definition of developables is relatively simple in terms of vectors. For these two reasons and in relation with the advantages mentioned above in using discrete differential geometry, rulings are defined based on vectors at discrete locations.

- Although the Gaussian curvature defines the theoretical relation of single curvature to double curvature, in practice merely denoting the Gaussian curvature of a surface does not fully extract the level of developability of a surface. This since a relatively low (but nonzero) curvature in one principal direction will flag the surface as nondevelopable, whereas the deviation may fall within material tolerance.

- Since elasticity and flexibility of material allows for certain deviation from theoretical developable surfaces, tolerances are ideally based on material properties which can be defined as an input parameter. When construction methods associated with fabrication are considered by having the input directly related to physical constraints of materials, the impact of the geometric results on the rationalised design process is more apparent.

- For most tools in the toolbox, numeric values of input parameters define the allowable tolerances or the number of surface defining rulings, allowing to manually defining the accuracy of the approximation, but also representing the relation with physical constraints.

- Developable surfaces possess the intrinsic characteristic that surfaces panels can be fabricated from flat material which is then bent in one direction in the factory or on site. The curvature in the direction perpendicular to the ruling direction is an important factor in determining the feasibility of the design. However, since the research omitted an in-depth research of bending properties of materials, single curvature constraints are not embedded in the tools.

development and programming aspects of the toolbox

- The tools present a logic which is defined by the type of surface class and which becomes available to the user via a set of methods and functionalities embedded in the tools.

- The code definitions of the components have been set up as individual classes per component and a set of general utility classes. Adding a functional component to the toolbox is therefore a matter of adding a new class.

- The general utility classes confine a number or Rhino SDK methods, such as the Loft method. Embedding these methods in the code lowered the complexity of some of the algorithms. However, when the algorithmic methods need to be transposed to other software applications or a standalone version, these methods have to be rewritten.

- In the component development, a standardisation of inputs and outputs is proposed in order to present and identify the toolbox as a coherent set of tools. The representation of the output of the surface modelling components is largely unified as two types of geometric definitions are provided; 1) the rulings which are conform the conditions of developability and 2) the lofted developable surface based on these rulings.

recommendations

The research focussed on the development of custom components for the parametric analysis and modelling of developable surfaces. Not all ideas related to this topic could be research during the research process. Below, a number of recommendations for further research are presented:

- Chapter 2 presented a set of surface rationalisation methods. For each of these methods a toolbox could be developed, expanding the possibilities of parametrically analysing and modelling to other types of surface rationalisation.

- In the proposed methods of based on discrete differential geometry, surfaces are approximated instances of theoretical surface definitions. Another method is to define rational surface based on a more direct relation with NURBS definitions.

- As mentioned before, adding an out of plane tolerance component together with a component which relates material properties with the single curvature value would be important additions to the current set of tools of the toolbox.

- On several projects of Gehry Partners, including the Guggenheim Museum and the Experience Music Project, the primary steel system only approximated the design surface. The actual dimensional control system for these projects was achieved through an additional system between the primary structure and the finish surface enclosure. On the Guggenheim project, the dimensional control system is developed through a series of template curved tubes that are attached to the primary system. On the EMP project the form is achieved by a shaped panel system attached to the primary structural steel rib system by adjustable connections[143]. Related to this, it is mentioned that it is of interest to be able to design developable surfaces used as a monocoque enclosure, but also to define a primary structural system which aligns with the developable enclosure, for instance by making use of the ruling direction.

[143] Shelden, Digital surface representation and the constructability of Gehry's architecture

- Optimisation routines defined by the user could possibly influence the resulting outcome of components to a large extend, such as for instance in the surface patching component or in approximating a double curved ruled surface to a developable surface.

- Partly just in the interest of adding a wow-factor to the research, but also of its possibilities in defining geometry, physically modelling designs with paper, scanning them with a 3D scanner and printing them to different scales with the help of a 3D laser cutter could be an eye-opener in modelling with developable surfaces*.

* http://www.makerbot.com/

references and appendices

references

books

Agoston, M.K., *Computer graphics and geometric modelling – Implementation and algorithms*. London: Springer-Verlag, 2005

Barnhill, R.E., *Geometry processing for design and manufacturing*. Philadelphia: SIAM, 1989

Blackwell, W., *Geometry in architecture*. Toronto: Wiley-Interscience, 1984

Bobenko, A.I. ed, et al, *Discrete differential geometry*. Basel: Birkhäuser, 2008

Davies, A., P Samuels, *An introduction to computational geometry for curves and surfaces*. Oxford: Clarendon Press, 1996

Do Carmo, M.P., *Differential geometry of curves and surfaces*. Englewood Cliffs: Prentice-Hall, Inc., 1976

Eisenhart, L.P., *A treatise on the differential geometry of curves and surfaces*. Boston: Ginn and Company, 1909

Farin, G., *Curves and surfaces for CAGD*. San Diego: Academic Press, Inc. 1993

Farin, G., D. Hansford, *The essentials of CAGD*. Natick: A K Peters, Ltd. 2000

Farin, G. ed, et al, *Handbook of computer aided geometric design*. Amsterdam: Elsevier Science, 2002

Goldman, R., *Pyramid algorithms – a dynamic programming approach to curves and surfaces for geometric modeling*, San Fransisco: Elsevier Science, 2003

Gray, A., *Modern differential geometry of curves and surfaces*, Boca Raton: CRC Press, 2000

Hardy, A., W.H. Steeb, *Mathematical tools in computer graphics with C# implementations*. Singapore: World Scientific, 2008

Kolarevic, B. ed., *Architecture in the digital age – design and manufacturing*. New York: Spon Press, 2003

Liberty, J., D. Xie, *Programming C# 3.0*. Beijing: O'Reilly, 2008

Lindsey, B., *Digital Gehry – Material resistance, digital construction*. Basel: Birkhäuser, 2001

Lynn, G., *Animate Form*. New York: Princeton Architectural Press, 1999

Iwamoto, L. *Digital fabrications – Architectural and material techniques*. New York: Princeton Architectural Press, 2009

Jodidio, P., *Architecture NOW!*. Köln: Taschen, 2007

Marsh, D. *Applied geometry for computer graphics and CAD*. London: Springer-Verlag, 1999

Mortenson, M.E., *Geometric modeling*. New York: Industrial Press Inc., 2006

National Information Standards Organization, *Understanding metadata*. Bethesda: NISO Press, 2004

Piegl, L., and W. Tiller, *The NURBS book*. Berlin: Springer-Verlag, 1995

Pottmann, H., et al., *Architectural geometry*. Exton: Bentley Institute Press, 2007

Pottmann, H., J. Wallner, *Computational line geometry*. Berlin: Springer-Verlag, 2001

Sakamoto, T. ed, et al, *From control to design – parametric/algorithmic architecture*. Barcelona: Actar-D, 2008

Salomon, D., *Computer graphics and geometric modelling*. New York: Springer-Verlag, 1999

Schlaich, J., and R. Bergermann, *Leicht Weit – Light Structures*. München: Prestel, 2005

Schodek, D., et al., *Digital design and manufacturing: CAD/CAM applications in architecture and design*. Hoboken (NJ): John Wiley & Sons, Inc., 2005

Stoker, J.J., *Differential geometry*. John Wiley & Sons, Inc., 1969

Szalapaj, P., *Contemporary architecture and the digital design process*. Oxford: Architectural Press, 2005

Taylor, W.F., *The geometry of computer graphics*. Belmont: Wadsworth inc., 1992

Watt, A., *The computer image*. Essex: Addison-Wesley Publishers Ltd., 1998

theses

Kocatürk, T., *Modelling collaborative knowledge in digital free-form design*. Delft : Delft University of Technology, 2006

Konesky, B.E., *Computer aided design of developable surfaces*. Vancouver: University of British Colombia, 1993

Shelden, D., *Digital surface representation and the constructability of Gehry's architecture*. Boston: MIT, 2002

papers

Arup, O., R. Jenkins, The evolution and design of the concourse at the Sydney Opera House, *Proceedings of the Institution of Civil Engineers*, April 1968.

Chu, C-H., C. Séquin, Developable Bézier patches: properties and design, *Computer aided design*, Volume 34, 2002, pp. 511-527

Flöry, S., H. Pottman, Ruled surfaces for rationalisation and design in architecture, *LIFE in:formation. On Responsive Information and Variations in Architecture*, 2010, pp. 103-109

Flyvbjerg, B, Design by deception: the politics of megaproject approval, *Harvard Design Magazine*, Volume 22, 2005

Glaeser, G., et. al., Developable surfaces in contemporary architecture, *Journal of Mathematics and the Arts*, 2007, pp59 — 71

Glymph, J., et al., A parametric strategy for free-form glass structures using quadrilateral planar facets, *Automation in Construction,* Volume 13, 2004, pp. 187-202

Herwijnen, F. van, et. al., Cold bent glass sheets in façade structures, *Structural Engineering International,* Volume 14, Number 2, May 2004

Huffman, D., Curvature and creasing: a primer on paper, *IEEE Transactions on Computers,* Volume C25 No 10, 1976, pp. 1010-1019

Nolan, T.J., Computer-aided design of developable hull surfaces. *The Society of naval architects and marine engineers*, May 1970

Pottmann, H., J. Wallner, Approximation algorithms for developable surfaces, *Computer Aided Geometric Design*, Volume 16, Issue 6, July 1999, pp. 539-556

Pottmann, H., et al., Geometry of multi-layer freeform structures for architecture, *Proceedings of ACM SIGGRAPH 2007*, Volume 26, Issue 3, Article No. 65, July 2007

Rose, K., et al., Developable surfaces from arbitrary sketched boundaries, *ACM International Conference Proceeding Series - Proceedings of the fifth Eurographics symposium on Geometry processing*, Volume 257, 2007, pp. 163-172

Singh, K., E. Fiume, Wires: A geometric deformation technique, *Computer Graphics 32, Annual Conference Series*, 1998, pp 405–414

Sun, M., E. Fiume, A technique for constructing developable surfaces, *Proceedings of Graphics Interface,* 1996, pp 176-185

internet

Cal Tech, *CS 286c/ACM 256 Discrete differential geometry: theory and applications*. February 2011
http://www.cs.caltech.edu

Design IntelligenceS, *Concepts as tools*. March 2010
http://designintelligences.wordpress.com

designtoproduction, *EPFL Learning Center, Lausanne 2008*. May 2010
http://www.designtoproduction.ch

The Gnomon Workshop, *Surface continuity tip*. August 2010
http://www.thegnomonworkshop.com/

Lenz, S. *Chesa Futura*. March 2010
http://www.daapspace.daap.uc.edu/

MIT, *Differential geometry of developable surfaces*. May 2010
http://web.mit.edu/

McNeel Wiki, *Rhino 4.0 labs tools*. February 2011
http://wiki.mcneel.com/labs/home

Penn Engineering, *Basics of affine geometry*. October 2010
http://www.cis.upenn.edu/

Penn Engineering, *CIS700 – emerging technologies*. September 2010
http://www.seas.upenn.edu/

Rhino3D, *Glossary*. December 2009
http://www.rhino3d.com

Saudi Aramco World, *The tiles of infinity*. May 2010
http://www.saudiaramcoworld.com

Sydney Opera House Trust, *Sydney Opera House, Utzon design principles*. January 2010
http://www.sydneyoperahouse.com

Wikipedia, *differential geometry of surfaces*. February 2011
http://en.wikipedia.org

Wikipedia, *Gauss map*. July 2010
http://en.wikipedia.org

Wikipedia, *Tangent developable*. January 2010
http://en.wikipedia.org

Wikipedia, *Tangential developable*. January 2010
http://en.wikipedia.org

Wolfram Mathworld, *Congruent*. May 2010
http://mathworld.wolfram.com

Wolfram Mathworld, *Curvature*. March 2010
http://mathworld.wolfram.com

Wolfram Mathworld, *Cylindrical projection*. March 2010
http://mathworld.wolfram.com

Wolfram Mathworld, *Envelope*. January 2011
http://mathworld.wolfram.com

Wolfram Mathworld, *Euler curvature formula*. February 2011
http://mathworld.wolfram.com

Wolfram Mathworld, *Klein bottle*. January 2011
http://mathworld.wolfram.com

Wolfram Mathworld, *NURBS surface*. March 2010
http://mathworld.wolfram.com

Wolfram Mathworld, *Tangent developable*. May 2010
http://mathworld.wolfram.com

magazines

Architectenweb Magazine, *Rolex Learning Centre - SANAA*. Volume 33, April 2010

appendix A. personal data

name:	R.J. (Roel) van de Straat
student number:	1041266
birth date:	March 3, 1982
home address:	Korvezeestraat 107 2628 DE Delft The Netherlands
telephone:	+31 (0)6 2810 7904
e-mail:	r.j.vandestraat@student.tudelft.nl rjvandestraat@gmail.com
work place:	Arup Amsterdam Naritaweg 118 1043 CA Amsterdam tel. +31 (0)20 752 3165 / +31 (0)20 305 8500 (front desk)

appendix B. graduation committee

name: **dr. ir. R.M.F. Stouffs** – first mentor (research)
university: Delft University of Technology
Faculty of Architecture
department of Building Technology
chair of Technical Design & Informatics
address: Julianalaan 134
2628 BL Delft
Room: 01WEST110
telephone: +31 (0)15 27 81295
e-mail: r.m.f.stouffs@tudelft.nl

name: **ir. F. Heinzelmann** – second mentor (design)
university: Delft University of Technology
Faculty of Architecture
department of Building Technology
chair of Architectural Engineering
address: Julianalaan 134
2628 BL Delft
e-mail: f.heinzelmann@tudelft.nl

name: **ir. J.L. Coenders** – third mentor (computation)
university: Delft University of Technology
Faculty of Civil Engineering and Geosciences
department of Structural and Building Engineering
chair of Building Engineering
BEMNextLab
address: Stevinweg 1
2628 CN Delft
Room: 6.66
telephone: +31 (0)15 27 85711
e-mail: j.l.coenders@tudelft.nl

company: Arup Amsterdam
address: Naritaweg 118
1043 CA Amsterdam
telephone: 31 (0)20 305 8500 (front desk)
e-mail: jeroen.coenders@arup.com

appendix C. toolbox installation manual

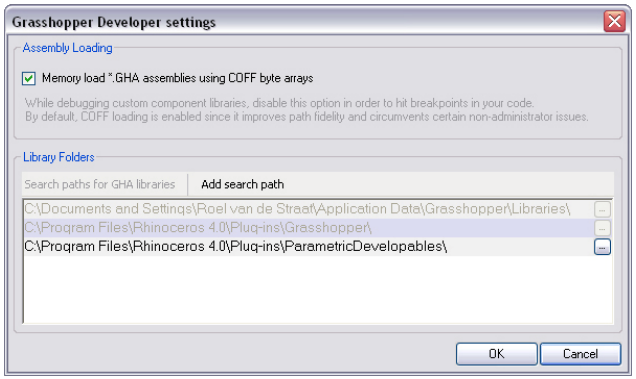
The tools have been developed as custom components for Grasshopper version 0.8.0004 in Rhinoceros 4.0 SR8 based on the RhinoCommon SDK*.
For a successful installation of the toolbox, Grasshopper 0.8.0004 or newer should be installed.

The following describes the installation procedure of the ParametricDevelopables toolbox.

1. save the ParametricDevelopables.gha assembly to a folder on your local hard drive

2. open Rhinoceros and start the Grasshopper plug-in by typing 'Grasshopper' in the command line

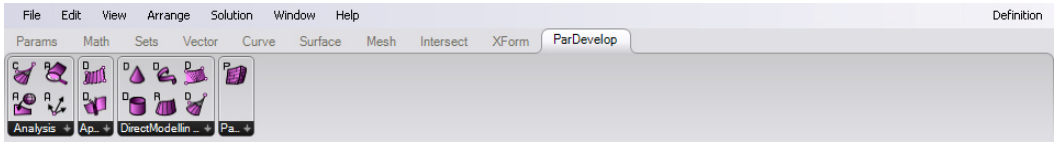
3. Type 'GrasshopperDeveloperSettings in the Rhinoceros command line. This will open the Grasshopper Developers setting menu, see image on the right.



4. Click on 'Add search path' and make a reference to the folder which contains the ParametricDevelopables.gha, see right.

5. Close the menu by clicking 'OK' and reopen Grasshopper.

6. The ParametricDevelopables plug-in should now be correctly installed and a new tab is added to the Grasshopper ribbon, see below.



* http://www.rhino3d.com/
http://www.grasshopper3d.com/
http://www.rhino3d.com/5/rhinocommon/