National Technical University of Athens

School of Civil Engineering

Department of Structural Engineering

Institute of Structural Analysis & Antiseismic Research

Isogeometric Static Analysis with T-SPLines

Supervisors:

Manolis Papadrakakis Professor NTUA

Panagiotis Karakitsios PhD Candidate NTUA

Dimitrios Tsapetis

Athens, May 2014

Acknowledgements

I would like to thank everyone who contributed in any way to the successful completion of my thesis. At first, I would like to express my heartfelt thanks to Professor **Manolis Papadrakakis**, not only for the opportunity he gave me to deal with Isogeometric Analysis and research in National Technical University of Athens, but also for the supervision of this thesis as well. I particularly thank PhD candidate **Panagiotis Karakitsios** for the excellent cooperation we have had over the last two years, which I really hope to be continued in the future. He was always available to address any questions concerning my academic and scientific issues, within and outside the framework of this study. I thank him sincerely for his ideas and guidance he gave me throughout the course of preparation of this work. Special thanks to researchers **Antonios Iakovos, Sofia Korlou, Athanasios Leontaris, George Karaiskos** and **Dimitrios Karras** for their contribution to the custody of the texts and the precious time spent and also for the cooperation so far as our team, which I am sure we will continue with the same pace in the future. I also thank **Katerina Dimitriou** for the cover page. I thank all my friends I met in university, and made these years a truly memorable experience. Of course, the biggest thanks I owe to my parents, whose faith in my capabilities made true all my dreams and targets.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω όλους όσους συνέβαλαν με οποιονδήποτε τρόπο στην επιτυχημένη εκπόνηση αυτής της διπλωματικής εργασίας. Ο πρώτος, στον οποίο θα ήθελα να εκφράσω τις θερμές ευχαριστίες μου, είναι ο Καθηγητής κ. **Μανόλης Παπαδρακάκης**, τόσο για την ευκαιρία που μου έδωσε να ασχοληθώ με την Ισογεωμετρική Ανάλυση και την έρευνα στο Εθνικό Μετσόβιο Πολυτεχνείο, όσο και για την επίβλεψη της συγκεκριμένης εργασίας. Θα ήθελα να ευχαριστήσω ιδιαίτερα τον Υποψήφιο Διδάκτορα κ. **Παναγιώτη Καρακίτσιο** για την εξαιρετική συνεργασία που είχαμε τα τελευταία δύο χρόνια και ελπίζω πραγματικά να συνεχίσουμε να έχουμε και στο μέλλον. Ήταν πάντα διαθέσιμος να ασχοληθεί με κάθε απορία μου σχετική με ακαδημαϊκά και επιστημονικά ζητήματα, εντός και εκτός των πλαισίων της παρούσας εργασίας. Τον ευχαριστώ θέρμα για τις ιδέες και την καθοδήγηση που μου προσέφερε καθ΄ όλη τη διάρκεια εκπόνησης αυτής της εργασίας και για όλες τις ερευνητικές συζητήσεις που πραγματοποιήθηκαν μέσα στα πλαίσια της ερευνητικής ομάδας GiGA Team. Θερμές ευχαριστίες στους ερευνητές **Αντώνιο Ιάκωβο, Σοφία Κορλού, Λεοντάρη Αθανάσιο, Γιώργο Καραΐσκο** και **Δημήτριο Κάρρα** για τη συνεισφορά τους στην επιμέλεια των κειμένων και για τον πολύτιμο χρόνο που αφιέρωσαν, καθώς και για την έως τώρα συνεργασία μας στα πλαίσια της GiGA Team, η οποία είμαι σίγουρος ότι θα συνεχιστεί με τους ίδιους ρυθμούς και στο μέλλον. Ευχαριστώ επίσης την **Κατερίνα Δημητρίου** για τη δημιουργία του εξωφύλλου. Ευχαριστώ τους φίλους και τις φίλες των φοιτητικών μου χρόνων, που έκαναν τα χρόνια αυτά μία πραγματικά αξέχαστη εμπειρία. Βέβαια, το μεγαλύτερο ευχαριστώ το οφείλω στους γονείς μου, των οποίων η πίστη στις δυνατότητες μου, αποτέλεσε αρωγό στην επίτευξη των στόχων και των ονείρων μου.

Abstract

The scope of this thesis is the investigation of static isogeometric analysis using a new type of shape functions, the T-SPLines. Isogeometric analysis is an innovative methodology of complete CAD – CAE integration introduced by J. Austin Cottrell, Thomas J. R. Hughes and Yuri Bazilevs. T-SPLine models have been analyzed assuming linear elastic behavior. T-SPLines and finite element analysis have been examined separately, as the two components of the isogeometric method. The code used to analyze T-SPLine models was programmed in MatLab, which allows easy matrix manipulation and strong mathematical background. Comparison with NURBS and geometrical representations were acquired through the first Greek IGA software, GeomIso. In order to acquire T-SPLine shapes, "Autodesk T-SPLine plug-in for Rhino" was utilized. The topics considered are T-SPLine formulation and properties, refinement techniques, stiffness matrix formulation, result post-processing (displacement, stress and strain field) and linear 2D applications investigating models of various representations.

Σύνοψη

Ο σκοπός της συγκεκριμένης διπλωματικής εργασίας είναι η διερεύνηση της ισογεωμετρικής στατικής ανάλυσης, αξιοποιώντας ένα νέο τύπο συναρτήσεων σχήματος, τις T-SPLines. Η ισογεωμετρική ανάλυση είναι μία καινοτόμος μεθοδολογία, η οποία προτάθηκε από τους J. Austin Cottrell, Thomas J. R. Hughes και Yuri Bazilevs και προσφέρει την πλήρη σύνδεση CAD - CAE. Στόχος είναι η γραμμική ελαστική ανάλυση φορέων με T-SPLines. Εξετάστηκαν τόσο οι T-SPLines και οι ιδιότητες τους όσο και η ανάλυση με χρήση πεπερασμένων στοιχείων, καθώς οι δύο αυτές ενότητες αποτελούν τις δύο βασικές συνιστώσες της ισογεωμετρικής μεθόδου. Ο κώδικας, ο οποίος χρησιμοποιήθηκε για την ανάλυση T-SPLine μοντέλων, έχει προγραμματιστεί σε MatLab, το οποίο διευκολύνει τις πράξεις πινάκων και διαθέτει ισχυρό μαθηματικό υπόβαθρο. Η σύγκριση με NURBS και οι γεωμετρικές αναπαραστάσεις πραγματοποιήθηκαν με το πρώτο ελληνικό λογισμικό ισογεωμετρικής ανάλυσης, το GeomIso. Ιδιαίτερη έμφαση δόθηκε στις T-SPLines και τις ιδιότητές τους, σε τεχνικές refinement, στη διατύπωση του μητρώου στιβαρότητας, στην επεξεργασία των αποτελεσμάτων (πεδίο μετατοπίσεων, ανηγμένων παραμορφώσεων και τάσεων) και σε εφαρμογές 2D τυπικών προβλημάτων.

Prologue

My involvement with isogeometric analysis began in June 2012, when I visited the office of PhD candidate Panagiotis Karakitsios to deliver the theme semester of the course Statics III and saw in his computer IGA shapes and asked what they are related to. He explained to me, and asked me if I was interested in the subject and after discussion with Professor Manolis Papadrakakis, he suggested the beginning of a collaboration that led to the formulation of my thesis.

Thus, began my collaboration with PhD candidate Panagiotis Karakitsios and join the research group GiGA Team, consisting of very talented researchers with remarkable intelligence and excellent teamwork. So, I was given the opportunity to become acquainted to a modern and innovative environment and start as a researcher in National Technical University of Athens. At a time, when our country losses most of its highly educated scientist, our team decided to give its own fight in research in one of the best institutions worldwide like NTUA and contribute to the international classification of the School of Civil Engineering at even higher levels of what it holds.

Dimitrios Tsapetis

Athens, May 2014

Table of contents

1. The Evolution of Isogeometric Analysis

1.1. Finite Element Analysis

The finite element analysis (FEA) is a numerical method (computational mechanics) for calculating approximate solutions to boundary value problems, which can be expressed by partial differential equations. Apart from the problems of solid mechanics, it applies to other fields like fluid mechanics, fluid dynamics, structure-fluid interaction, biomechanics, bioengineering, electromagnetism, heat transfer, acoustics and numerous other applications.

1.1.1. Historical Overview

FEA has its roots in ancient times, when mathematicians tried to approach the value of π using circumscribed polygons. It was until 1909 when the German mathematician Ritz conceived the initial principles of this method. In 1915, the Russian mathematician Galerkin developed further the theory of finite elements.

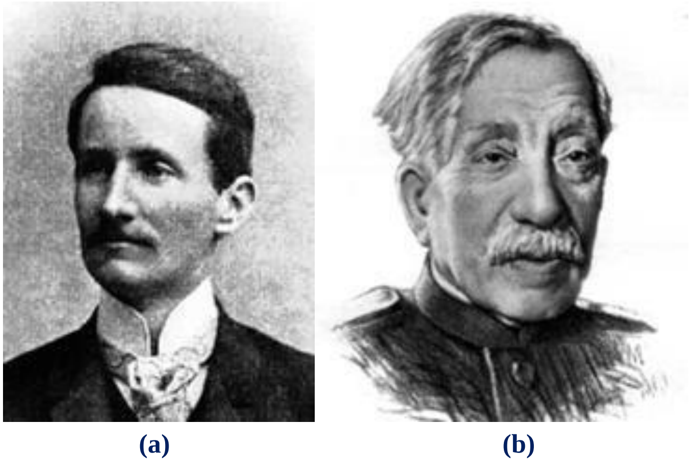


Figure 1.1.
(a) Ritz (b) Galerkin

FEA evolved along with the rapid advances of computers and soon became quite popular among computational mechanics community. The method was firstly used by the academic community and researchers, who supplied professional engineers with relevant software.

The idea of FEA was born in aeronautics. It was used to design inclined wings for jet fighter aircrafts able to fly in higher speeds. In 1941, Henrikoff introduced the so called framework method. In 1943, the German mathematician Courant solved the problem of torsion using triangular elements by taking advantage of the principle of minimum potential energy (Rayleigh-Ritz method). Courant's theory was soon forgotten as it could not be applied, but only until the time of computer invention.

Figure 1.2. Richard Courant (1888-1972).
German mathematician.

In 1944, John Argyris, who worked as a researcher at the Royal Aeronautical Society in Britain, grappled with this problem, as all known methods of analysis were unable to reliably simulate the inclined geometry wings. After painstaking efforts, he coined the use of triangular element and held its first implementation in electromechanical computers season with maximum algebraic systems capable of handling up to 64 unknowns. This was the time of the birth of the finite element method. In 1955, Argyris wrote a book entitled “Energy theorems” and the method of registers and evolved the method of FEA.

Figure 1.3. John Argyris (1913-2004).
Greek civil engineer.

BRIEF HISTORICAL OVERVIEW		
DATE	**SUBJECT**	**AUTHOR**
1779	Lagrange polynomials	
1864	Hermite polynomials	
1943	Linear triangle	Courant
1954-1955	Articles	Argyris and Kelsey
1956	Articles	Turner
1960	Finite element method	Argyris and Kelsey
1960	Term finite elements	Clough
1961	Bilinear quadrilateral	Taig
1962	Linear tetrahedron	Gallagher
1965-1968	C1-continuous triangles and quadrilaterals	Claugh, Tocher
1966	Isoparametric elements	Irons
1968	Isoparametric elements	Zienkiewicz and Cheung
1969	Constant pressure bilinear quadrilateral element	Hughes and Allik
1970	Variable-number-of-nodes elements	Zienkiewicz
1971	The eight-node serendipity quadrilateral	Zienkiewicz et al.
1972	Variable-number-of-nodes	Taylor
1974	Constant pressure bilinear quadrilateral element	Nagtegaal et al
1977	Constant pressure bilinear quadrilateral element	Hughes
1977	Effective one point integration Quadrilateral bending elements	Hughes et al
1978	Constant pressure bilinear quadrilateral element	Malkus and Hughes
1981	Quadrilateral bending elements	Hughes and Liu
1978	Scaled Lumbed rotatoty Inertia mass matrices	Hughes et al
1980	Constant pressure bilinear quadrilateral element	Hughes
1981	Effective one point integration Quadrilateral bending elements	Flanagan and Belytschko
1981	Effective one point integration Quadrilateral bending elements	Belytschko and Tsay
1982	Stabilized methods	Brooks and Hughes
1984	Effective one point integration Quadrilateral bending elements	Belytschko et al
1992-1996	Meshless methods	
2000	Selective integration	Hughes
2000	Lagrange and Hermite polynomials widely utilized	Hughes
2000	Nodes	Hughes
2000	Inertia mass matrices	Hughes
2004	Stabilized methods	Hughes

Table 1.1. Brief historical overview.
Finite element method.

In 1956, American scientists Turner, Clough, Martin and Top calculated the stiffness matrix of various elements, such as beam elements. In 1960, John H. Argyris and Kelsey published their work that was based on the principles of finite elements. In 1960, Clough (University of California Berkley) used first the name finite elements in his work and soon the term became the one that defined the method among scientists. In 1967, Zienkiewicz and Chung wrote the first book of finite elements. FEA's utilization to solve real engineering problems is tightly connected with the progress in computer technology. A large number of publications and books followed, which allowed engineers to become acquainted to the method, to handle more degrees of freedom and finally to apply it to numerous applications overdrawing their own scientific field.

1.1.2. Basic Idea

FEA introduces a system of algebraic equations, whose number coincides with the number of degrees of freedom. The unknown quantities are displacements at element nodal points. In order to solve complex engineering problems with a large number of degrees of freedom, the structure is divided into small parts that compose the analyzed geometry. FEA is able to provide reliable results, even though an approximate method, for a wide range of applications. However, as the geometry's complexity increases, the results prove to be less accurate and require quite more computational effort. This drawback lies on the inability to utilize the exact geometry mesh, as FEA uses an approximate one.

The application of FEA requires the following steps.

1. Geometry design in a CAD program.

2. Mesh generation. In the early stages of the method, this process was performed by hand, requiring a lot of time. Nowadays, this laborious task is replaced by computer algorithms called mesh generators.

3. Definition of analysis data. Pre-processor defines constitutive law, external loads, boundary conditions and other analysis parameters, such as shape function polynomial degree and number of Gauss points.

4. Calculation of the unknown nodal displacements by the solver as the numerical solution of the corresponding partial differential equations (Gauss elimination).

5. Depiction of the results by post processing algorithms, which provide displacement, strain and stress contour.

FEA approximates the solution with piecewise polynomial functions, called shape functions N, which calculate the displacement value $\{d\}$ at any internal point (x, y, z) of the element by interpolation of the nodal displacements $\{D\}$.

$$\underset{(3x1)}{\{d(x,y,z)\}} = \underset{(3x3n_e)}{\left[N(x,y,z)\right]} \cdot \underset{(3n_e x1)}{\{D\}}$$

where n_e is the number of finite elements.

The larger the number of elements (nodes, degrees of freedom, Gauss points) the more accurate the numerical solution is, especially when it refers to complex geometries.

The next step is to define strain and stress vectors, which are connected through the elasticity matrix (constitutive law).

$$\underset{(6x1)}{\{\sigma\}} = \begin{bmatrix}\sigma_X & \sigma_Y & \sigma_Z & \sigma_{XY} & \sigma_{YZ} & \sigma_{ZX}\end{bmatrix}^T \text{ (3D case)}$$

$$\underset{(6x1)}{\{\varepsilon\}} = \begin{bmatrix}\varepsilon_X & \varepsilon_Y & \varepsilon_Z & \gamma_{XY} & \gamma_{YZ} & \gamma_{ZX}\end{bmatrix}^T \text{ (3D case)}$$

$$\underset{(6x1)}{\{\sigma\}} = \underset{(6x6)}{[E]} \cdot \underset{(6x1)}{\{\varepsilon\}}$$

Deformation matrix [B] evaluates strains anywhere (x,y,z) in the model from nodal displacements.

$$\underset{(6x3n_e)}{\left[B(x,y,z)\right]} = \underset{(6x9)}{\left[B_1(x,y,z)\right]} \cdot \underset{(9x3n_e)}{\left[B_2(x,y,z)\right]}$$

$$\underset{(6x1)}{\{\varepsilon(x,y,z)\}} = \underset{(6x3n_e)}{\left[B(x,y,z)\right]} \cdot \underset{(3n_ex1)}{\{D\}}$$

where n$_e$ is the number of nodes per element.

The local stiffness matrix [k] of each model's element is given by the following integral.

$$\underset{(3n_ex3n_e)}{[k]} = \int_V \underset{(3n_ex6)}{\left[B(x,y,z)\right]^T} \cdot \underset{(6x6)}{[E]} \cdot \underset{(6x3n_e)}{\left[B(x,y,z)\right]} dV$$

Distributed loads can be transformed into equivalent nodal loads, according to the next equation.

$$\underset{(3n_ex1)}{\{r\}} = \int_V \underset{(3n_ex3)}{\left[N(x,y,z)\right]^T} \cdot \underset{(3x1)}{\{f(x,y,z)\}} dV$$

The local stiffness matrix [k] of each element is added to the total stiffness matrix [K] of the structure. The displacement vector is calculated with Gauss elimination by the system:

$$\underset{(3nx1)}{\{R\}} = \underset{(3nx3n)}{[K]} \cdot \underset{(3nx1)}{\{D\}}$$

where n is the model's total node number.

1.1.3. Method Development

A variety of complex problems was until recently considered unsolved, as it relied on the large computational effort needed. This fact made their solution practically impossible. The first application of finite element method was the triangulation of airplane wings, in order to determine how they are overworked by wind forces. It was the requirement of numerous calculations done by hand that made the process inapplicable. The progress of computer technology and computational methods (computational mechanics & geometry) enabled the programming of finite element method and thus made scientists able to solve such issues in short amount of time.

The main reasons why FEA is so widely spread and accepted by the scientific community are the following. FEA can be used in problems that require numerical approximation, as analytical solution is not attainable. In addition, the enforcement of complex geometries, physical loads and material properties to the simulated model is possible, without arising any significant difficulties. It also enables the connection of complex composite materials, while it can be easily refined and altered to approach the accuracy requirements of each problem. Finally, the introduction of dynamic analysis and non-linear material and geometry behavior enables the approximation of real materials.

The need to apply this innovative technique in industrial scale encouraged the development of finite element software. These programs needed to adopt a double nature by supporting both design and analysis process, which is a very difficult task. The combination of graphical representation and numerical analysis with finite elements would simplify the tasks of engineers, as it would call them to focus only on the essential aspects of each problem, neglecting the need of complex calculations.

Nowadays, commercial software partially incorporates both graphic design and finite element analysis. State of the art programs are now able to solve a wide range of problems from soil mechanics and structural analysis to thermodynamics and solid fluid interaction, covering practically most problems an engineer can encounter. The additional combination of non–linear behavior and dynamic analysis transforms FEA programs into the proper tool every engineer needs to have in his hands to face sufficiently even the most difficult problems.

Finally, the method of finite elements is not only used by engineers. All scientific fields have become acquainted to this method and use it constantly. For instance, doctors with aid of engineers analyze blood flow in vessels to determine weak points of the walls and diagnose and prevent heart diseases (thoracic and abdominal aortic aneurysm). At the same time, all mechanical tools and substitutes used in surgeries to replace human parts could be designed via FEA. In general, FEA tends to be the widest utilized tool for all fields of science.

1.1.4. Drawbacks

Even though the evolution of finite element method has been rapid since its birth, main drawbacks have not been overcome yet. For example, exact geometrical representation of the model with any mesh is almost impossible. Even isoparametric elements can only produce a more precise but still approximate mesh of the geometry. The most challenging tasks, that engineers face today, often require exact geometrical representation in order to achieve the desired accuracy. Except of mesh generation of CAD design, the initial geometry plays no more role in the analysis. This seems to be wrong, as the accurate geometry is replaced with an approximate one. As expected, it produces a vast number of issues. The inevitable geometrical approximation means there will be convergence errors by definition, regardless of the solution methods and the available computational power. This affects the efficiency of the solution.

In case a better approximation is required, refinement algorithms will return to the initial geometry and produce a different more precise one. The new, fine mesh cannot be directly produced from the previous. This means that procedures already completed have to be repeated in order to create the new mesh. Precious analysis time is required and the geometrical differences between the coarse and fine mesh make it difficult to compare the result. It is obvious that a new approach is needed that will integrate CAD programs and FEA software. Unfortunately, computational geometry and numerical analysis have been developed in different eras. Computer aided design has evolved greatly since its birth. On the other hand, finite element design geometries are unable to follow the rapid CAD evolution. These problems were always present throughout the history of FEM. Complicated geometries used in mechanics led to the development of algorithms and computational methods to minimize this effect. The problem is that the definition of finite elements does not facilitate the progress towards the full cooperation of design and analysis.

The solution to the drawback of geometric approach mentioned above is to embrace the development of IsoGeometric Analysis (IGA) using T-SPLines and subdivision surfaces, taking advantage of the method’s main idea, which suggests that any shaped geometry remains intact and the analysis is performed on the exact geometric model. Shape functions that are used in isogeometric analysis are in fact those used for design. This breakthrough of IGA allows the analysis of structures with complex shape, without making approximations. In addition, a designer is able to draft any object using less control points. When the object is about to undergo analysis, the engineer can effortless obtain the suitable mesh. Even when refinement procedure is needed, the engineer does not have to go back to the initial geometry. Refinement is performed always directly on the imported geometry, following the rules of design. Consequently, geometry does not change, leading to important time economy by procedures that will not be performed again.

1.2. Computer Aided Design

BRIEF HISTORICAL OVERVIEW		
DATE	SUBJECT	AUTHOR
1912	Bernstein polynomials	Bernstein
1946	Schoenberg coins the name "spline"	Schoenberg
1956	Subdivision seminal ideas	Rham
1959	de Casteljau algorithm	De Casteljau
1966-1972	Bezier curves and surfaces	Bezier-Faget
1971-1972	Cox de Boor recursion	Cox-de Boor
1972	B-SPLines	Riesenfeld
1974	Subdivision seminal ideas	Chaikin
1975	NURBS	Versprille
1978	Subdivision surfaces	Catmull-Clark and Doo-Sabin
1980	Olso knot insertion algorithm	Oslo
1987	Loop subdivision	Loop
1987, 1989	Polar forms, blossoms	Ramshaw
1996-present	Triangular and tetrahedral B-SPLines	Lai-Schumaker
1997	SPLine finite elements	Sabin
2003	T-SPLines	Sederberg
2008	Replace trimmed NURBS surface with untrimmed T-SPLines	Sederberg et al.

Table 1.2. Brief historical overview.
Computer Aided Design.

Computer Aided Design (CAD) is a drafting procedure using the aid of computer. It was invented and developed to help designers make more effective and accurate blueprints, requiring less effort. Drawing tools used to draw by hand were replaced by digital ones. Designers were able to manipulate, erase or correct the drawing without having to start it over again. The technology of polynomial B-SPLines and NURBS allows them to create any type of surface. These technologies are used for design of structures and common objects to Computer Aided Engineering (CAE). Advancements in mathematics and the parallel computer development have created new types of design tools. Subdivision surfaces is a 3D computer graphics tool, that creates smooth surfaces with the use of piecewise polynomial meshes. This technology is “rival” to T-SPLines (a generalization of NURBS). They allow rows of control points to terminate at any junction (no full tensor product nature), so their needed number is the less possible. Designers can create exactly the same geometry with NURBS using much less control points. Most of the previously mentioned technologies are used in Computer Aided Engineering to depict real objects. NURBS and T-SPLines are a mighty tool for isogeometric analysis, because their properties are suitable both for analysis and design.

1.2.1. Historical Overview

Despite the fact that SPLines in mathematical terms stand for smooth polynomial lines, their inspiration came from aircraft and ship hulls design. Before computer and computer aided design, draftsmen had to use flexible wooden or metal stripes in order to design the curves they needed. These stripes were fixed to the curvature of the curve using lead weights called ducks (the name ducks derives from their duck like shape).

(a) (b)

Figure 1.4.
(a) Sergei Natanovich Bernstein was a Russian mathematician, who contributed to partial differential equations.
(b) Pierre Étienne Bézier was a French engineer of Renault, who was the founder of physical modelling.

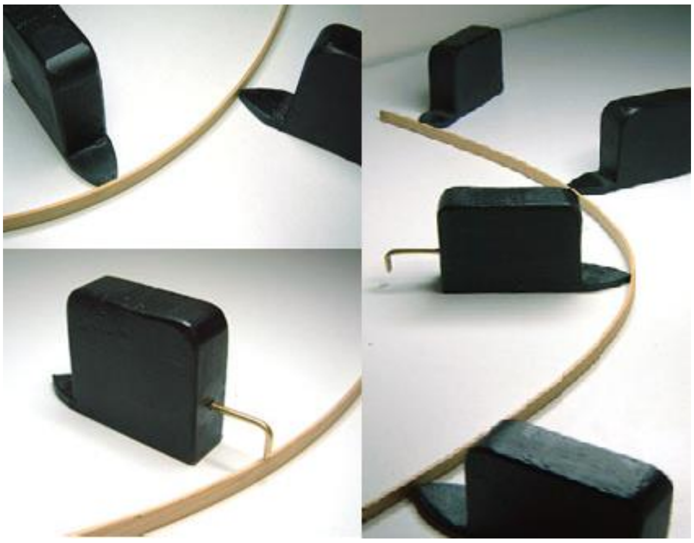
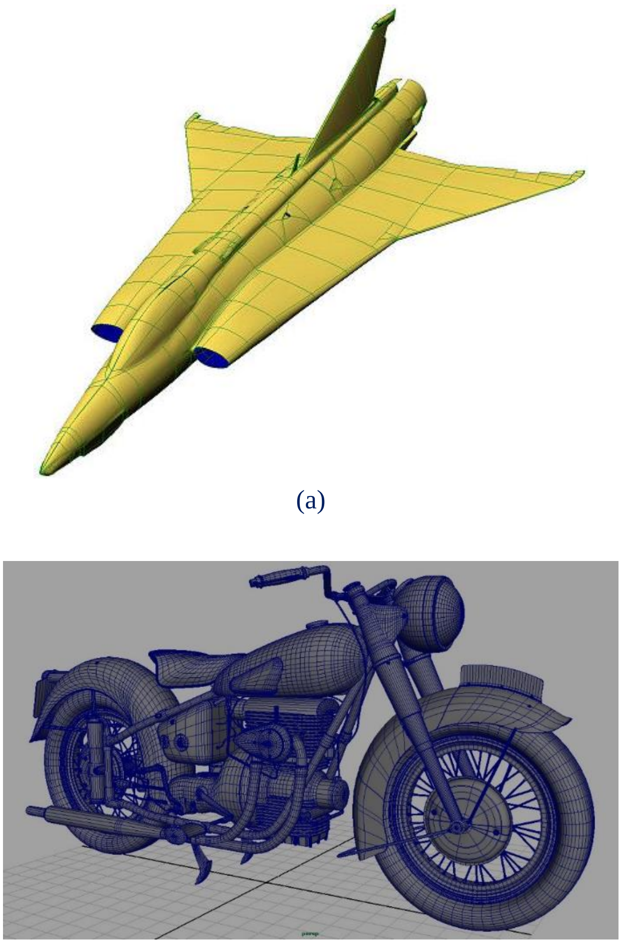


Figure 1.5. SPLines.
Metal or wooden elastic strip used for design.
(duckworksmagazine)

In 1946, mathematicians started studying the duck shape and eventually formulated the functions that form those polynomial curves. After the computer breakthrough, SPLines were widely embraced by the designing community, as CAD programs made it easy for draftsmen to create curves, overcoming the previous laborious “ducking” procedure. It is of major importance that these curves can approximate any polynomial degree p. Thus, they are C^{p-1} continuous functions, means they have p-1 smooth derivatives.

The introduction of SPLines into computer created the so called B-SPLines and their generalization NURBS. Their development was enforced by the need of precise representation of freeform surfaces. At first, they were only a tool of the automotive industry, but gradually were integrated into all commercial CAD programs. These new types of SPLines enable the designers to easily manipulate the geometry even more locally, only by moving control points. This was a milestone for CAD community.



(a)

(b)

Figure 1.6. NURBS objects.
(a) Airplane (Cadalyst).
(b) Motorcycle (seraphinacorazza.wordpress.com).

Finally, T-SPLines can be thought as NURBS for which a row or a column of control points can be terminated at any junction, not obligatory at the end of the corresponding parametric axis. This means that the mesh is now allowed to have T-, L-, I-, point-junctions, and not only cross ones.

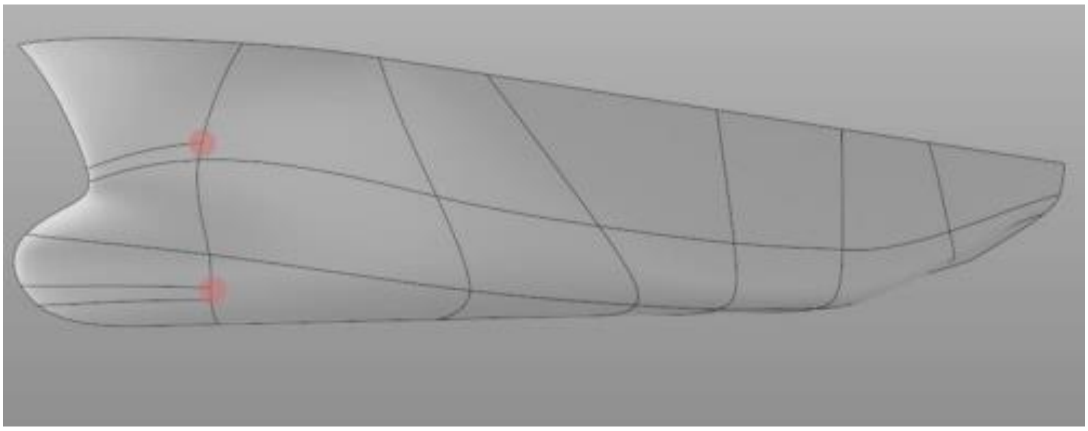


Figure 1.7. T-SPLines.
Ship hull designed with Rhinoceros.
(tsplines.com)

Modelling with T-SPLines drastically reduces the number of control points needed. This property is crucial to designers who want to define a surface at the lowest possible computational cost. In addition, all T-SPLine surfaces can be converted into B-SPLines by knot insertion and vice versa.

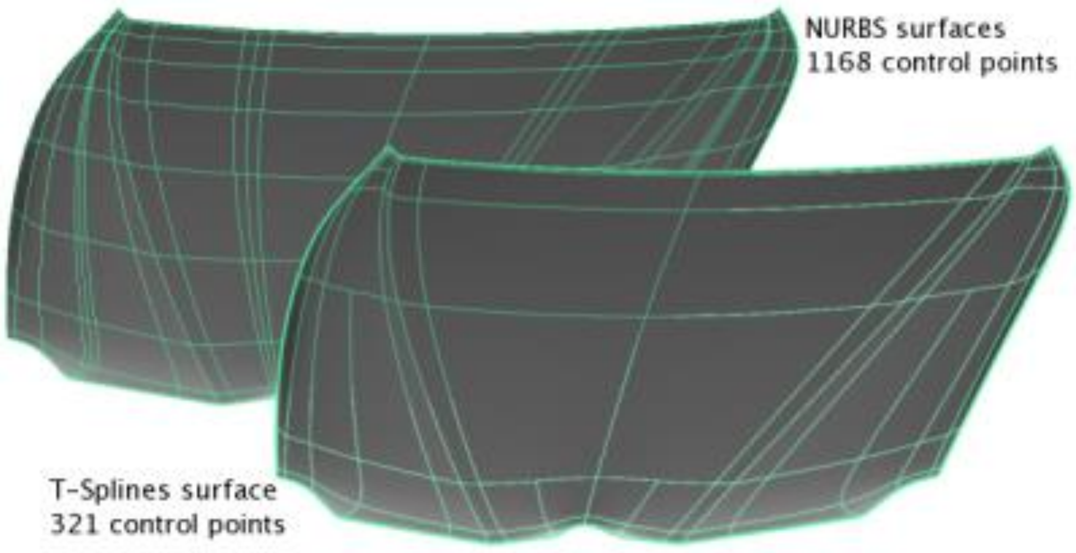


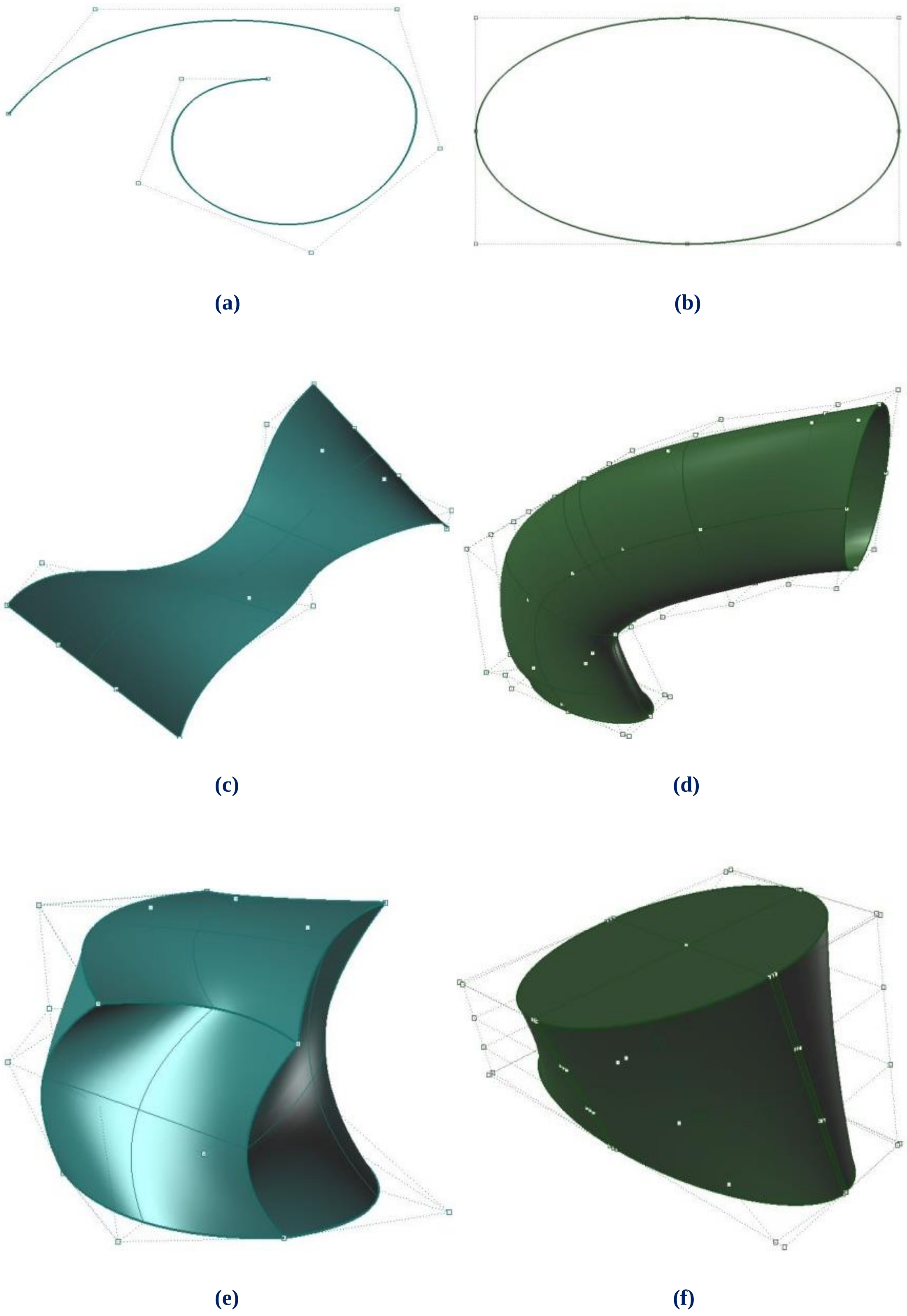
Figure 1.8. T-SPLines and NURBS.
Car hood designed with both technologies.
(tsplines.com)

1.3. Isogeometric Analysis

Isogeometric analysis (IGA) is a methodology first emerged at 2003, when Thomas J. R. Hughes (professor at the University of Austin Texas) expressed the need to integrate computer aided geometry design (CAGD) and finite element analysis (FEA), in order to overcome difficulties that arise in analysis. As a result, IGA was born.

Figure 1.9. Thomas J. R. Hughes.
Professor of Aerospace Engineering and Engineering Mechanics.
Initiator of Isogeometric Analysis (IGA).

The main idea behind IGA is to use the same mesh for the geometric design and the analysis process. Until then, it was inevitable for the finite element practitioner to take the design and approximate it with a new mesh, applying algorithms, the so-called mesh generators. In the case that meshing was inappropriate for analysis, the procedure had to be repeated, until reaching a certain convergence. In counter to how obvious and necessary was the transition to isogeometric analysis in order to reduce the computational time, the task was not so easy. Since CAD and CAE technologies appeared at discrete periods, they evolved individually. Different technologies apply to each of them, such as Lagrange polynomials for CAE and NURBS and T-SPLines for CAD. Isogeometric analysis is now present to bridge the gap between these two separate technologies, by using the design polynomials for analysis as well. This advancement will enable industries to easily design objects while using the same design for analysis purposes too. Many polynomial technologies have appeared, such as B-SPLines, NURBS, T-SPLines, polycube SPLines and subdivision surfaces. Their variety is without a doubt eligible to satisfy all types of problems that an engineer might encounter. Due to its important features, IGA has recently received great attention in the computational mechanics community and more and more CAE researchers are starting to study it.



(a) **(b)**

(c) **(d)**

(e) **(f)**

Figure 1.10. B-SPLine and NURBS objects.
(a) B-SPLine curve (1D). **(b)** NURBS curve (1D).
(c) B-SPLine surface (2D). **(d)** NURBS surface (2D).
(e) B-SPLine solid (3D). **(f)** NURBS solid (3D).
(Created by GiGA Team)

1.3.1. Basic Idea

IGA’s main idea is to use the same mathematical description for the geometry in the design (CAD) and the analysis (FEA). This implies to use SPLines, functions commonly used in computer-aided design to describe the geometry, as the basis functions for the analysis mesh.

While CAD community uses NURBS, T-SPLines and subdivision surfaces, the CAE one generally uses Lagrange polynomials to approximate the geometry. FEA’s approximate mesh necessitates a re-parameterization of the CAD geometry, which makes labor cost quite intensive and introduces automatically geometrical error. The basic idea of isogeometric analysis is to use the same shape functions and mesh both for design and analysis. In 2003, the research on isogeometric analysis started to focus on the question if finite element analysis could be done with Non-Uniform Rational B-SPLines (NURBS), the most widely used computational geometry tool in commercial CAD programs.

Recent research on isogeometric analysis uses Non-Uniform Rational B-Splines (NURBS) as shape functions, because they are extremely popular in engineering design systems. It has been shown that NURBS-based finite elements are very well suited for computational analysis, leading to more accurate results in comparison with standard finite elements based on Lagrange polynomials. It was the need for reduced computational time that initially motivated the isogeometric analysis concept in order to merge geometry design and mesh generation. In FEA software, practically around 80% of the overall analysis time is consumed in efforts to acquire a satisfactory approximate mesh, which has to be quite close to the accurate mesh of the geometry and suitable for analysis purposes.

NURBS were until lately the main shape functions used in isogeometric analysis. In 2008, T-SPLines were introduced as a worthy “opponent” that holds all the benefits of NURBS and at the same time permits local refinement. The scope of this thesis is to present the main properties of T-SPLines and make a comparison with NURBS. Water tightness, less control points, more design capabilities, but also sophisticated nature and complex implementation are its main features, which will be analyzed in the following chapters.

As a conclusion, isogeometric analysis is a methodology that has come to bridge the gap between computer aided design and finite element analysis. The utilization of the exact geometry mesh for analysis eliminates the approximation geometric error, while reduces significantly the analysis time. In addition, there is no need to repeat the geometry design for refinement purposes, which simplifies (more efficient and applicable) the whole refinement procedure. In contrast, remeshing with more and smaller elements is the standard technique in FEA, as it cannot utilize 100% the available data of the exact geometry mesh. All the above conclude that IGA has a bright future ahead.

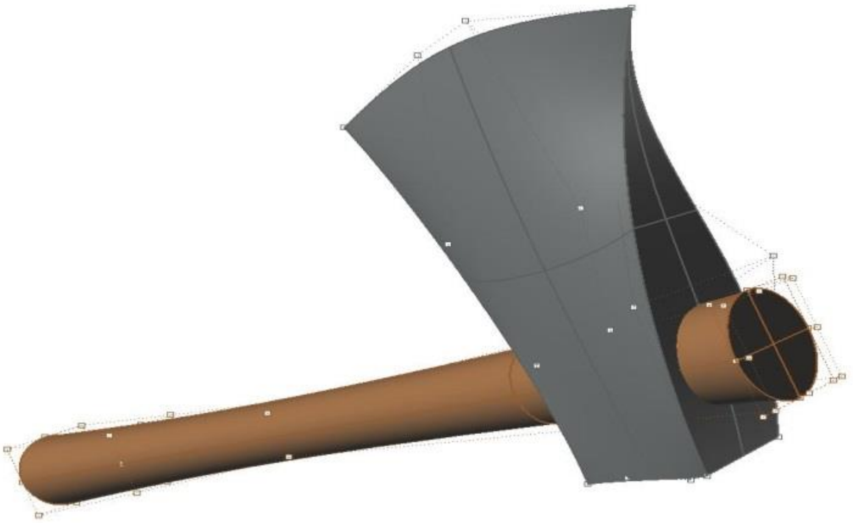


Figure 1.11. NURBS object consisting of two patches (one for wood, one for steel). Control points are shown as white circles. (Created by GiGA Team)

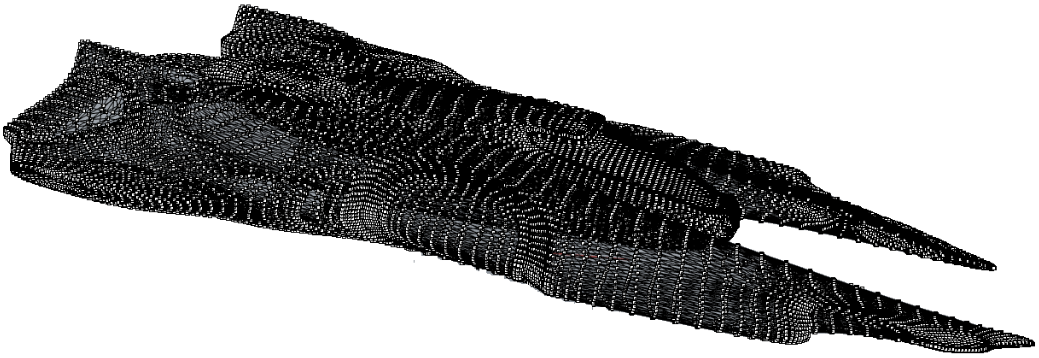


Figure 1.12. T-SPLine futuristic aircraft with complex geometry and large number of degrees of freedom.

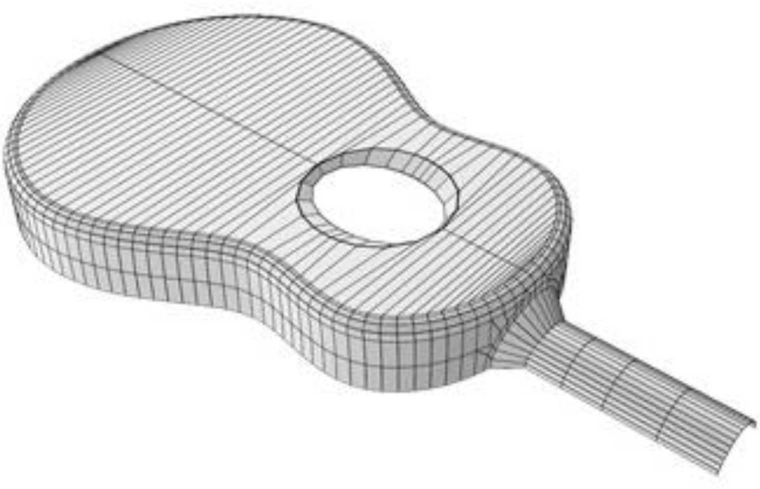


Figure 1.13. Guitar designed with subdivision surfaces.

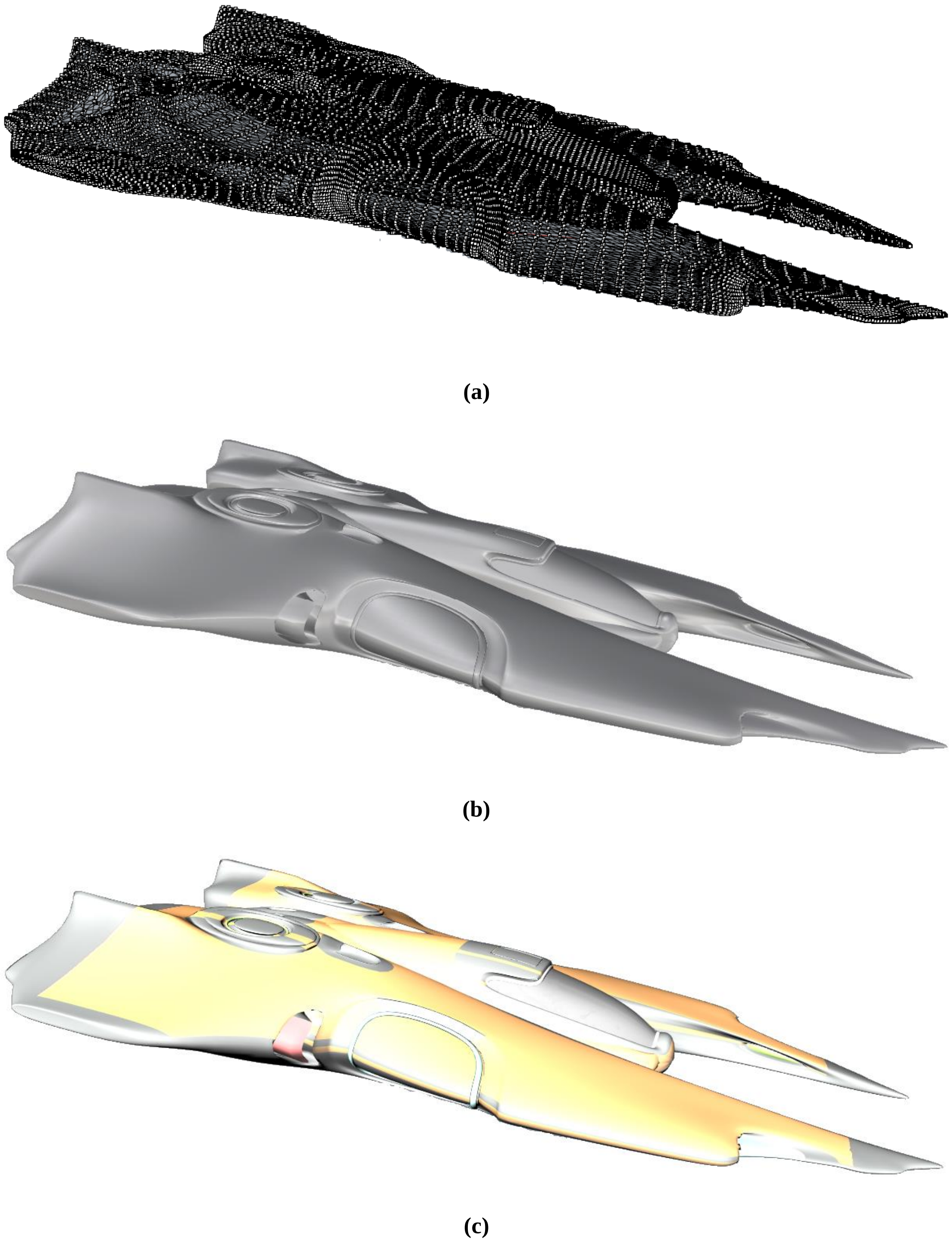
2. T-SPLines: Basic Ingredients

2.1. Introduction

It is since the dawn of time that engineers try to overcome everyday life problems by giving the best possible solution to them. Until the first half of 20th century, it was attainable for scientists to acquire an exact solution, something that unfortunately proved to be impossible nowadays. As computer technology rises, so do the demands of scientists to solve problems of incrementally increasing complexity. Even though it sounds contradictory, computer technology proves to be not only a powerful tool in engineers' hands, but also a major 'obstacle' for giving exact solutions, as its evolution increases the gap between the provided and the desired computing power. The development of computational mechanics made it easy to find an approximate and at the same time satisfactory solution for such complicated issues. The contribution of CAD industry is of great importance for this procedure, as common objects are approximated using CAD programs with great ease, enabling designers to make better and more complex drafts. The design gives the so called physical space, which can be of extreme geometrical complexity and thus hard to analyze. Although the isoparametric concept is being used in FEA, the exact geometry cannot be easily approximated due to the restricted element shape. This major drawback of FEA was the motivation to use another concept of analysis. This concept is expressed by IGA, that uses the shape functions of the geometry in order to approximate the numerical solution (displacement, strain and stress field) and is able to utilize all the known CAD tools (NURBS, T-SPLines, subdivision surfaces, polycube SPLines) for analysis purposes. Compared with NURBS technology, T-SPLines offer great design flexibility independent of geometry's complexity (no trimmed surfaces, watertight patch connection, NURBS patch merging), higher accuracy per degree of freedom (especially for complex geometries), significant reduction of the required control point number and local refinement. However, its formulation is quite sophisticated and unknown to the majority of engineers.

This chapter serves as an introduction to T-SPLines and places great emphasis on their formulation and properties. Important issues are:

- index space (essential), parameter space (auxiliary), physical (Cartesian) space
- junctions (T-Mesh, index space)
- global knot value vector
- anchors
- local knot value vectors
- T-SPLine blending functions (formulation, properties)
- T-SPLine shape functions (formulation, properties)
- T-SPLine geometry (properties, examples)
- refinement



(a)

(b)

(c)

Figure 2.1. Futuristic aircraft designed with T-SPLine technology.
This extreme complex geometry could be designed and analyzed efficiently with T-SPLines.
On the other hand, with NURBS it will be computationally demanding
due to the required vast number of control points and degrees of freedom
in order the same geometric accuracy.

(a) Physical space with control lattice.
Material points are shown in black, while control points in white.

(b) Shaded model. Common technique for designers.

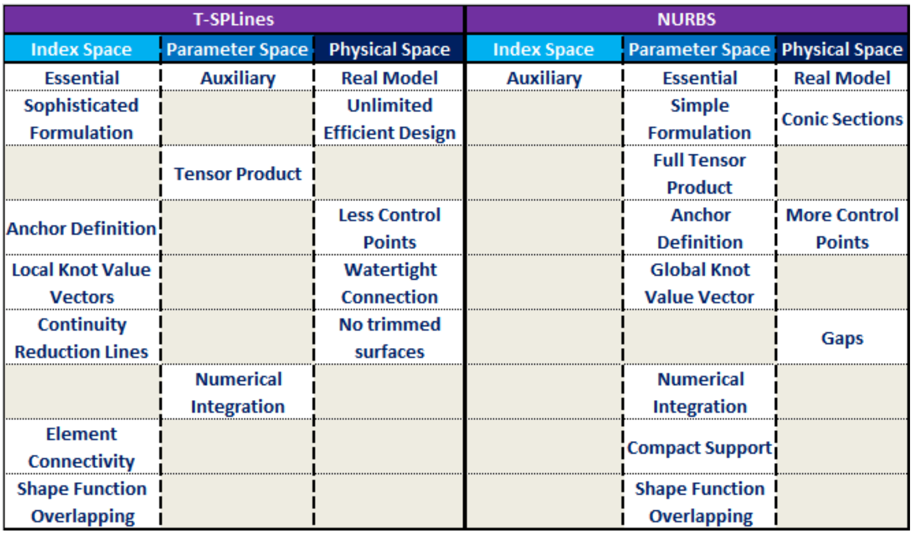
(c) Rendered model. Augmented reality.

2.2. Index, Parameter and Physical Space

In order to solve a boundary value problem with the isogeometric method, an engineer has to work with three spaces. These are:

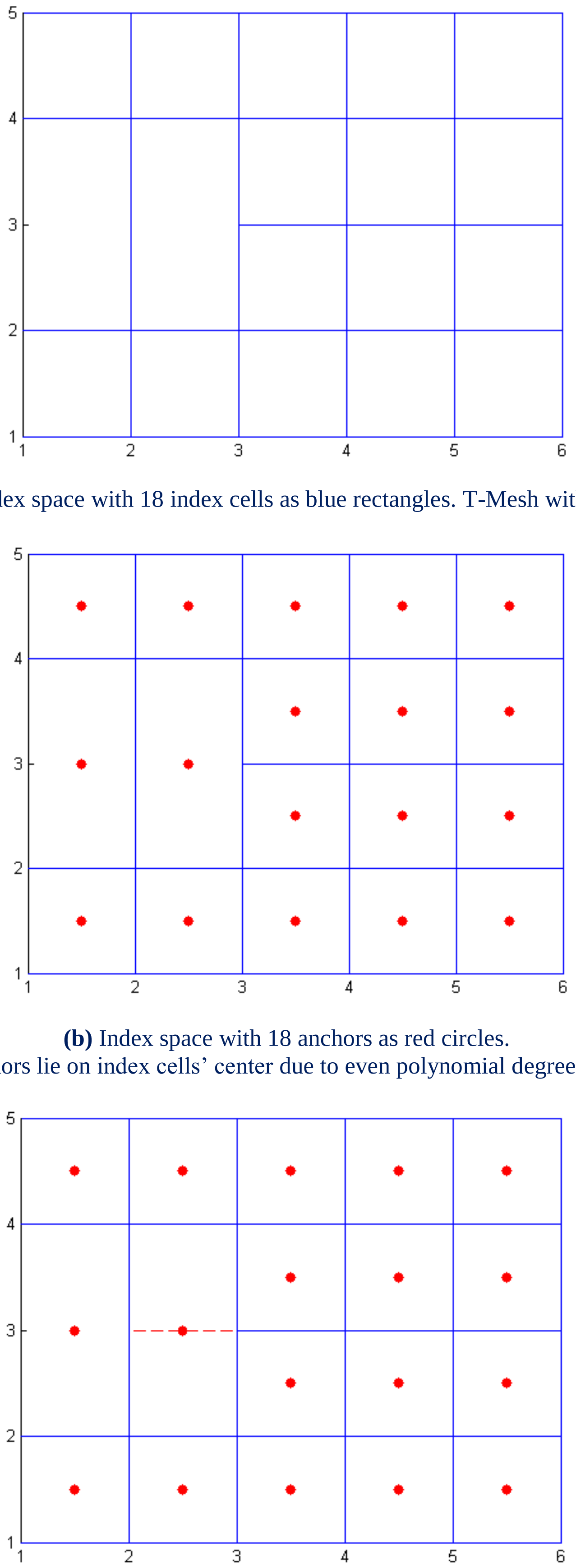
- Index space (essential for T-SPLines, auxiliary for NURBS)
- Parameter space (auxiliary for T-SPLines, essential for NURBS)
- Physical space

T-SPLines' formulation is tightly connected to index space, where junctions, anchors, local knot value vectors, continuity reduction lines and elements are defined. Knot values are positioned equally spaced on the index axes. Each knot value is depicted regardless to its magnitude. Index space plays a crucial role for this CAD technology in contrary to NURBS, where it is rather auxiliary. On the other hand, parameter space represents knots and not knot values, means that knot values with the same magnitude are placed in the same position on the parametric axes. Continuity reduction lines define elements, where Gauss points are located and numerical integration is taking place for each T-Mesh. Random curves, surfaces and solids are transformed into simple line segments, rectangular planes and cuboids. These normalized shapes allow flexible manipulation and analysis of the object, as their common pattern enables the application of efficient programming techniques. Last but not least is the physical Cartesian space, which represents the real geometry of the analyzed object. All known commercial CAD software aim to physical space and provide a variety of tools for its creation.



T-SPLines			NURBS		
Index Space	Parameter Space	Physical Space	Index Space	Parameter Space	Physical Space
Essential	Auxiliary	Real Model	Auxiliary	Essential	Real Model
Sophisticated Formulation		Unlimited Efficient Design		Simple Formulation	Conic Sections
	Tensor Product			Full Tensor Product	
Anchor Definition		Less Control Points		Anchor Definition	More Control Points
Local Knot Value Vectors		Watertight Connection		Global Knot Value Vector	
Continuity Reduction Lines		No trimmed surfaces			Gaps
	Numerical Integration			Numerical Integration	
Element Connectivity				Compact Support	
Shape Function Overlapping				Shape Function Overlapping	

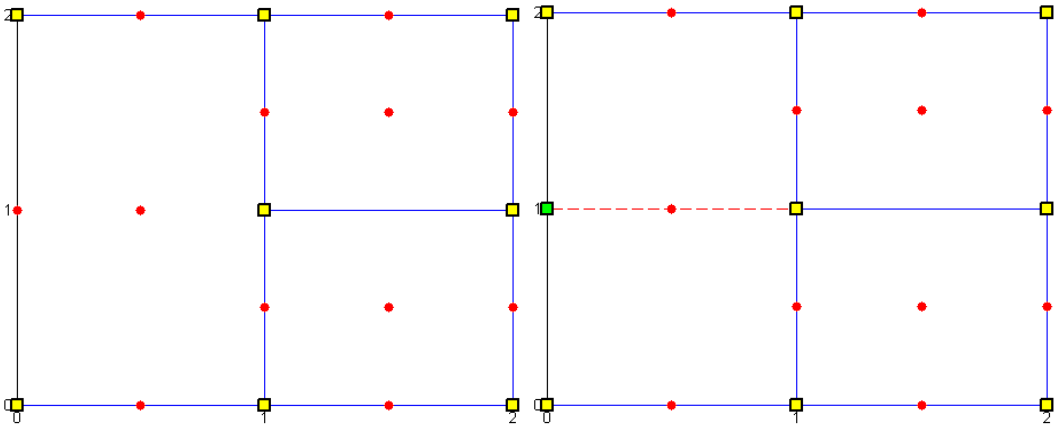
Table 2.1. T-SPLines compared with NURBS.
The essential space for T-SPLines is index space, while for NURBS is parameter space.
The appropriate data of T-Mesh (junctions, anchors, local knot value vectors
and blending functions) are defined in index space.
On the other hand, NURBS' less sophisticated formulation requires only the number of control points,
the BSPLine basis functions' polynomial degree and the global knot value vector (per axis).



(a) Index space with 18 index cells as blue rectangles. T-Mesh with junctions.

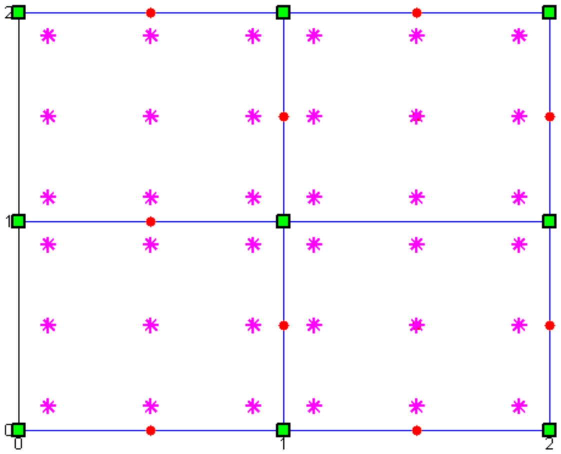
(b) Index space with 18 anchors as red circles.
Anchors lie on index cells' center due to even polynomial degree p, q.

(c) In index space, continuity reduction lines (red dashed)
divide the domain into 4 integration elements.

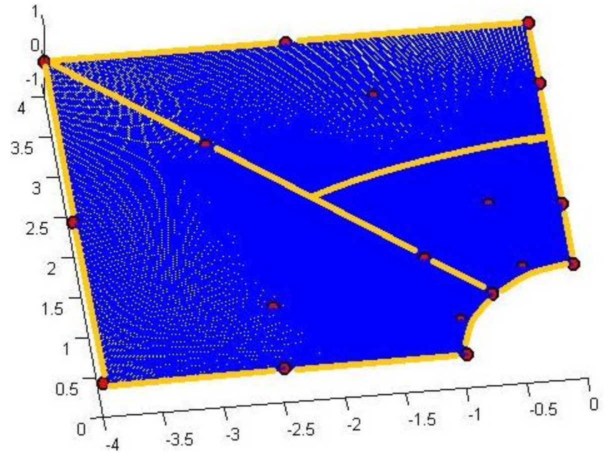


(d) **(e)**

Parameter space. Yellow squares are already existent knots.
Continuity reduction (red dashed) line adds one more knot, the green one.
The Gauss point number is equal to p+1=3 per knot span,
(the minimum number required for exact integration).

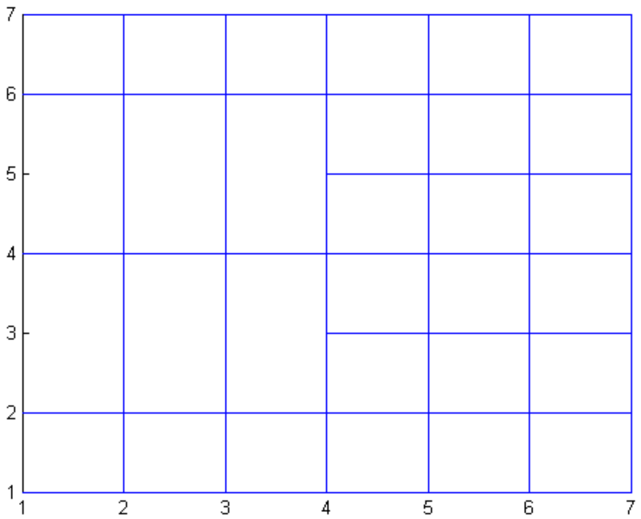


(f) Parameter space with Gauss points as magenta stars.

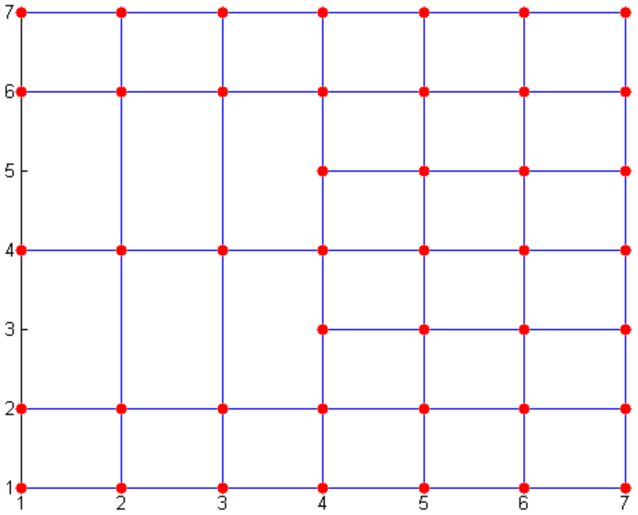


(e) Physical space. Plate with a hole.
Physical space is divided into 3 elements, which they do not coincide with the integration elements.
Control points are shown with red circles and
due to knot value repetition the edge ones are interpolatory to the curve.

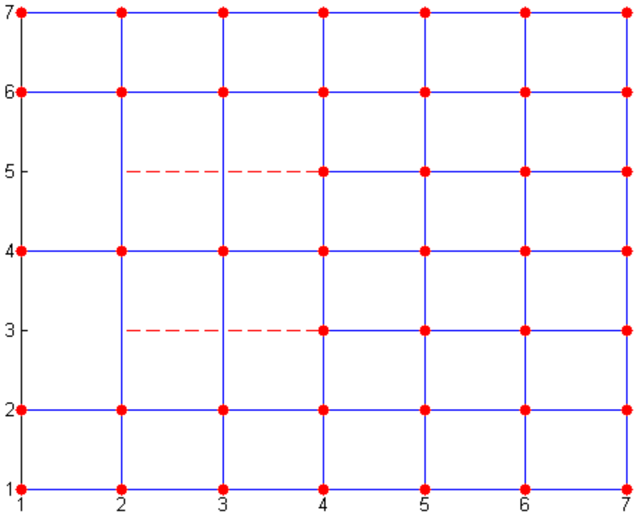
Figure 2.2. Index, parameter and physical space. (Created with GeomIso)
$\Xi=\{0,0,1,1,2,2\}$, p=2 and H={0,0,1,2,2}, q=2



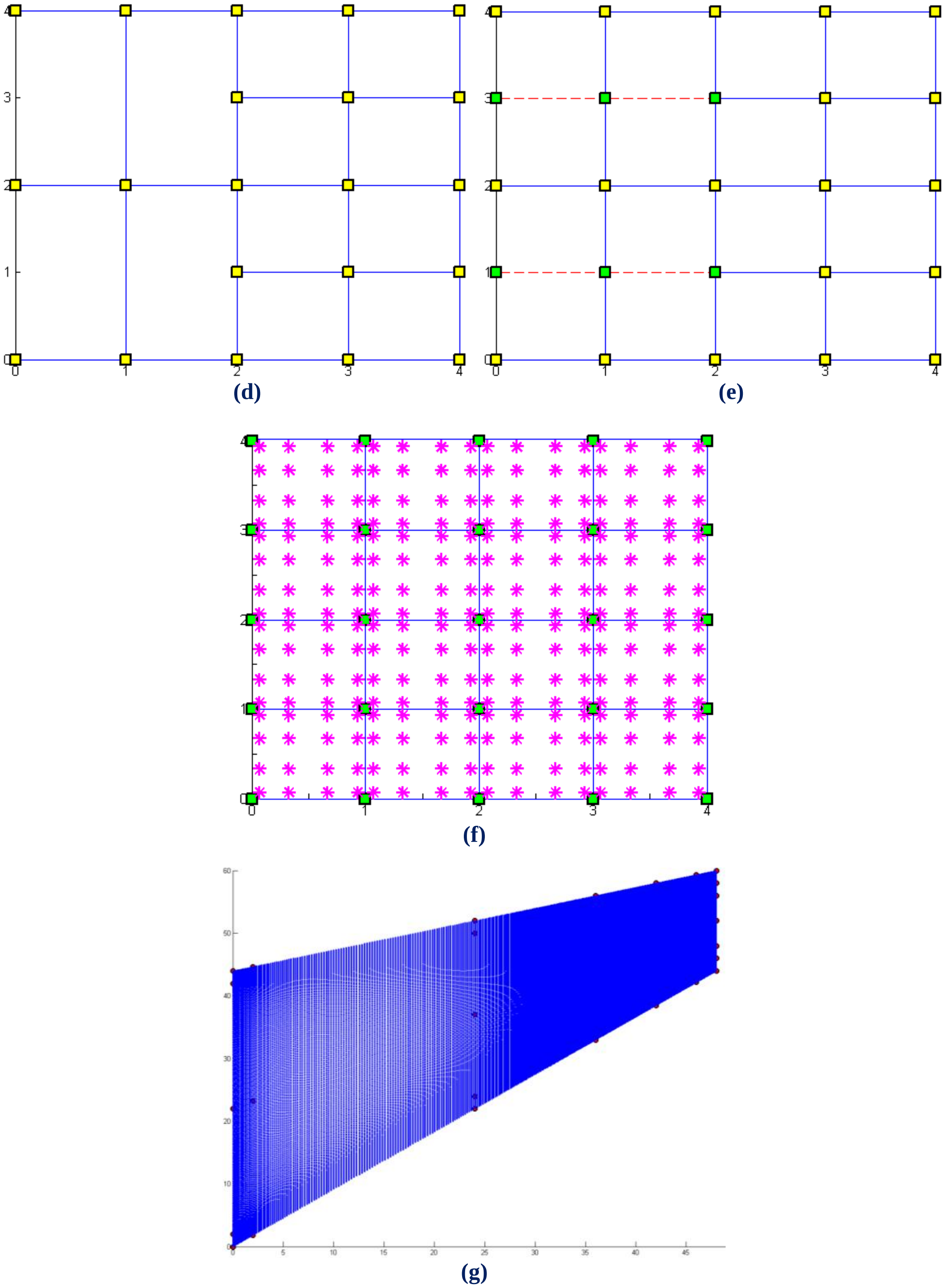
(a)



(b)



(c)

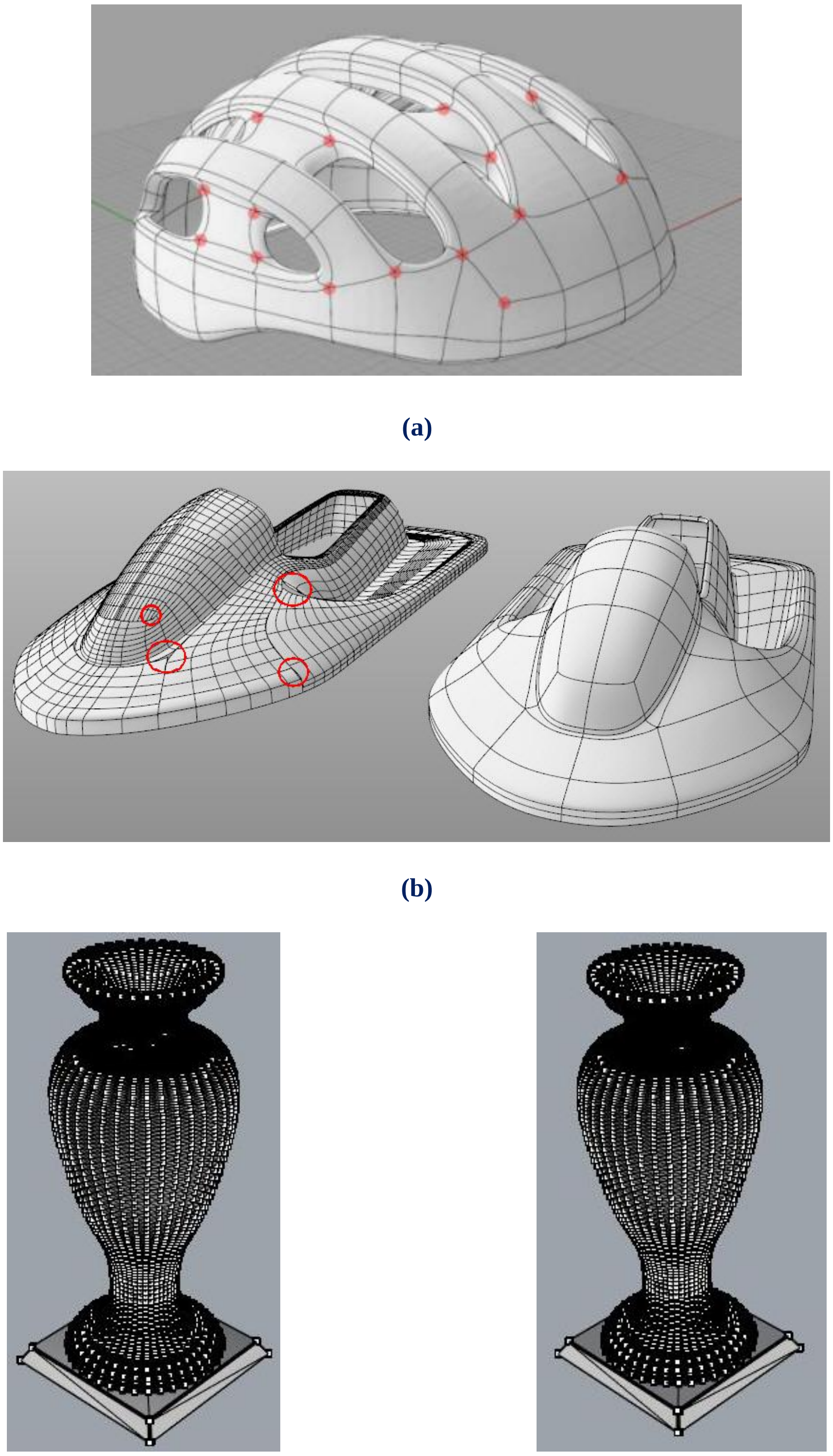


(d) (e)

(f)

(g)

Figure 2.3. Cook’s Cantilever. Index, parameter and physical space. (Created with GeomIso)
Ξ={0,0,1,2,3,4,4}, p=3 and H={0,0,1,2,3,4,4}, q=3
Subfigures **(b)** and **(c)** show index space before and after the creation of continuity reduction lines.
The edge control points are interpolatory to the curve, due to knot value repetition.
Anchors of index space are located on junctions (odd polynomial degree).
Subfigures **(d)**, **(e)** and **(f)** show parameter space that derives form index space with knot value vectors
Ξ=[0,0,1,2,3,4,4] and H=[0,0,1,2,3,4,4] and integration mesh created respectively.
Full tensor product structure is present in this example, but only for integration purposes.
Continuity reduction lines divide elements into smaller ones.



(a)

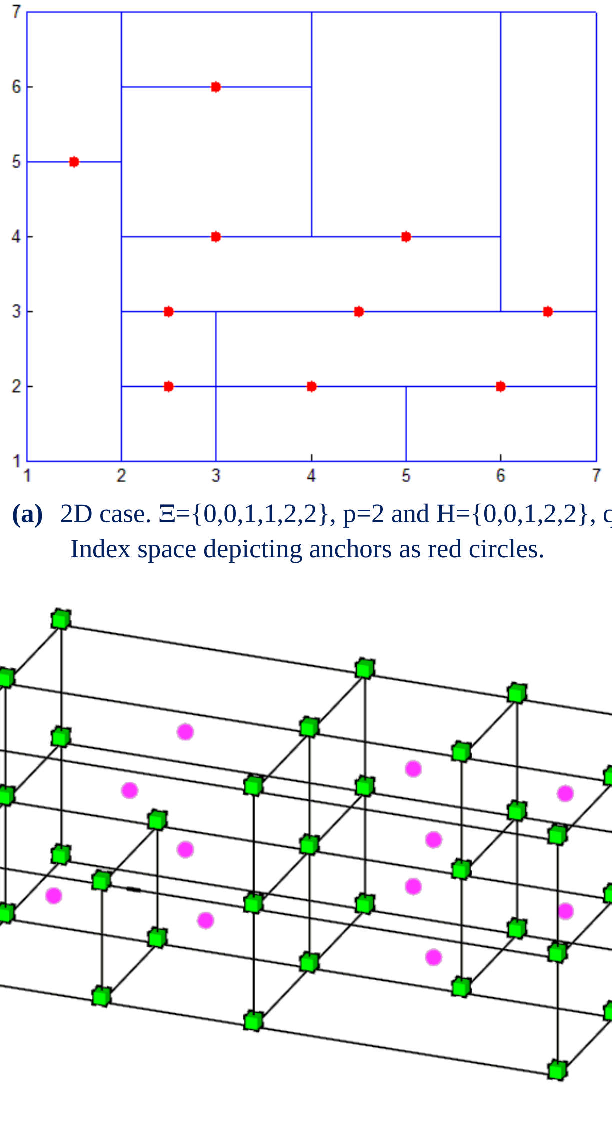
(b)

(c)

Figure 2.4. Physical Space. (tsplines.com)
(a) Helmet designed with T-SPLines in Rhinoceros. Red dots show the interconnection of NURBS patches, which were merged into a single T-Mesh.
(b) Jet ski designed with NURBS (left) and T-SPLines (right).NURBS require more control points. Marked areas show problematic patch interconnection (no watertightness, major drawback of NURBS). T-SPLines are able to design the same object with much less control points, all united in one watertight and compact T-Mesh.
(c) Vase with NURBS (left) and T-SPLines (right).

2.2.1. Index Space

Index space is created by equally spacing all indices, the number of which is equal to the number of knot values. Each index line is labeled with its sequence number (counting knot values). In order to define anchors and local knot value vectors, it is obligatory to identify for every blending function the start and end index lines. For T-SPLines, index space plays major role, as all basic variables initiate there, in contrary to NURBS, where it is used as an auxiliary tool. Anchors are defined first as the centers of their real support of p+2 knot value spans per direction. Then, the corresponding local knot values are produced in order to generate T-SPLine blending functions. T-SPLine blending functions are combined with the Cartesian coordinates of control points and generate T-SPLine shape functions.



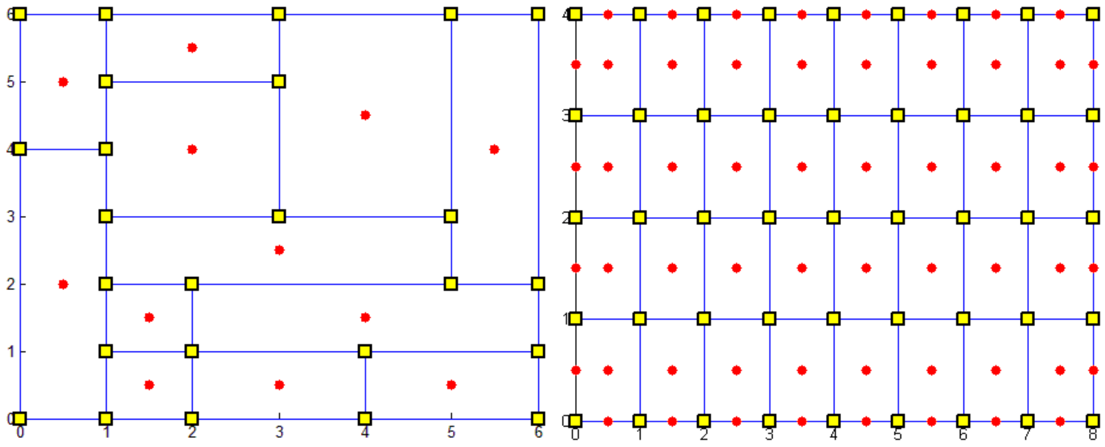
(a) 2D case. Ξ={0,0,1,1,2,2}, p=2 and H={0,0,1,2,2}, q=2. Index space depicting anchors as red circles.

(b) 3D case. Ξ={0,0,1,1,2,2}, p=2, H={0,0,1,2,2}, q=2 and Z={0,0,1,2,2}, r=2

Figure 2.5. Index Space.
(Created with GeomIso, Karras Dimitrios)

2.2.2. Parameter Space

Parameter space arises from the isoparametric concept of isogeometric method. It is the transformation of the frequently obscure shape of the object examined. Parametric axes ξ, η, ζ with the aid of a Jacobian transformation alternate the uneven element shape and turn them into pure and basic geometrical shapes. Specifically, randomized curves, surfaces and solids are turned into line segments, rectangular planes and cuboids respectively. This transformation enables easier analysis while ensuring the accuracy needed in the geometrical representation of real objects. In case of NURBS, parameter space is created by knot lines for all axes ξ, η and ζ, that are extended throughout the rest of the parametric domain, the so called full tensor product. Knot lines exist for each and every one of the knot values of the global knot value vector. This extension of knot lines is an identity that is no longer valid in T-SPLines. Note that parameter space due to the possible multiplicity of some knot values can be divided into two separate ones, i.e. parameter knot value space and parameter knot space. Parameter knot value space has the same number of indices per axis as index space. Indices now are the knot values spaced evenly despite the fact that more than one may have the same value. Parameter knot space has similar principles to parameter knot value space, but now knot values are no longer equally spaced and identical value intervals seem vanished and so they are called trivial spans.



(a) (b)

Figure 2.6. (Created with GeomIso)

(a) Parameter space 2D T-SPLines (no full tensor product).
Ξ={0,1,2,3,4,5,6}, p=2 and H={0,1,2,3,4,5,6}, q=2.
Knots are presented as yellow squares, anchors as red circles and elements as blue rectangles.

(b) Parameter space 2D NURBS (full tensor product)
Ξ={0,0,1,2,3,4,5,6,7,8,8}, p=2 and H={0,0,1,2,3,4,4},q=2.

2.2.3. Physical Space

Physical space is the well-known three dimensional Cartesian space in which structures are shaped. Each anchor and its local knot value vector, i.e. its support, is found in index space. Utilizing these vectors, blending functions are shaped and with the linear combination of control points' data, objects are formed in physical space. Note that control points are entities that describe the geometry of the object. They do not necessarily lie on the surface of objects. Their relative distance from the geometry depends on the mesh at their specific place and on their weights. An increasing weight draws the geometry closer to the control points.

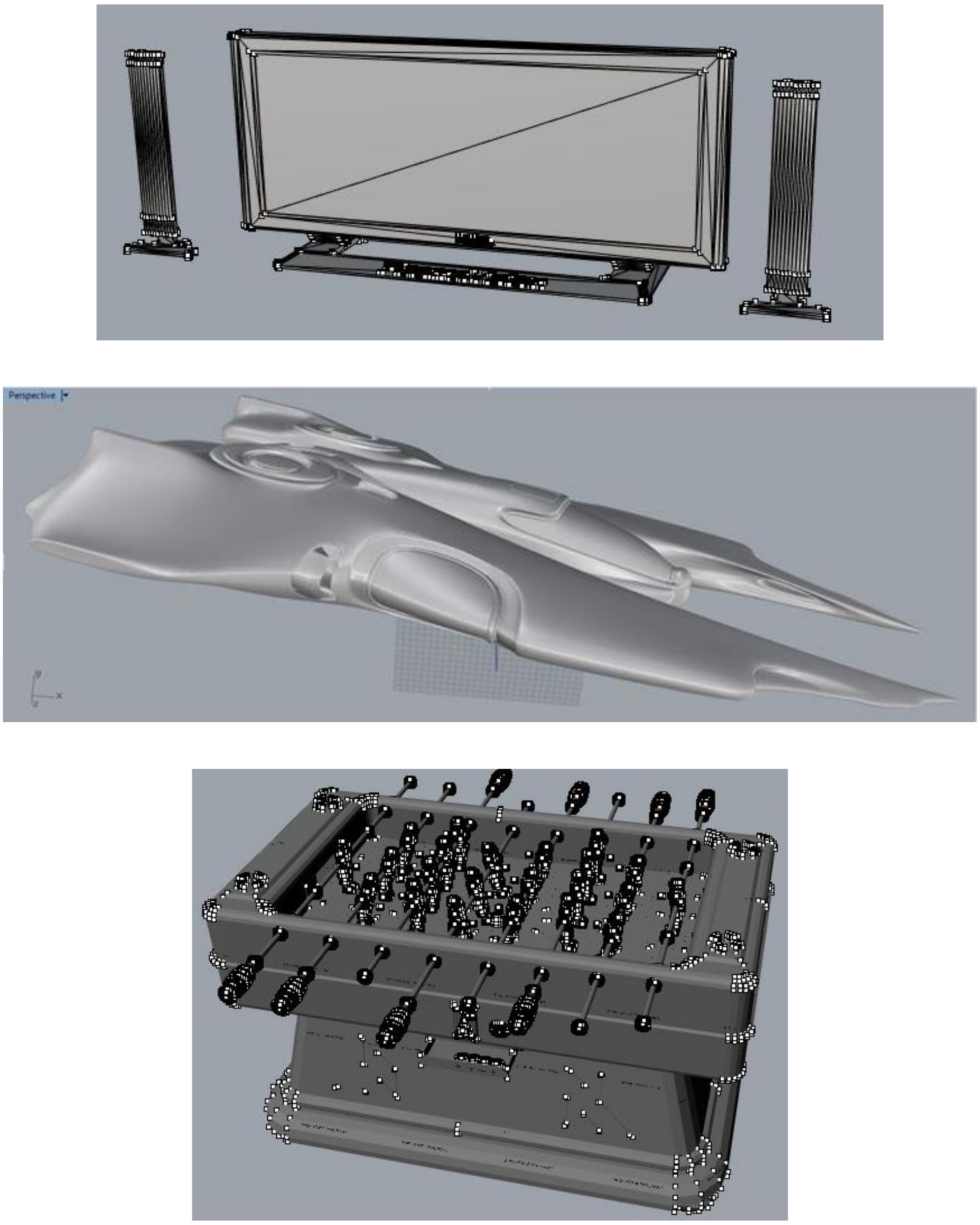


Figure 2.7.
T-SPLine objects. Physical space.

2.3. T-SPLine Blending Function

Given the fact that T-SPLines are a generalization of NURBS, the full tensor product property does not always apply here. T-Mesh is produced in index space and the rectangular tiling does not have a constant pattern like in NURBS. The major constraint, when creating a T-Mesh, is that every edge should lie on a positive integer index value. Thus, we can conclude that many types of edge to edge interconnections can exist. When four edges are connected, then we can see a normal interconnection, just like in NURBS. This is called a cross junction. Likewise, when three different edges connect, then we have a T-junction, two edges can form an L-junction or I-junction and finally, when in 3 consecutive index values don't connect at all with each other, then we have the so called node junction. Typically the last type does not exist and thus cannot be seen in T-Meshes, but we emphasize in its existence in order to show that full tensor product property no longer is mandatory. All the mentioned junctions are given below.

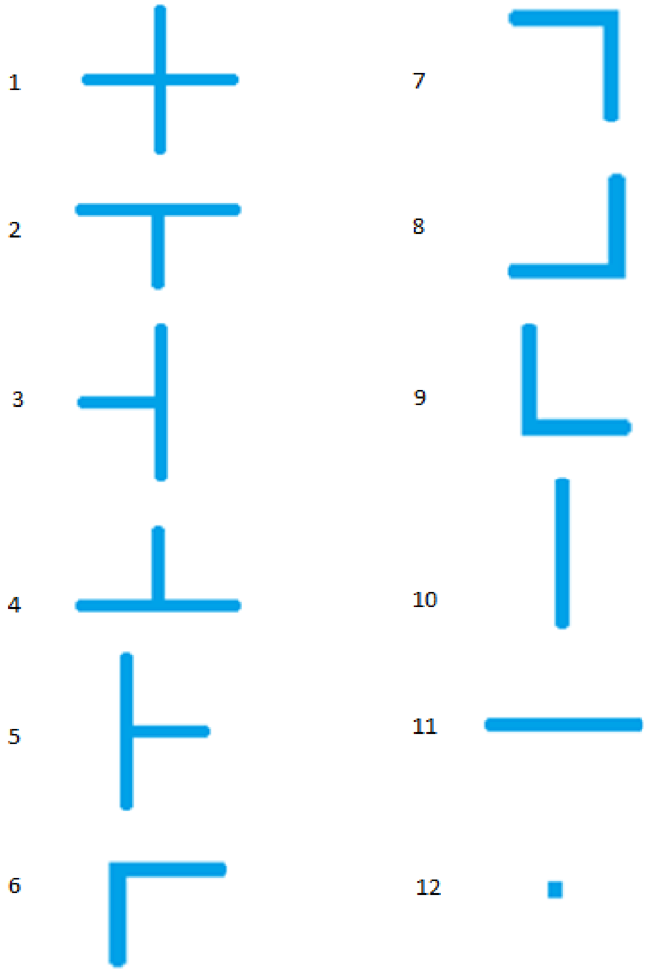
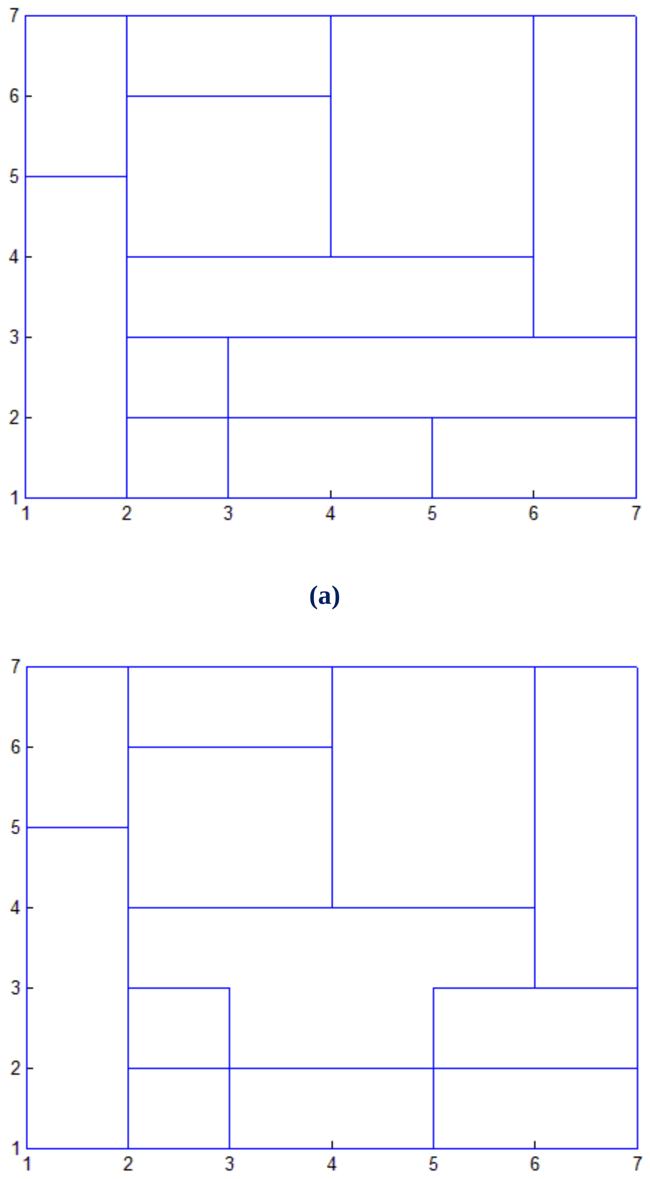


Figure 2.8. Junction Types.
Junctions in 2D meshes are 12 different types as depicted above.
The most special type of junction is node junction, where no knot lines are connected.

2.3.1. Degree limitations

After defining the possible edge interconnections, it is imperative to enlighten some ambiguities a finite element practitioner might encounter. So, despite the fact that all the above junctions are likely to exist in a given T-Mesh, L-junctions can only exist when odd degree is given for one of the axes. But even now a random T-Mesh might seem weird. It is time to make clear that a T-Mesh and the integration mesh do not have to coincide.



(a)

(b)

Figure 2.9. (Created with GeomIso)
(a) Index space of even degree on both axes.
(b) Index space with at least one odd degree. L-junctions can appear.

2.3.2. Global Knot Value Vector

In order to extract the so called anchors in SPLines, we have to define a domain. As far as B-SPLines and NURBS are concerned, if a knot value vector is given per axis, it is straightforward to obtain all possible anchors and accordingly each one of the local knot value vectors. These vectors now can be linked to control points and basis functions can be created.

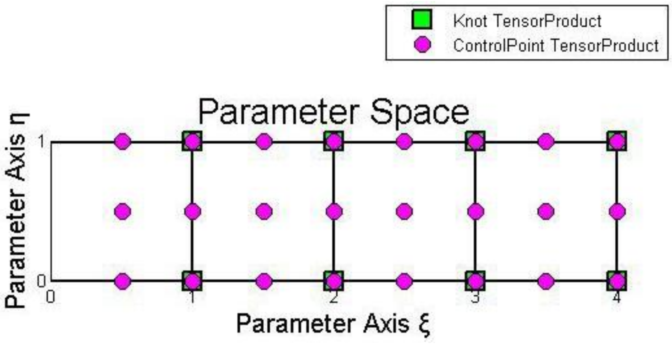


Figure 2.10. (Created with GeomIso)
Parameter space with global knot value vector depicted.

In case of T-SPLines, in order to describe a given T-Mesh, it is vital to provide a global knot value vector. This vector's content should be an ascending sequence of numbers of equal multitude to the indices of index space. Having evaluated the positions of anchors and their indices vector, we are able to match indices with knot values and thus create local knot value vectors.

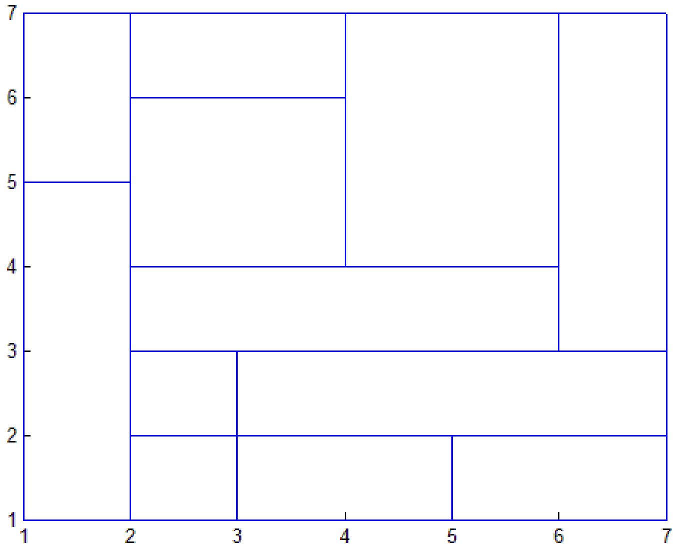
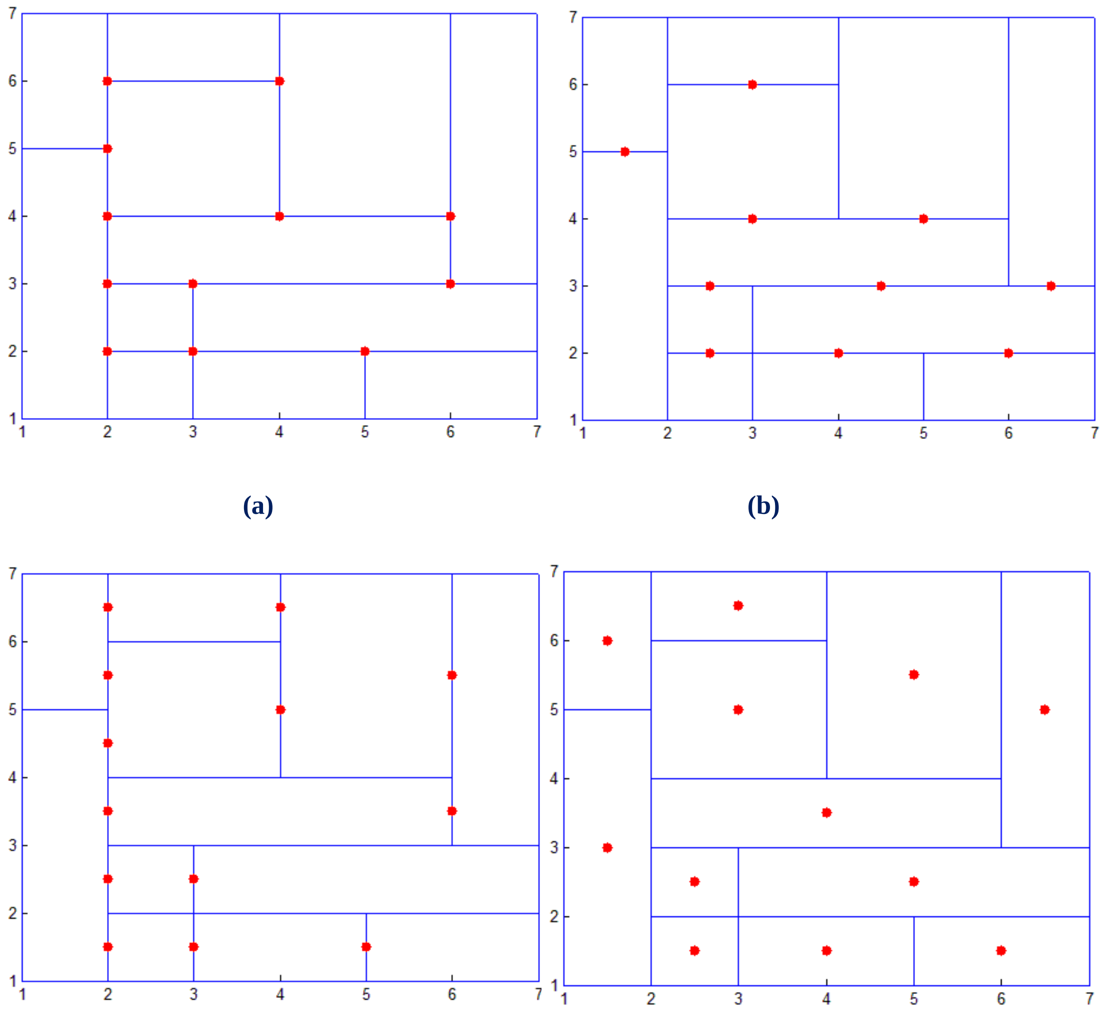


Figure 2.11. (Created with GeomIso)
T-Mesh with global knot value vector.
Ξ={0,0,1,2,3,4,5,6,7,8,8} and H={0,0,1,2,3,4,4}.

2.3.3. Anchor - Control Point

In order to define how each control point acquires a certain local knot value vector, it can be initially described how anchors are produced in such a complex mesh. As shown in the figures below, anchors lie in different points depending on the combination of polynomial degrees chosen for each function. In case of odd degree for all axes, anchors lie on the vertices defined in the T-Mesh.



(a)

(b)

(c)

(d)

Figure 2.12. Index space with anchors as red circles. (Created with GeomIso)
(a) Odd degree for both axes.
(b) Even degree for ξ and odd degree for η.
(c) Odd degree for ξ and even degree for η.
(d) Even degree for both axes.

When both odd and even degrees are present, anchors lie on the center of horizontal or vertical line segments. Finally, when even degree dominates, anchors lie on the center of rectangles.

2.3.4. Local Knot Vector

Generally every anchor contains information about the local knot value vector of each control point, as they are directly related. So, in order to acquire the local knot value vector, we have to find p+2 values that form it. Having previously defined the anchors, it is now deriving effortlessly.

In case of odd polynomial degree for both axes, the anchors lie on the vertices of the T-Mesh that form an angle. So, for such an anchor s_a with coordinates $[\xi_{\alpha,}\ \eta_{\alpha}]$, the local knot vector for axis ξ, Ξ_{α} arises with the following procedure. Initially, the length of the vector is p+2, which means that once more its multitude is odd. The middle position of the vector is occupied by the anchor’s index itself. In order to complete the rest of the empty spaces of the knot value vector, we travel left horizontally and fill blanks with (p+1)/2 values given by the indices of the vertical lines or encountered vertices. The final (p+1)/2 spaces of the vector are defined once more by travelling right and horizontally and writing down with the aforementioned procedure the rest of the knot values. Note that some anchors may be positioned close to the border of the T-Mesh and it may be reached before the local knot value vector is completed. In this case, the standard procedure is to repeat the final index so many times as needed in order to complete the vector. The same procedure is conducted along axis η. The middle position of the knot vector is occupied by the ordinate η_{α} of the anchor, while we budge initially downwards to complete the first (p+1)/2 spots with the η index of the horizontal lines or vertices encountered.

The procedure is similar in even degree. Anchors do not necessarily have integer coordinates, as they are positioned at the center of 2D knot value cells (rectangles). For axis ξ, it becomes obligatory to fill the knot value vector with an even number of p+2 ξ indices. As the total required number of indices is once again even, the vector will be now completed only by moving leftwards and rightwards for ξ and upwards and downwards for η (completing each time p/2+1 knot value spans). This procedure is repeated for every anchor in order to define the corresponding local knot vectors.

In Figure 2.13, examples of local knot value vectors in index space are depicted.

(a) Odd degree in both axes.
Top-left anchor: Ξ_{α}={1,1,2,4,6} and H_{α}={4,5,6,7,7}.

(b) Odd degree for axis ξ and even degree for axis η.
Top-right anchor: Ξ_{α}={1,2,6,7} and H_{α}={3,4,5,6,7}.

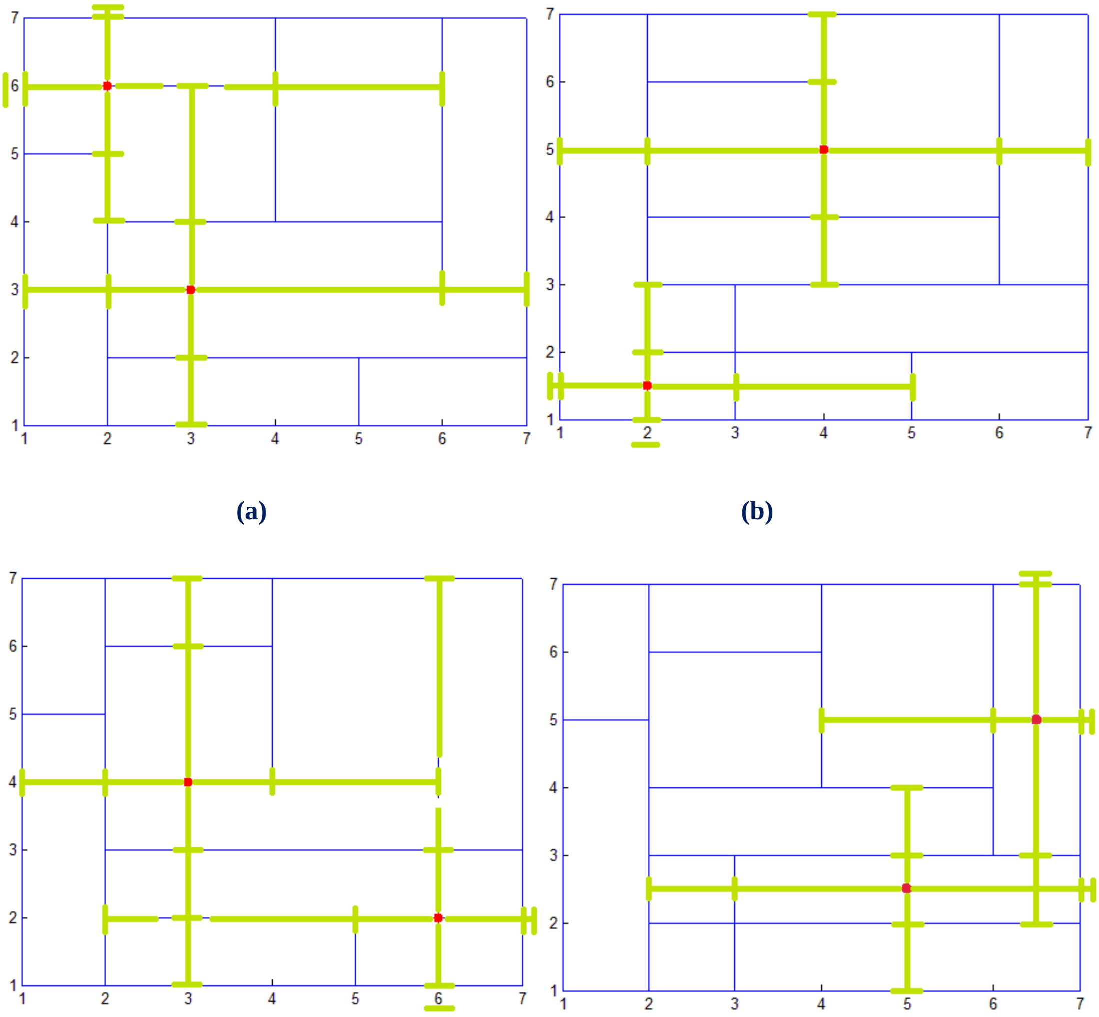
(c) Even degree for axis ξ and odd degree for axis η.
Bottom-right anchor: Ξ_{α}={2,5,6,7,7} and H_{α}={1,1,3,7}.

(d) Even degree in both axes.
Top-right anchor: Ξ_{α}={4,6,7,7} and H_{α}={2,3,7,7}.

At this point, the previous knot value vectors (for the bottom-right anchor, subfigure c), will be calculated analytically for both axes. Initially, one must make clear, if the axes are even or odd (and obviously all of their combinations). As mentioned before, a function with degree (p) has a local knot value vector with (p+2) values. In this example, the degree is equal to 3 (odd) for the ξ axis and 2 (even) for the η axis, thus the knot value vectors will have 5 and 4 values respectively.

$$\Xi_{\alpha}=\{\ ,\ ,\ ,\ ,\ \}$$

$$H_{\alpha}=\{\ ,\ ,\ ,\ \}$$



(a)

(b)

(c)

(d)

Figure 2.13. (Created with GeomIso)
Index space with anchors.
Local knot value vectors for even degree for axis ξ and odd degree for axis η.

Axis ξ: Calculating the knot value vector will start by filling the anchor's ξ-coordinates in the 3rd (=middle) position of the empty vector { , ,6, , }.

Now, moving leftwards (p+1)/2 or equally two values must be found, by extending an imaginary line parallel to axis ξ, where the anchor belongs to. The needed knot values will be shown by the intersections of the imaginary (yellow) line with the vertical lines of the cell elements. The empty vector will now have the following form {2,5,6, , }. Following the same procedure rightwards, a problem will occur. As someone can easily notice, there are not enough vertical lines to "produce" (p+1)/2 or equally two different intersections (values). In this case, a multiplicity in the last intersection is needed to satisfy the knot value vector. The solution to this problem is given by inserting the last intersection with ξ-coordinate ξ_i=7 twice, in order to give the final form of the ξ local knot value vector {2, 5, 6, 7, 7}.

Axis η: The axis is even, thus the vector will have the form { , , , }. Anchor's coordinates won't participate in the construction of the vector. In fact, the anchor will only be a benchmark that will make the procedure easier. Moving upwards, starting from the anchor, the needed values will be found as the intersection points of another imaginary line parallel to axis η and the lines of the cell-elements that are parallel to axis ξ, leading to the following knot value vector's form { , ,3,7}. Following the same procedure this time downwards, yet again, the same problem is encountered (just as the ξ-axis), as there are not enough intersections (knot values). As before, a multiplicity is needed in the last intersection in order to satisfy the knot value vector.

The needed multiplicity of the last point's coordinate is never a fixed number and depends on the axis degree and on the T-Mesh. It occurs as (p+1)/2 intersections until the edge for odd polynomial degree and (p/2+1) for even.

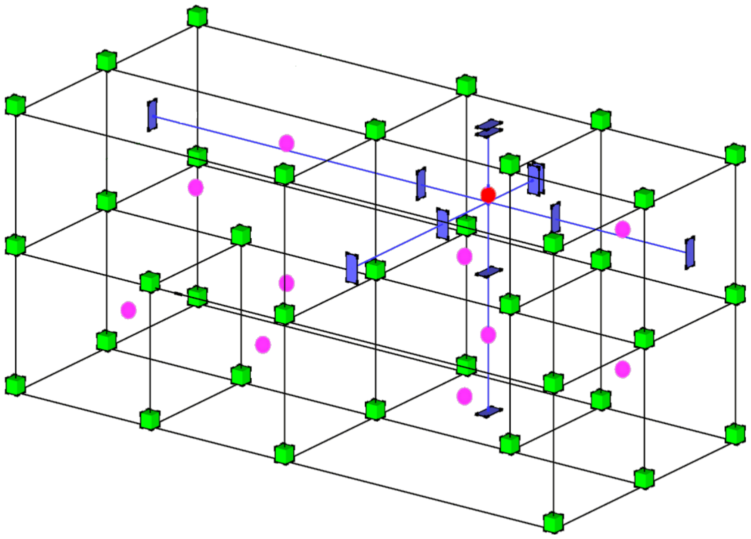


Figure 2.14. Index space with anchors and local knot value vector 3D. (Created by Karras Dimitrios)
Magenta spheres represent three dimensional anchors.
Since quadratic blending functions will be used for all axes, anchors lie on the center of cuboids.
Green cubes represent knots that exist only on junctions.
For the red anchor, local vectors are found by extending lines for all three dimensions.
Purple faces show the values that contribute to the creation of the knot value vector in each dimension.

2.3.5. T-SPLine Blending Function

Given a T-Mesh and degrees p, q for both parametric axes ξ and η, $\Omega \subseteq \square^{2}$ is the index domain that encloses every index a, such that s_a is an anchor. With local knot value vectors $\Xi_\alpha = \left\{ \xi_1, \xi_2, \dots, \xi_{p+1}, \xi_{p+2} \right\}$, $H_\alpha = \left\{ \eta_1, \eta_2, \dots, \eta_{p+1}, \eta_{p+2} \right\}$, we define for both axes the univariate functions recursively as follows.

Beginning with constant piecewise polynomials (p=0), we have:

$$N_{\mathrm{i},0}(\xi) = \begin{cases} 1 & \text{if } \xi_{\mathrm{i}} \le \xi < \xi_{\mathrm{i}+1} \\ 0 & \text{otherwise} \end{cases} \tag{0.1}$$

For p=1, 2, 3…, we define higher degree functions by the Cox-de Boor recursive formula:

$$N_{\mathrm{i},p}(\xi) = \frac{\xi - \xi_{\mathrm{i}}}{\xi_{\mathrm{i}+p} - \xi_{\mathrm{i}}} N_{\mathrm{i},p-1}(\xi) + \frac{\xi_{\mathrm{i}+p+1} - \xi}{\xi_{\mathrm{i}+p+1} - \xi_{\mathrm{i}+1}} N_{\mathrm{i}+1,p-1}(\xi) \tag{0.2}$$

with the assumption of $\frac{0}{0} \square 0$.

Given the fact that the generated functions no longer stem from a global knot value vector, rather a local one of arbitrary length, they cannot be called basis functions any more as no specific space is defined for them. Instead, they will be referred to as blending functions. We can associate each and every knot value vector with a single univariate function of as described above. Given many univariate functions with not related knot value vectors, we could assign coefficients to them and thus create a curve as follows:

$$C(\xi) = \sum_{\mathrm{i}=1}^{\mathrm{n}} \left\{ N_{\mathrm{i,p}}(\xi) \cdot X_{\mathrm{i}} \right\} \tag{0.3}$$

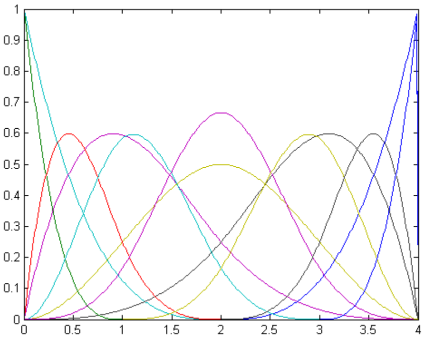
In this part, it is vital to explain the distinction between the terms order and degree. The definition of degree of the polynomial is given by CAD designers, as the highest degree of all its terms. The term order refers to a related concept. It expresses the maximum number of terms a polynomial has. Usually a polynomial of p-th degree has order of p+1. In general, in this thesis, the adopted notations follow those of the published isogeometric analysis papers, so that the reader can easily read through it and be able to refer to the sources at the end for more sophisticated explanations. On the other hand, in the code developed by the writer, the use of such abbreviations is discouraged and rather replaced with the use of full names.

2.3.6. T-SPLine Blending Function Properties

According to “Isogeometric Analysis using T-Splines” and “T-SPLines as a Design-Through-Analysis Technology”, T-SPLine basis functions possess the following important properties:

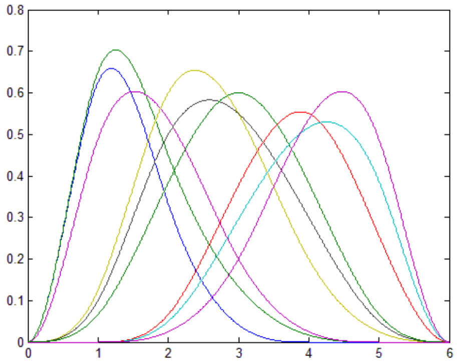
1. Local support:

$$N_{\mathrm{i,p}}(\xi)=0 \ \ \forall \xi \notin [\xi_{\mathrm{i}}, \xi_{\mathrm{i+p+1}}) \tag{0.4}$$



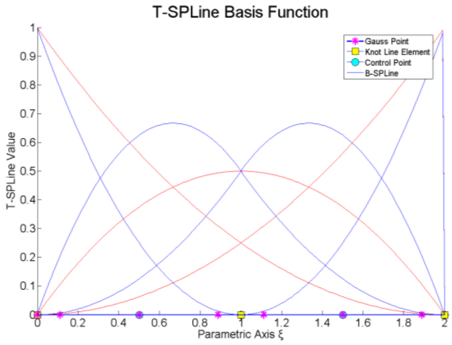
2. Non-negativity:

$$N_{\mathrm{i,p}}(\xi)\geq 0 \ \ \forall \xi,\mathrm{i,p} \tag{0.5}$$



3. Partition of unity:

$$\sum_{\mathrm{i=1}}^{\mathrm{n}} N_{\mathrm{i,p}}(\xi)=1 \ \ \forall \xi,\mathrm{p} \tag{0.6}$$



4. C^{p-m} continuity across knots with multiplicity m.

2.3.6.1. Local Support

Local support means that blending functions are positive only in certain spans in parameter space. This can be expressed by:

$$N_{i,p}(\xi) = 0 \ \ \forall \xi \notin [\xi_i, \xi_{i+p+1})$$

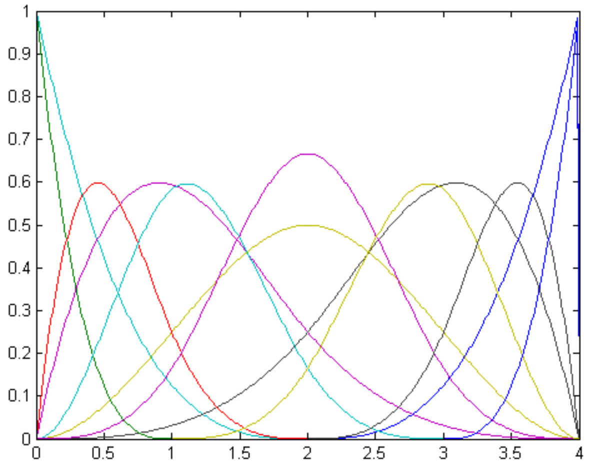


Figure 2.15. Parameter space with blending functions per axis ξ. (Created with GeomIso) Each one of the varied colored shape functions has different support.

Local support is a result of the recursive Cox de Boor algorithm. For the creation of one univariate function of degree p, two consecutive blending functions of degree p-1 are used. For the creation of each of those functions, another two consecutive functions will be used. Since one function will be common for both of them, we conclude that three functions are finally needed. Inductively, p+1 constant functions are required in order to create one blending function of degree p. Each box in T-SPLines has support of one span. This means that constant functions in contrast to NURBS can have a support of more than one consecutive knot value spans. The property remains, as the final blending function of degree p support is defined by the union of the lower degree functions, hence now p+1 consecutive knot spans.

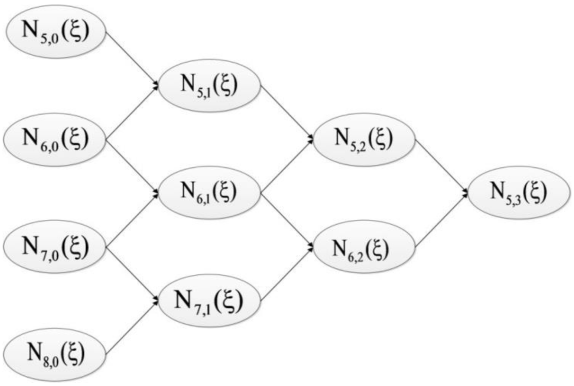


Figure 2.16.

Lower order basis functions required for the creation of $N_{5,3}(\xi)$.

2.3.6.2. Non-negativity

It has been proven that:

$$N_{i,p}(\xi) \geq 0 \quad \forall \xi, i, p \tag{0.7}$$

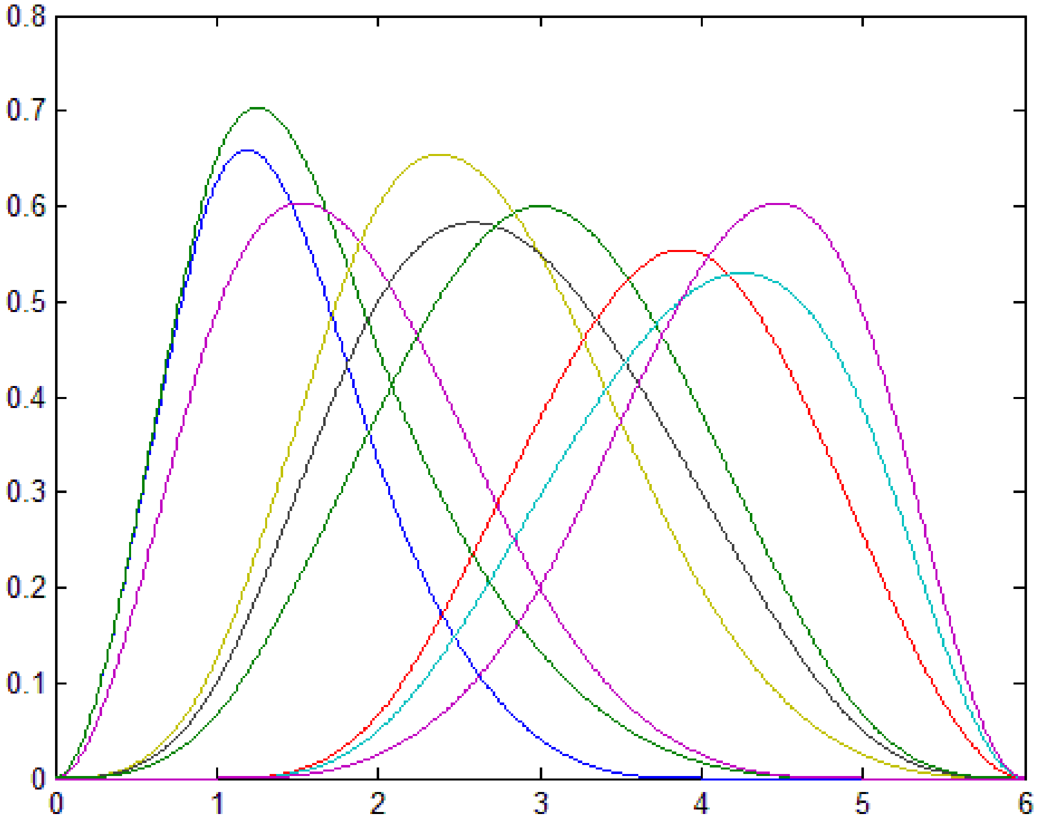


Figure 2.17. (Created with GeomIso)
T-SPLine blending functions.

In Figure 2.17, we can see various functions of degree p=3, which are all non-negative for every ξ_i. This is an important feature that both B-SPLines and T-SPLines have. It is of major importance in isogeometric analysis, as the common practice of finite elements does not impose positive shape functions. As this property applies for every axis, it can be easily generalized for 2D and 3D blending functions as well. Inductively, the same principle applies for T-SPLine surfaces:

$$T = \frac{\sum_{A=1}^{n} P_A \cdot w_A \cdot N_A}{\sum_{A=1}^{n} w_A \cdot N_A} \tag{0.8}$$

where P_A are the control points, w_A their weights and N_A the blending functions.

2.3.6.3. Partition of Unity

Partition of unity is established by:

$$\sum_{\mathrm{i=1}}^{\mathrm{n}} N_{\mathrm{i,p}}(\xi) = 1 \quad \forall \xi,\mathrm{p} \tag{0.9}$$

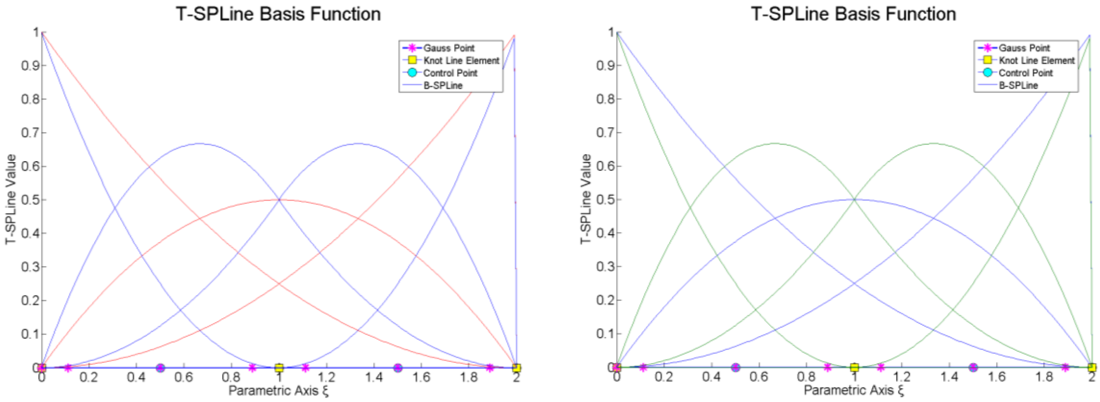


Figure 2.18. (Created with GeomIso)
T-SPLine blending functions for open knot value vector in 2D.

Partition of unity is a property easily found in case of NURBS, as a common section of a specific knot value can reveal it. On the other hand, in T-SPLines, this is not that obvious, since partition of unity can only be examined for as many dimensions as the simulation demands. For instance, in 1D applications of T-SPLines, partition of unity is examined only in one dimension like NURBS. Higher degree applications require that partition of unity is valid for every point regardless of ξ, η and ζ. This concludes that this property is far more difficult to be verified in T-SPLine technology, as it cannot be generalized from a specific section as in NURBS.

Partition of unity applies for multi-dimensional shape functions as well. In 2D, it is expressed as:

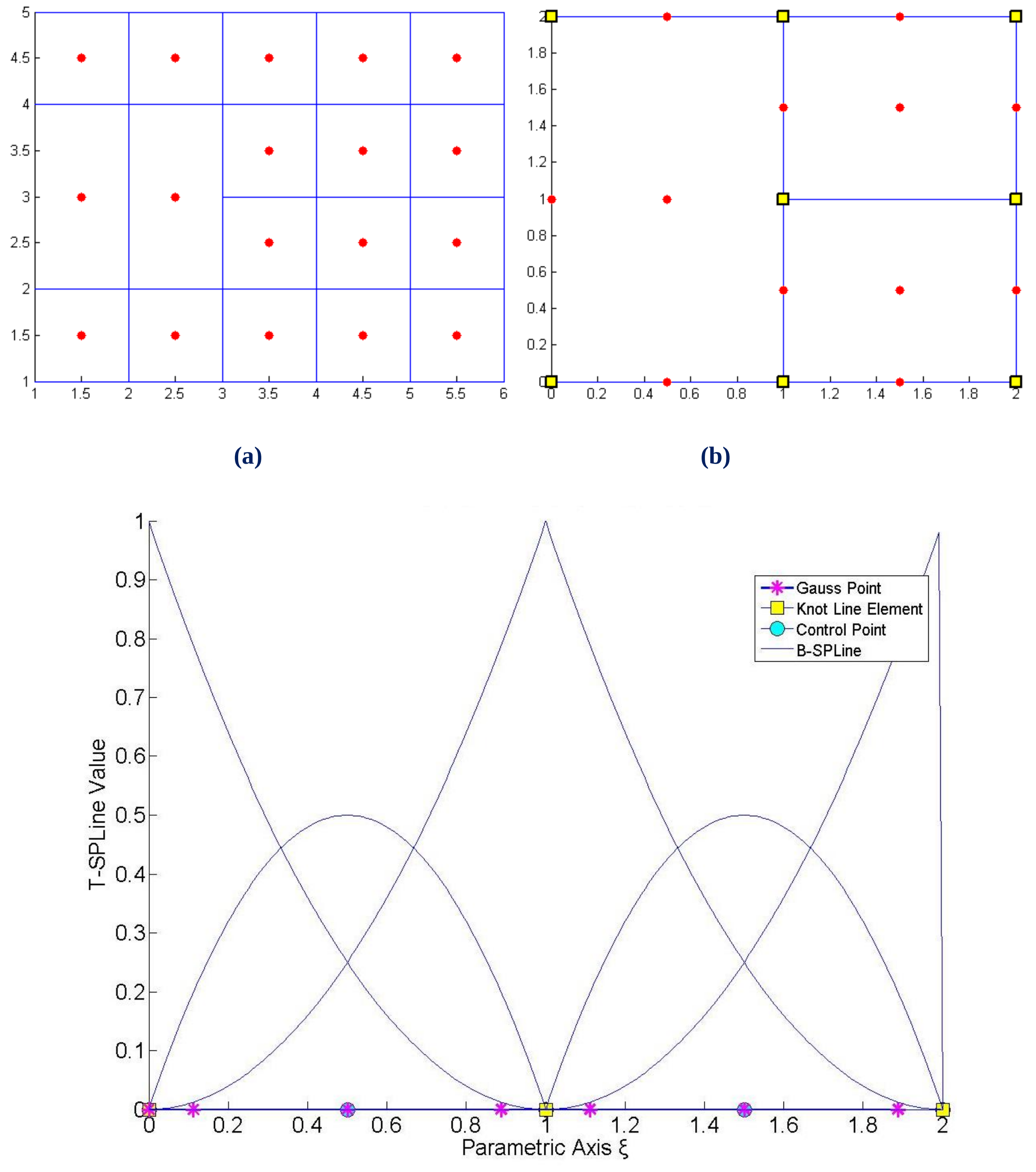
$$\sum_{\mathrm{i=1}}^{\mathrm{n}} \sum_{\mathrm{i=1}}^{m} N_{\mathrm{i,p}}(\xi) \cdot M_{\mathrm{j,q}}(\eta) = 1 \quad \forall \xi,\eta,\mathrm{p},\mathrm{q} \tag{0.10}$$

Analogously, 3D blending functions also possess partition of unity as described before.

$$\sum_{\mathrm{i=1}}^{\mathrm{n}} \sum_{\mathrm{j=1}}^{m} \sum_{\mathrm{k=1}}^{l} N_{\mathrm{i,p}}(\xi) \cdot M_{\mathrm{j,q}}(\eta) \cdot L_{\mathrm{k,r}}(\zeta) = 1 \quad \forall \xi,\eta\zeta,\mathrm{p},\mathrm{q},\mathrm{r} \tag{0.11}$$

2.3.6.4. $C^{p\text{-}m}$ Continuity along knots of m- multiplicity

This property exists like in B-SPLines. Given a knot value vector of an anchor, if the multiplicity of a knot value is greater than one, the blending function has $C^{p\text{-}m}$ continuity. This means that this specific blending function can produce p-m continuous derivatives. Continuity less than C^0 is not acceptable for internal knots. As a result, a certain knot value can be repeated at most p times in a local knot value vector. Note that as continuity decreases, blending functions tend to become steeper.



(a) **(b)**

(c)

Figure 2.19. (Created with GeomIso)
T-SPLine blending function of degree p=2.
Due to knot repetition, blending function with reduced continuity are created.

2.4. T-SPLine Shape Function

Given the local knot value vectors of an anchor for all axes ξ, η, ζ, the univariate T-SPLine blending function $N_{i,p}(\xi)$, $M_{j,q}(\eta)$, $L_{k,r}(\zeta)$ are created. With the aid of these functions, the multivariate T-SPLine shape functions are created. For the one dimensional case, we obtain:

$$R_i^p(\xi) = \frac{N_i^p(\xi)\cdot w_i}{W(\xi)} = \frac{N_i^p(\xi)\cdot w_i}{\sum_{\hat{i}=1}^{n} N_{\hat{i}}^p(\xi)\cdot w_{\hat{i}}}$$

Inductively for 2D and 3D cases:

$$R_{i,j}^{p,q}(\xi,h) = \frac{N_i^p(\xi)\cdot M_j^q(\eta)\cdot w_{i,\mathrm{j}}}{\sum_{\hat{i}=1}^{n}\sum_{\hat{j}=1}^{m} N_{\hat{i}}^p(\xi)\cdot M_{\hat{j}}^p(\xi)\cdot w_{\hat{i},\hat{j}}}$$

$$R_{i,j}^{p,q}(\xi,h) = \frac{N_i^p(\xi)\cdot M_j^q(\eta)\cdot L_k^r(\zeta)\cdot w_{i,\mathrm{j,k}}}{\sum_{\hat{i}=1}^{n}\sum_{\hat{j}=1}^{m}\sum_{\hat{k}=1}^{l} N_{\hat{i}}^p(\xi)\cdot M_{\hat{j}}^p(\xi)\cdot L_{\hat{k}}^r(\eta)\cdot w_{\hat{i},\hat{j},\hat{\mathrm{k}}}}$$

2.4.1. T-SPLine Shape Function Properties

Properties of T-SPLine blending functions can be inductively proven for shape functions as well:

1. Local support:

$$R_{\mathrm{i,p}}\left(\xi,\eta\right) = 0 \quad \forall \xi,\eta \notin [\xi_{\mathrm{i}},\xi_{\mathrm{i+p+1}}) \cup [\eta_{\mathrm{i}},\eta_{\mathrm{i+p+1}}) \tag{0.12}$$

2. Non-negativity:

$$R_{\mathrm{i,p}}(\xi,\eta) \geq 0 \quad \forall \xi,\eta,\mathrm{i,j,p,q} \tag{0.13}$$

3. Partition of unity:

$$\sum_{\mathrm{i=1}}^{\mathrm{n}} R_{\mathrm{i,j,p,q}}(\xi,\eta) = 1 \quad \forall \xi,\eta,\mathrm{p,q} \tag{0.14}$$

4. C^{p-m} continuity across knots with multiplicity m.

5. Linear independence of blending functions.

2.4.1.1. Local Support

Local support means that blending functions are positive only in certain spans in parameter space. This can be expressed by:

$$R_{i,p}\left(\xi,\eta\right)=0 \quad \forall\xi,\eta \notin [\xi_i,\xi_{i+p+1})\cup[\eta_i,\eta_{i+p+1})$$

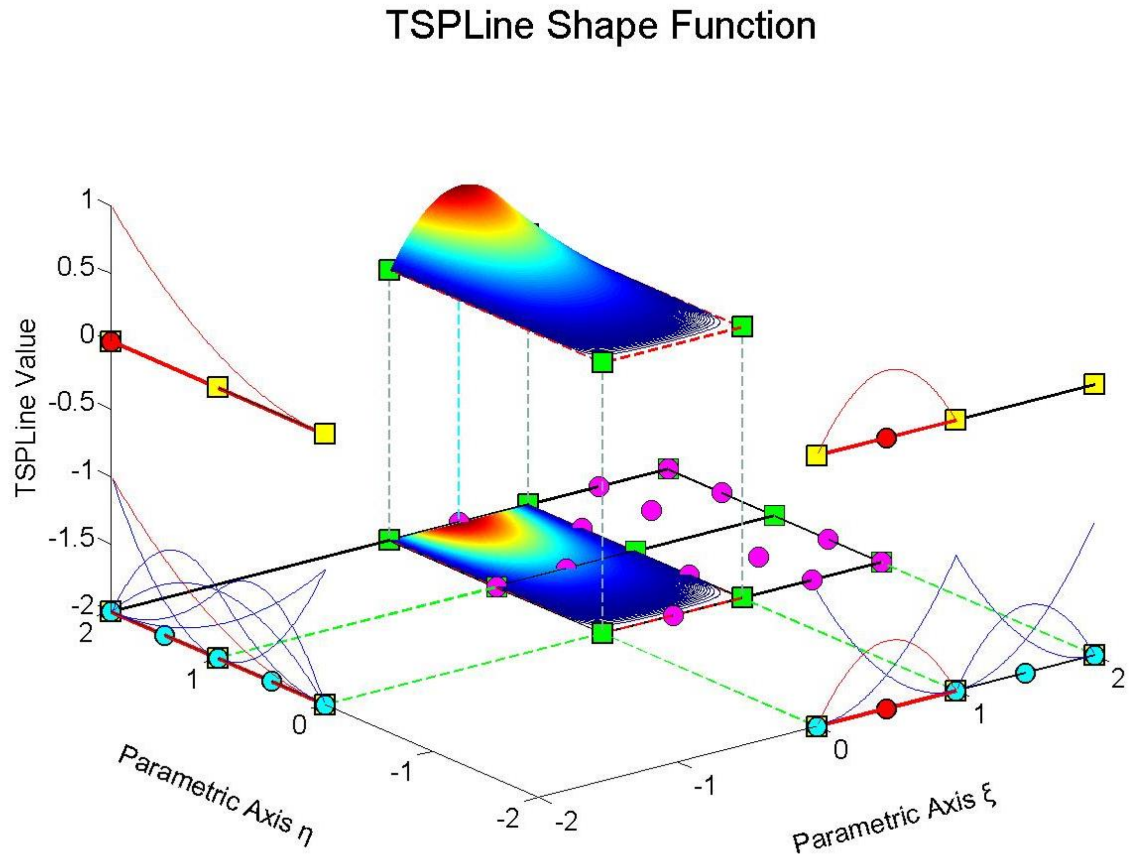


Figure 2.20. Parameter space. (Created with GeomIso)
In this figure, local support of this function can be seen for both axes.
For axis ξ, the support of the depicted shape function is [0,1], while for η is [0,2].
The cyan circle shows the projection of the anchors in 1D.
Magenta circles are the real parametric coordinates of the tensor product anchors.
The red colored 1D blending functions are combined to give the" countered" shape,
i.e. the shape functions and their support in 2D.

Local support is a result of the recursive Cox de Boor algorithm. For the creation of one univariate function of degree p, two consecutive blending functions of degree p-1 are used. For the creation of each of those functions, another two consecutive functions will be used. Since one function will be common for both of them, we conclude that three functions are finally needed. Inductively, p+1 constant functions are required in order to create one blending function of degree p.

Each box in T-SPLines has support of one span. That means that constant functions in contrast to NURBS can have a support of more than one consecutive knot value spans. The property remains and the support of the final blending function of degree p is defined by the union of the lower degree functions, hence now p+1 consecutive knot spans.

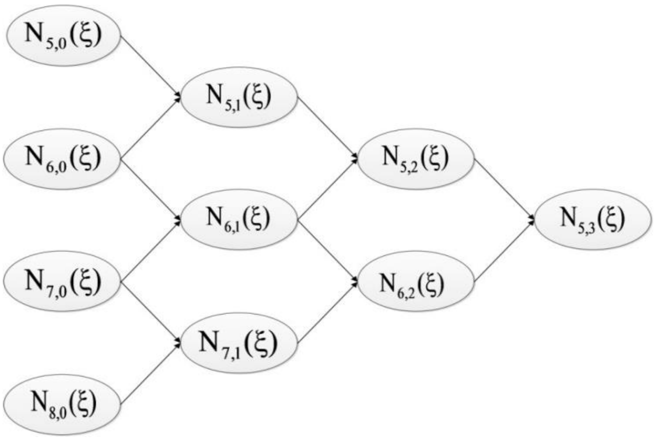
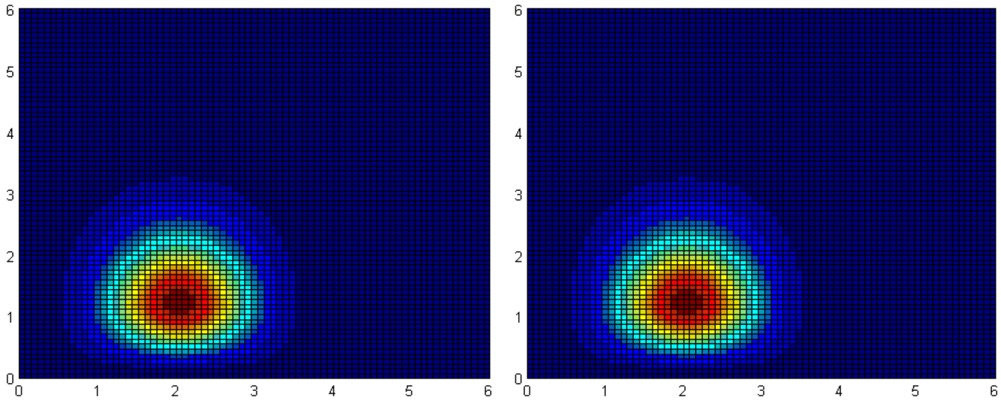
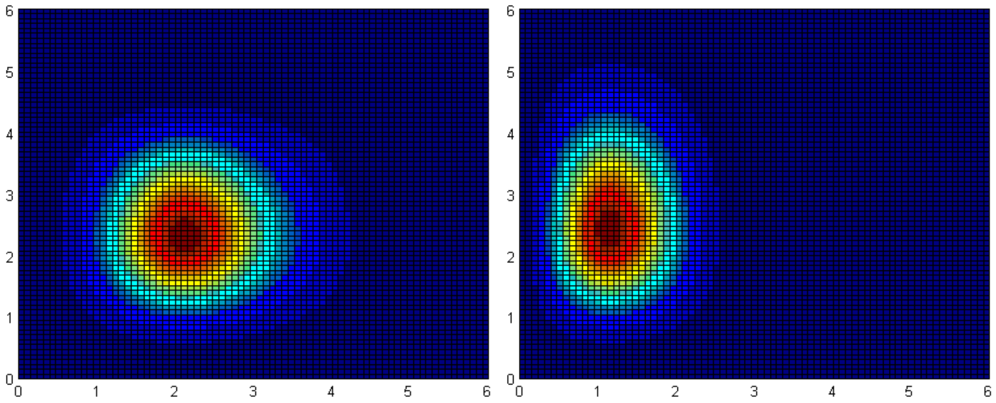


Figure 2.21.

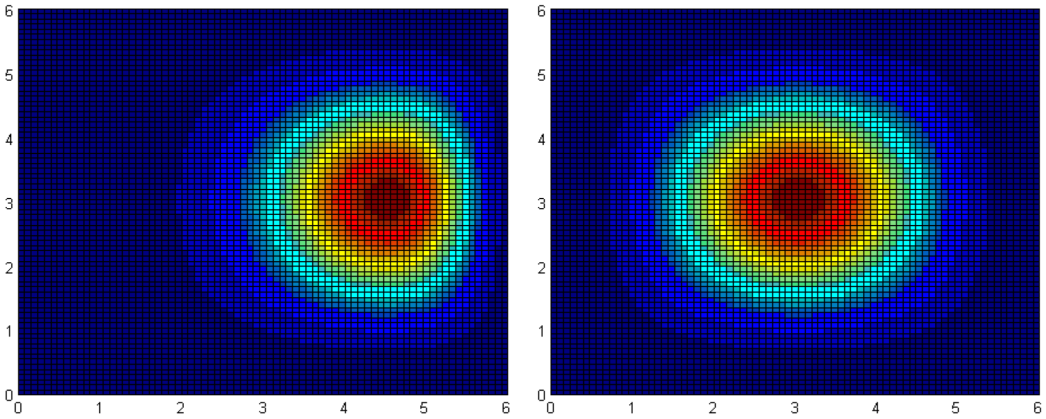
Lower order basis functions required for the creation of $N_{5,3}(\xi)$.



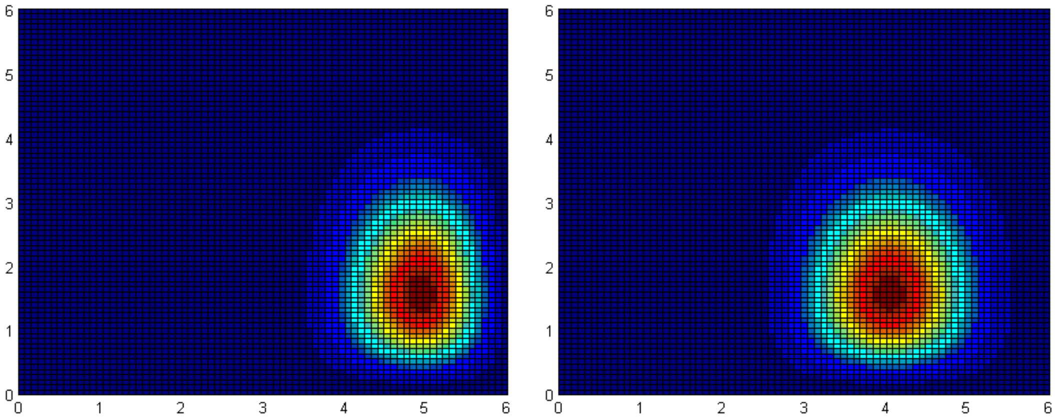
(a)



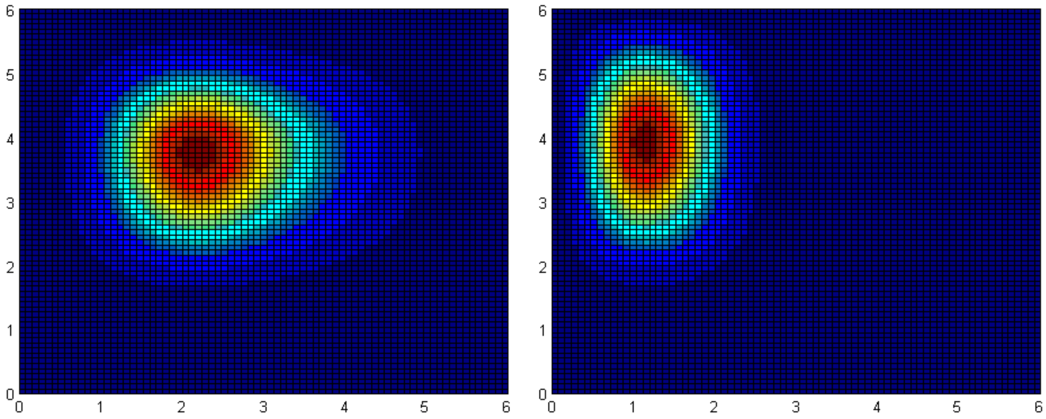
(b)


0
1
2
3
4
5
6
0
1
2
3
4
5
6
0
1
2
3
4
5
6
0
1
2
3
4
5
6

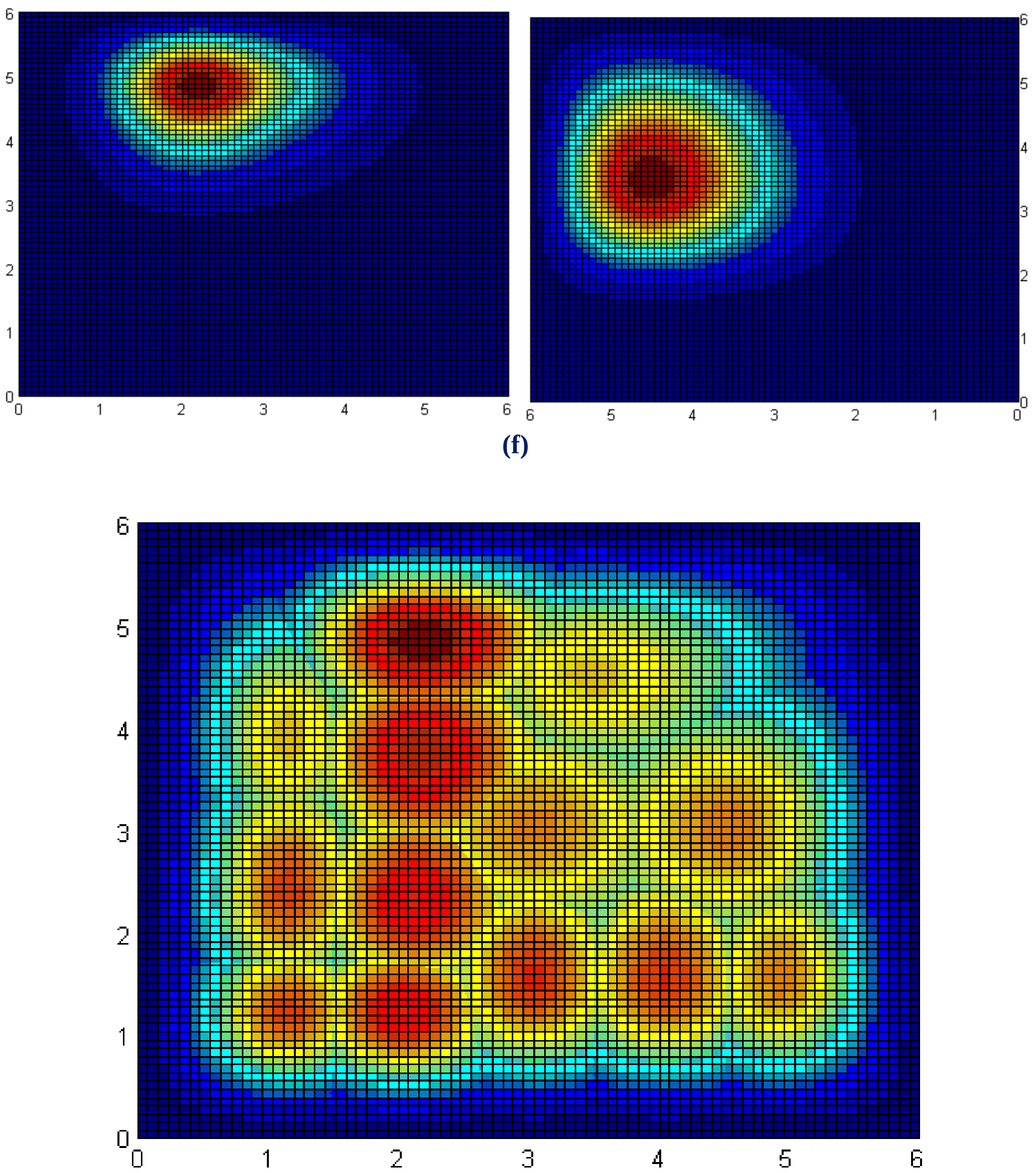
(c)


0
1
2
3
4
5
6
0
1
2
3
4
5
6
0
1
2
3
4
5
6
0
1
2
3
4
5
6

(d)


0
1
2
3
4
5
6
0
1
2
3
4
5
6
0
1
2
3
4
5
6
0
1
2
3
4
5
6

(e)



(f)

(g)

Figure 2.22. (Created with GeomIso)
Subfigures (a) to (f) depict each of the supports of the anchors.
Areas that are not colored blue are considered as the anchors' supports.
Note that the support differs for the various functions.
p+2 spans support per axis for each anchor create different pictures of supports.
This can easily be explained, as one span may contain more than one T-SPLine blending functions.
Subfigure (g) shows the magnitude that each one of them contributes to partition of unity.

Note that support in 2D and 3D T-SPLines is not constant per axis. Despite the fact that every support is p+2 spans, this number is not constant for all axes and shape functions. T-Mesh topology defines each time the local support and it may vary, as one span in T-SPLines does not necessarily coincide with one knot value span. This is why, as shown in figure 2.22 above, shape functions affect different types of the domain.

2.4.1.2. Non-negativity

It has been proven that:

$$R_{\mathrm{i,p}}(\xi,\eta)\geq 0 \quad \forall\xi,\eta,\mathrm{i,j,p,q} \tag{0.15}$$

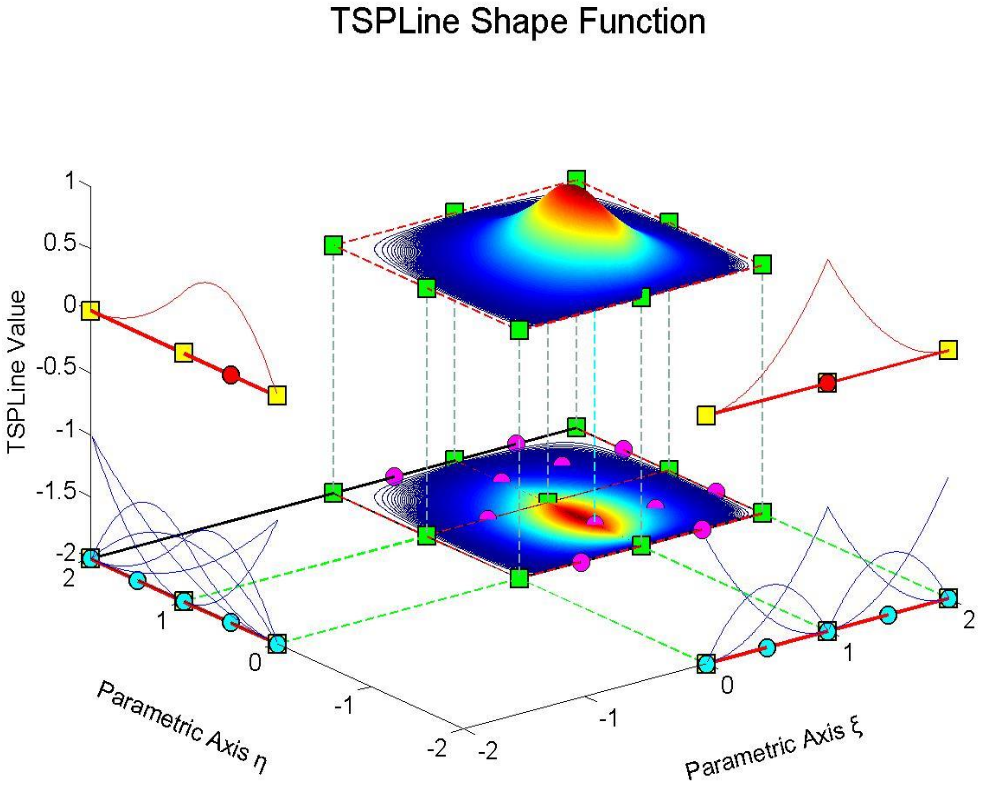


Figure 2.23. (Created with GeomIso)
T-SPLine shape functions.

In Figure 2.23, we can see various functions of degree p=3. All blending functions are positive for every ξ_i. This is an important feature that both B-SPLines and T-SPLines possess. It is of major importance in isogeometric analysis, as the common practice of finite elements does not impose positive shape functions. As this property applies for every axis, it can easily generalized for 2D and 3D shape functions as well. Inductively, the same principle applies for T-Spline surfaces:

$$T=\frac{\sum_{A=1}^{n} P_A\cdot w_A\cdot N_A}{\sum_{A=1}^{n} w_A\cdot N_A} \tag{0.16}$$

where P_A are the control points, w_A their weights and N_A the blending functions.

2.4.1.3. Partition of Unity

Partition of unity is established by:

$$\sum_{i=1}^{n} R_{i,j,p,q}(\xi,\eta)=1 \quad \forall \xi,\eta,p,q \tag{0.17}$$

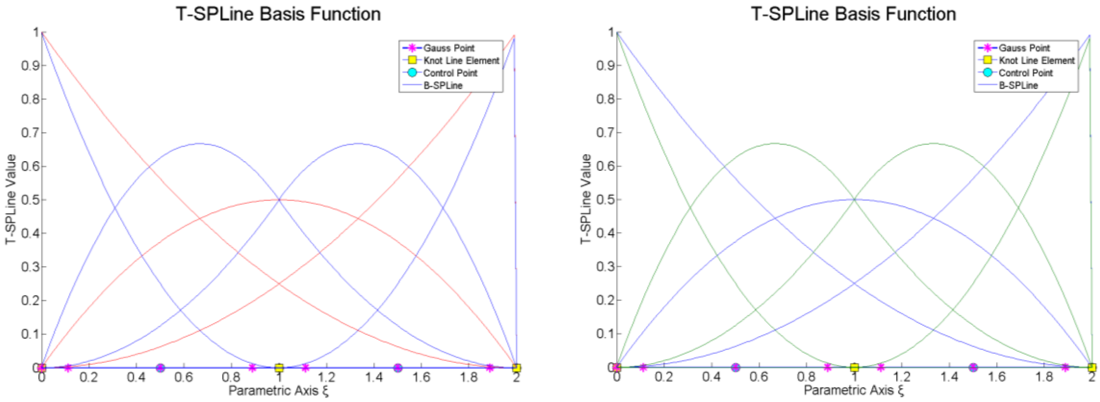


Figure 2.24. (Created with GeomIso)
T-SPLine blending functions for open knot value vector in 2D.

Partition of unity is a property easily found in case of NURBS, as a common section of a specific knot value can reveal it. On the other hand, in T-SPLines, this is not that obvious, since partition of unity can only be examined for as many dimensions as the simulation demands. For instance, in 1D applications of T-SPLines partition of unity is examined only in one dimension like NURBS. Higher degree applications require that partition of unity is valid for every point regardless of ξ, η, ζ. This leads to the conclusion that this property is quite more difficult to be verified, because it cannot be generalized from a specific section as NURBS.

Partition of unity applies for multi-dimensional shape functions as well. In 2D, it is expressed as:

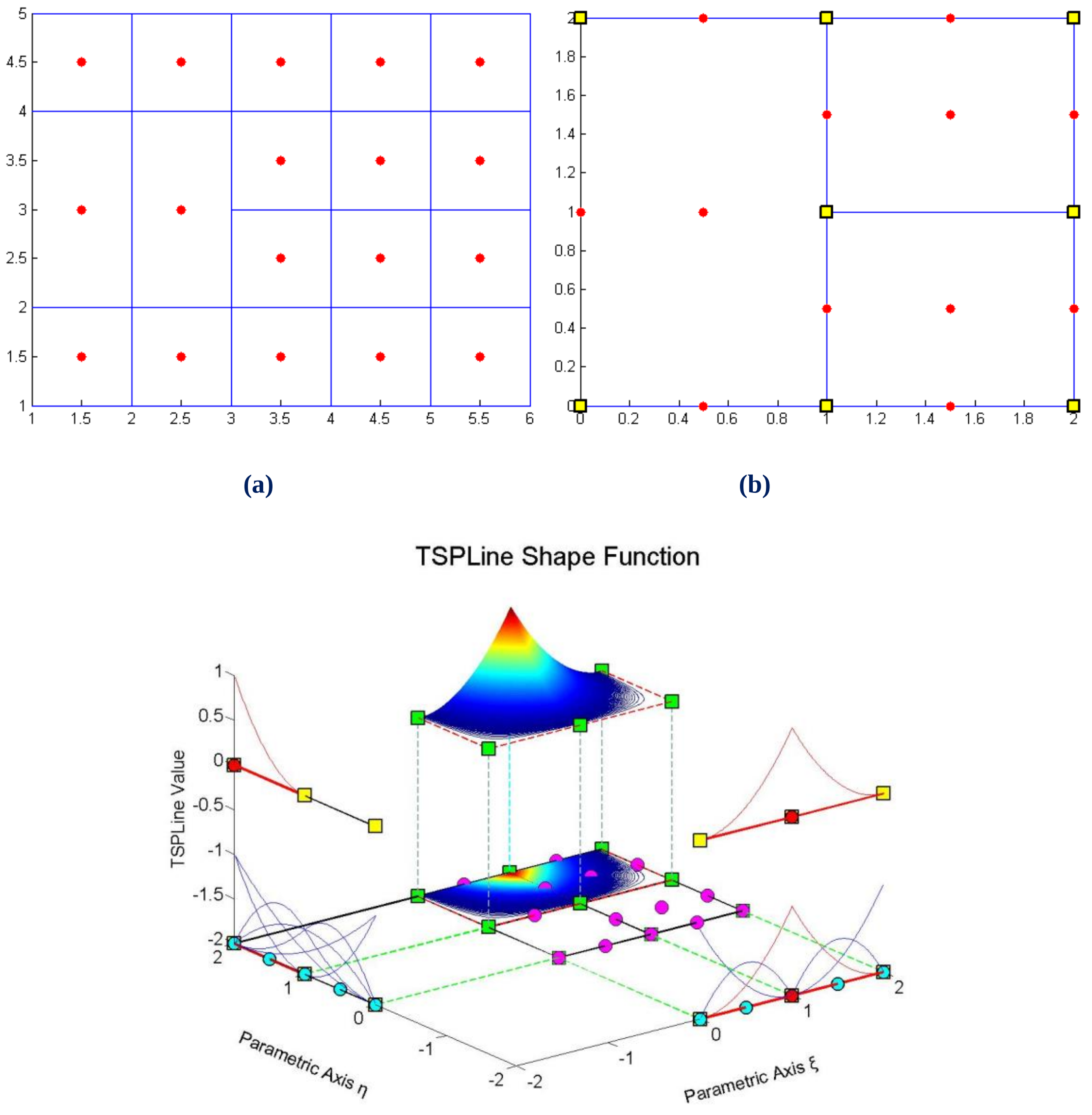
$$\sum_{i=1}^{n}\sum_{i=1}^{m} N_{i,p}(\xi)\cdot M_{j,q}(\eta)=1 \quad \forall \xi,\eta,p,q \tag{0.18}$$

Analogously, 3D blending functions also possess partition of unity, as described before.

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l} N_{i,p}(\xi)\cdot M_{j,q}(\eta)\cdot L_{k,r}(\zeta)=1 \quad \forall \xi,\eta\zeta,p,q,r \tag{0.19}$$

2.4.1.4. $C^{p\text{-}m}$ Continuity along knots of m- multiplicity

This property exists like in NURBS. Given a knot value vector of an anchor, if the multiplicity of a knot value is greater than one, the blending function has $C^{p\text{-}m}$ continuity. This means that this specific blending function can produce p-m continuous derivatives. Continuity less than C^0 is not acceptable for internal knots, meaning that a certain knot value can be repeated in a local knot value vector at most p times. Note that as continuity decreases, blending functions tend to become more steep.



(a)

(b)

(c)

Figure 2.25. (Created with GeomIso)
Quadratic T-SPLine shape function is created by two blending functions, which have reduced continuity due to knot value multiplicity.
For ξ: local knot value vector {0,1,1,2} and C^0 continuity.
For η: local knot value vector {0,0,0,1} and C^{-1} continuity.

2.4.1.5. Linear Independence

It is mathematically proven that T-Meshes in general do not produce linear independent T-SPLine blending functions. This is a major drawback of T-SPLines, hence linear independence of basis functions is crucial for isogeometric analysis and computational mechanics in general. In that case, the system of equations is not invertible and the stiffness matrix as well. As a consequence, not every T-Mesh is acceptable for analysis. It is proven that meshes that follow certain constrains always define an independent base of blending functions.

2.4.1.6. Analysis Suitable Elemental T-Mesh

An analysis-suitable T-SPLine is one whose extended T-mesh is analysis suitable. In an analysis-suitable extended T-mesh, no T-junction extensions intersect.

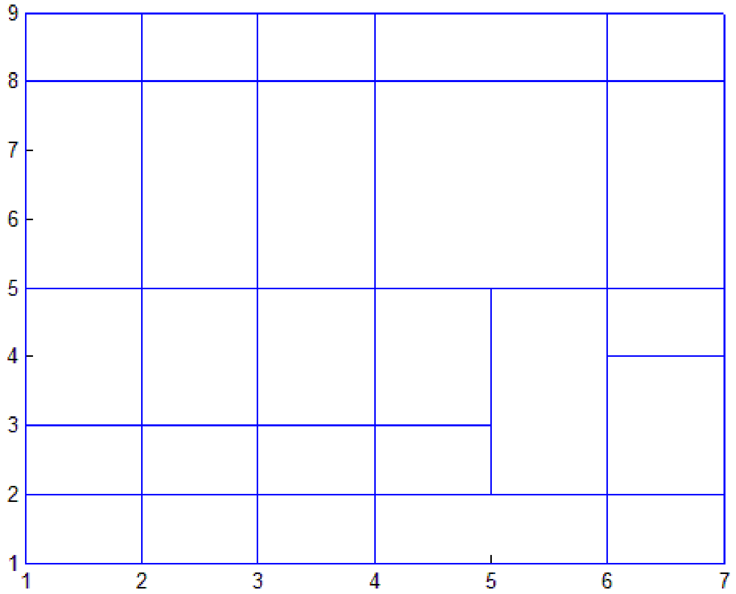


Figure 2.26. (Created with GeomIso)
Analysis Non-Suitable Mesh.

As A. Scott defines, analysis suitable is a mesh whose extended mesh is analysis suitable. Figure 2.26 illustrates a typical T-Mesh. Nevertheless, this mesh cannot be considered analysis suitable. By extended mesh, we define the mesh in Figure 2.27, which is obtained by extending the T-junctions in both directions. For instance, if a T-junction is created by removing the horizontal left line of a cross junction, then the extensions will be the following. The main extension will be on the vanished side, i.e. the black dashed line, of the junction and it can be extended throughout the mesh, but only the segment of the extension until the next knot line or junction counts. The other extension is the red dashed line as depicted below, that has an opposite direction to the previous one.

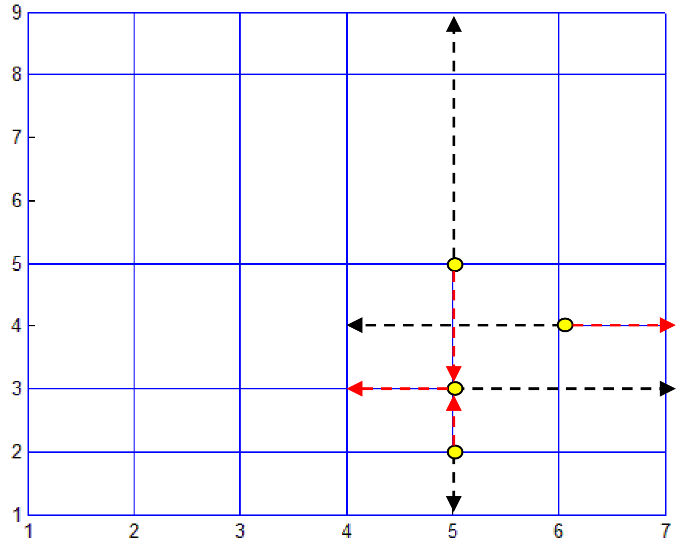


Figure 2.27. (Created with GeomIso)
Extended Analysis Non-Suitable Mesh.

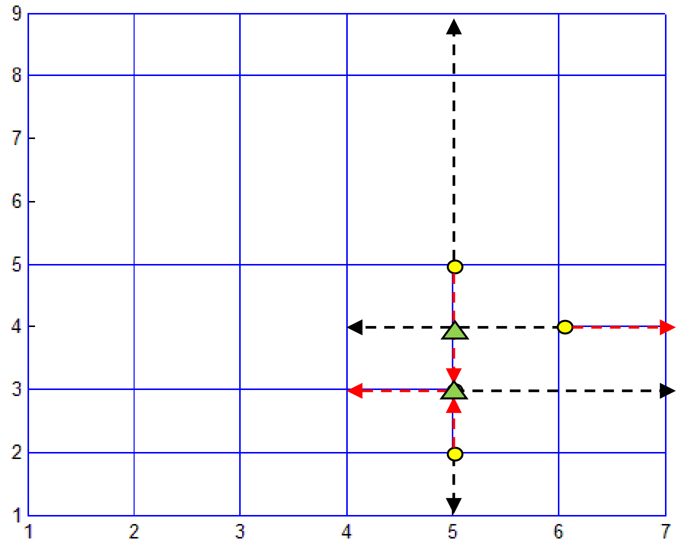


Figure 2.28. (Created with GeomIso)
Extended Analysis Non-Suitable Mesh with intersections.

As we can see, the numerous extensions of the T-junctions intersect with each other. The intersection points are given above as green triangles. We conclude that if such intersections exist, then the T-Mesh is not analysis suitable. Luckily with a simple refinement of the mesh, it is plausible to convert the mesh to an analysis suitable one, which creates linearly independent blending functions.

2.4.2. Derivatives

Taking into account that one univariate blending function derives from the combination of Cox-de Boor recursive algorithm and a local knot vector, we conclude that univariate blending functions and inductively multivariate blending functions derive from the following recursive definition.

$$\frac{d}{d\xi}N_{\mathrm{i,p}}(\xi)=\frac{p}{\xi_{\mathrm{i+p}}-\xi_{\mathrm{i}}}\cdot N_{\mathrm{i,p\text{-}1}}(\xi)-\frac{p}{\xi_{\mathrm{i+p+1}}-\xi_{\mathrm{i+1}}}\cdot N_{\mathrm{i+1,p\text{-}1}}(\xi) \qquad (0.20)$$

This leads us to a generalized equation for the k-th derivative:

$$\frac{d^{\mathrm{k}}}{d\xi^{\mathrm{k}}}N_{\mathrm{i,p}}(\xi)=\frac{p!}{(p-k)!}\cdot\sum_{\mathrm{j=0}}^{\mathrm{k}}a_{\mathrm{k,j}}\cdot N_{\mathrm{i+j,p\text{-}k}}(\xi) \qquad (0.21)$$

$$a_{0,0}=1$$

$$a_{\mathrm{k,0}}=\frac{a_{\mathrm{k\text{-}1,0}}}{\xi_{\mathrm{i+p\text{-}k+1}}-\xi_{\mathrm{i}}}$$

$$a_{\mathrm{k,j}}=\frac{a_{\mathrm{k\text{-}1,j}}-a_{\mathrm{k\text{-}1,j\text{-}1}}}{\xi_{\mathrm{i+p+j\text{-}k+1}}-\xi_{\mathrm{i+j}}},\quad \mathrm{j=1,...,k\text{-}1}$$

$$a_{\mathrm{k,k}}=\frac{-a_{\mathrm{k\text{-}1,k\text{-}1}}}{\xi_{\mathrm{i+p+1}}-\xi_{\mathrm{i+k}}}$$

In addition, partial derivatives of two-dimensional T-SPLine shape functions can be easily obtained by application of the quotient rule:

$$\frac{\partial}{\partial\xi}R_{\mathrm{i,j}}^{\mathrm{p,q}}(\xi,\eta)=\left(\frac{d}{d\xi}N_{\mathrm{i,p}}(\xi)\right)\cdot M_{\mathrm{j,q}}(\eta) \qquad (0.22)$$

$$\frac{\partial}{\partial\eta}R_{\mathrm{i,j}}^{\mathrm{p,q}}(\xi,\eta)=N_{\mathrm{i,p}}(\xi)\cdot\left(\frac{d}{d\eta}M_{\mathrm{j,q}}(\eta)\right) \qquad (0.23)$$

Three dimensional shape functions can be obtained in a similar fashion.

$$\frac{\partial}{\partial\xi}R_{\mathrm{i,j,k}}^{\mathrm{p,q,r}}(\xi,\eta,\zeta)=\left(\frac{d}{d\xi}N_{\mathrm{i,p}}(\xi)\right)\cdot M_{\mathrm{j,q}}(\eta)\cdot L_{\mathrm{k,r}}(\zeta) \qquad (0.24)$$

$$\frac{\partial}{\partial\eta}R_{\mathrm{i,j,k}}^{\mathrm{p,q,r}}(\xi,\eta,\zeta)=N_{\mathrm{i,p}}(\xi)\cdot\left(\frac{d}{d\eta}M_{\mathrm{j,q}}(\eta)\right)\cdot L_{\mathrm{k,r}}(\zeta) \qquad (0.25)$$

$$\frac{\partial}{\partial\zeta}R_{\mathrm{i,j,k}}^{\mathrm{p,q,r}}(\xi,\eta,\zeta)=N_{\mathrm{i,p}}(\xi)\cdot M_{\mathrm{j,q}}(\eta)\cdot\left(\frac{d}{d\eta}L_{\mathrm{k,r}}(\zeta)\right) \qquad (0.26)$$

2.5. T-SPLine Geometry

2.5.1. T-SPLine Curves, Surfaces and Solids

Given a knot value vector Ξ and a polynomial degree p, we can evaluate the T-SPLine functions at every ξ. In order to create a T-SPLine curve, we also need a vector of coordinates for each basis function, the control points. $X_i = \{X_i \ , \ Y_i \ , \ Z_i\}$. After evaluating the shape functions, the T-SPLine surfaces defined in analogy to the T-SPLine curve:

$$C(\xi) = \sum_{\mathrm{i=1}}^{\mathrm{n}} \left\{ N_{\mathrm{i,p}}(\xi) \cdot X_{\mathrm{i}} \right\} \qquad (0.27)$$

Similarly, in three dimensional case once we determine the local knot value vectors, T-SPLine blending functions are obtained through:

$$S(\xi,\eta) = \sum_{\mathrm{i=1}}^{\mathrm{n}} \sum_{\mathrm{j=1}}^{\mathrm{m}} \left\{ N_{\mathrm{i,p}}(\xi) \cdot M_{\mathrm{j,q}}(\eta) \cdot X_{\mathrm{i,j}} \right\} = \sum_{\mathrm{i=1}}^{\mathrm{n}} \sum_{\mathrm{j=1}}^{\mathrm{m}} \left\{ R_{\mathrm{i,j}}^{\mathrm{p,q}}(\xi,\eta) \cdot X_{\mathrm{i,j}} \right\}$$

Thus, given a T-Mesh and an appropriate set of control points, we can define a three dimensional T-SPLine volume using:

$$S(\xi,\eta,\zeta) = \sum_{\mathrm{i=1}}^{\mathrm{n}} \sum_{\mathrm{j=1}}^{\mathrm{m}} \sum_{\mathrm{k=1}}^{\mathrm{l}} \left\{ N_{\mathrm{i,p}}(\xi) \cdot M_{\mathrm{j,q}}(\eta) \cdot L_{\mathrm{k,q}}(\zeta) \cdot X_{\mathrm{i,j,k}} \right\} = \sum_{\mathrm{i=1}}^{\mathrm{n}} \sum_{\mathrm{j=1}}^{\mathrm{m}} \sum_{\mathrm{k=1}}^{\mathrm{l}} \left\{ R_{\mathrm{i,j,k}}^{\mathrm{p,q,r}}(\xi,\eta,\zeta) \cdot X_{\mathrm{i,j,k}} \right\}$$

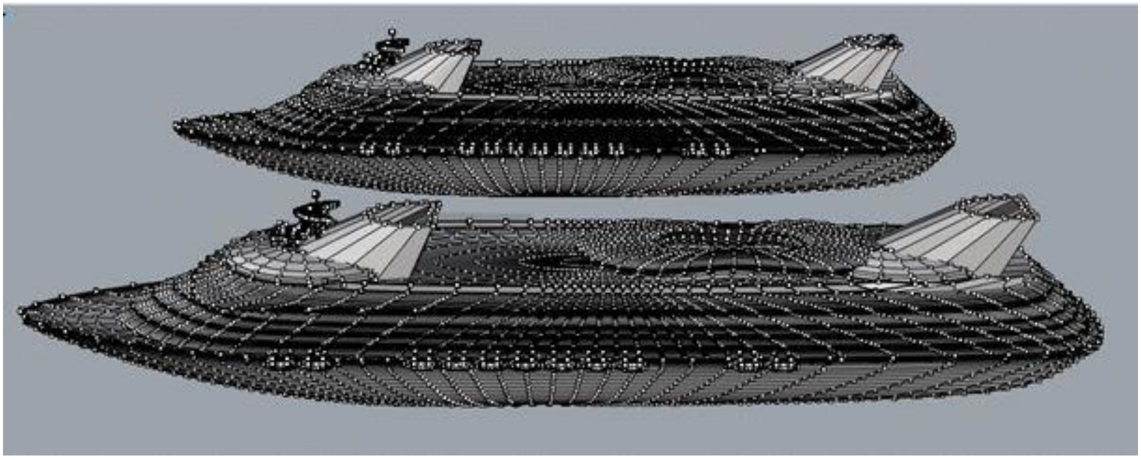


Figure 2.29.
T-SPLine Solid.

2.5.2. T-SPLine Curve Properties

As a generalization of NURBS, T-SPLine curves acquire most of their properties. The most important ones are:

1. T-SPLine curves can be considered as a generalization of Bezier curves with the difference that now a knot span is extended to a span.
2. C(ξ) is a piecewise polynomial curve.
3. Each blending function and by extension anchor corresponds to a control point.
4. Any control point corresponding to C^0-continuity is interpolatory to the curve.
5. T-SPLine curves maintain convex hull property.
6. Moving a control point X_i affects only part of the curves, due to its local support.
7. The control polygon represents a piecewise linear approximation to the curve.
8. Different control points are enabled to have same coordinates.
9. Affine covariance applies to T-SPLine curve’s control points.
10. Hence C(ξ) is a linear combination of univariate blending functions. Curve properties, like continuity and differentiability, stem straightforward out of them.
11. Two merged T-SPLines have a water-tight interconnection

2.5.2.1. Generalization of Bezier curves

T-SPLine curves are a generalization of B-SPLine curves and namely produced from a knot vector and the Cox de Boor recursive formula. Blending functions are linearly combined with control points and produce curves. This latter procedure is similar as far as Bezier curves are concerned. Bezier polynomials are produced given a knot value vector, that extends only in one knot span. So, they can be described as B-SPLines, that have support of only one span.

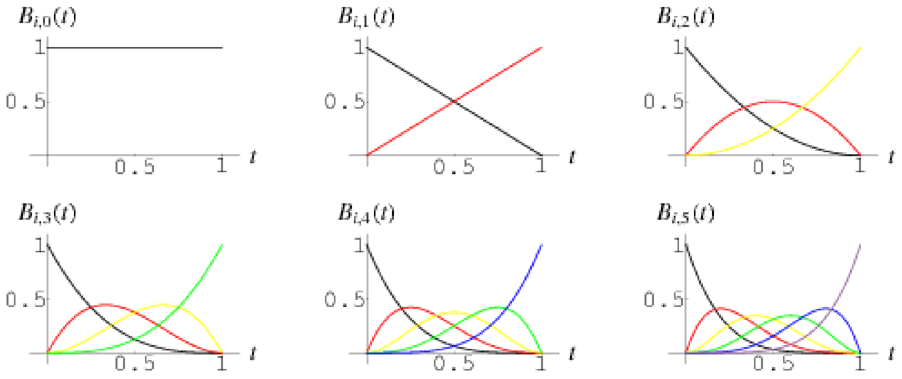


Figure 2.30.
Bezier Polynomials

2.5.2.2. Piecewise polynomial curve

C(ξ) is formed from piecewise polynomials $N_{i,p}(\xi)$ and therefore is a piecewise polynomial curve.

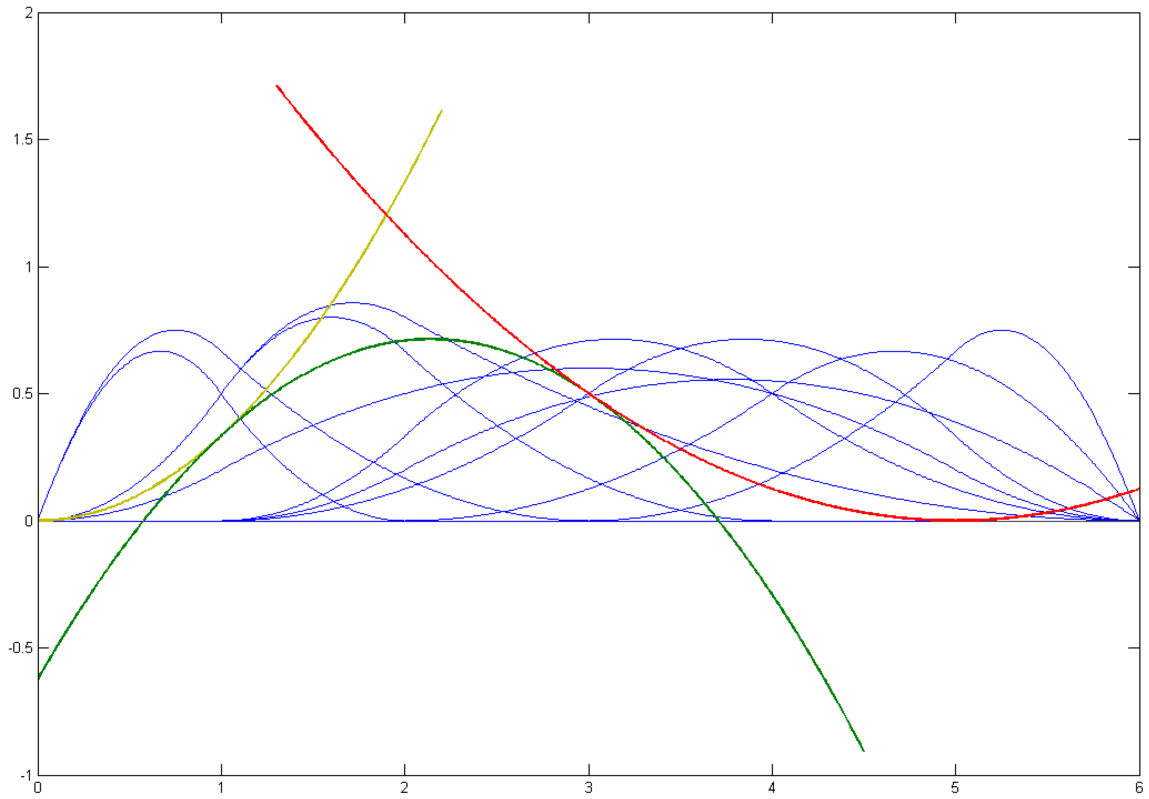


Figure 2.31. (Created with GeomIso)
Piecewise polynomials that form a T-SPLine blending function.

A T-SPLine curve is obtained through the curve function:

$$C(\xi) = \sum_{i=1}^{n} \left\{ N_{i,p}(\xi) \cdot X_i \right\}$$

T-SPLine curve is a linear combination of piecewise polynomial basis functions and control point coordinates. This applies in multi-directional entities as well. Two dimensional shapes derive from univariate piecewise polynomial functions Generalizing, T-SPLine surfaces (2D case, ξ, η) and T-SPLines solids (3D case, ξ, η, ζ) can be also considered piecewise.

2.5.2.3. Control Point – Basis Function Correspondence

Each blending function corresponds to a certain control point. There are n blending functions and n control points in a T-SPLine curve. In case of surfaces, there is a similar correspondence. Anchors (defined in index space) correspond to one local knot vector per direction. These local knot vectors are necessary to create the T-SPLine blending functions. Finally, in order to create a curve, each anchor must be linked with a control point. A surface stems from linear combination of control points and blending functions. In the figure below, we can see the one to one correspondence of anchors-control points and blending functions.

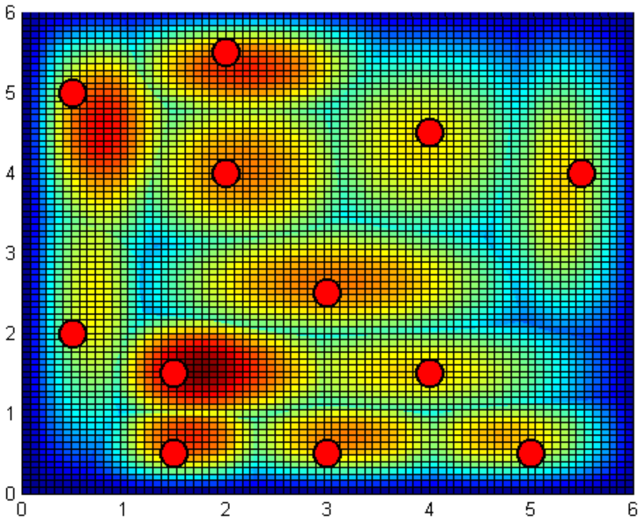


Figure 2.32. (Created with GeomIso)
Blending function to anchor correspondence.

It is obvious that anchors do not correspond to the peak of the blending functions. This is why anchors' local support on one or both directions has multiple knot values on its edges.

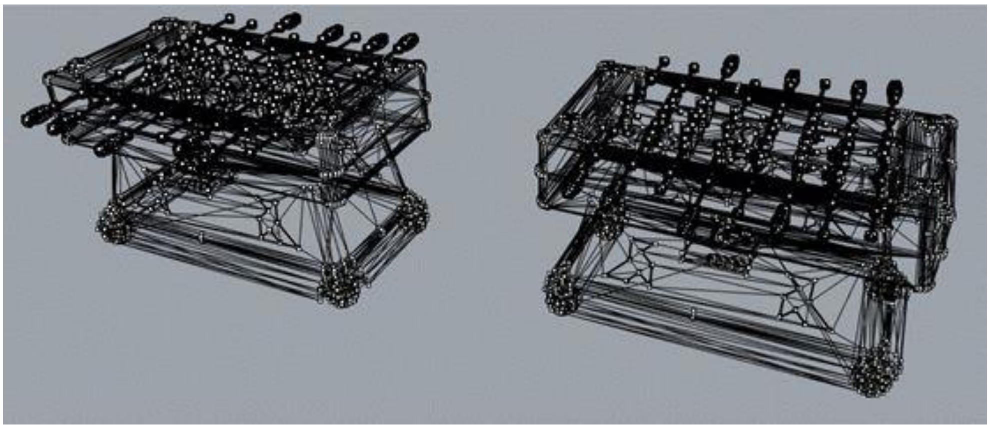


Figure 2.33. Physical Space.

2.5.2.4. Interpolation to the curve

Any control point corresponding to a blending function with C^0 continuity is interpolatory to the curve.

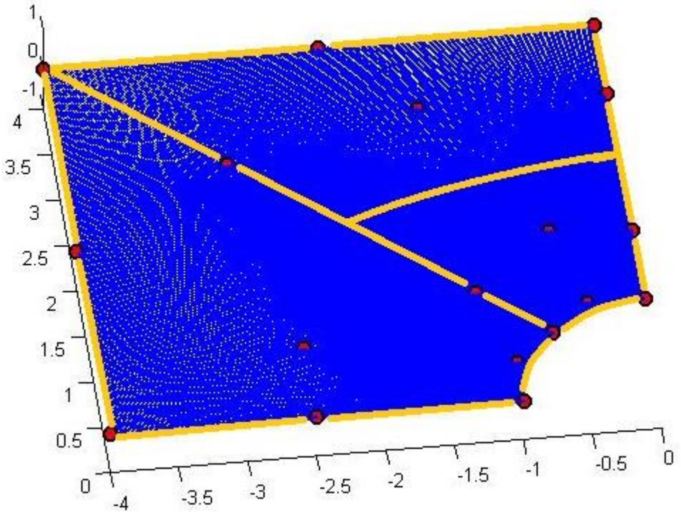


Figure 2.34. (Created with GeomIso)
Physical space with control points. Plate with a hole.

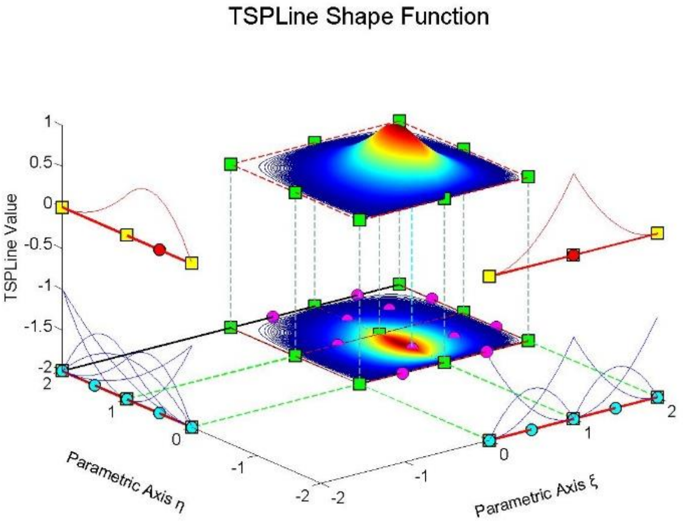


Figure 2.35. (Created with GeomIso)
Physical space's blending functions. 8th control point.

Figure 2.35 shows that control points with blending functions of C^0 Continuity are interpolatory to the curve. This can be explained with the help of the following curve equation.

$$C(\xi) = \sum_{i=1}^{n} \left\{ N_{i,p}(\xi) \cdot X_i \right\}$$

For $\xi = 0$, it applies that:

$$C(0) = \sum_{i=1}^{n} N_{i,2}(0) \cdot X_i$$

where

$$N_{1,2}(0) = 1$$

$$N_{i,2}(0) = 0, \; i = 2,...,6$$

so,

$$C(0) = N_{1,2}(0) \cdot X_1 = X_1$$

And for $\xi = 3$:

$$C(3) = \sum_{i=1}^{n} N_{i,2}(3) \cdot X_i$$

$$N_{6,2}(3) = 1$$

$$N_{i,2}(3) = 0, \; i = 1,...,5$$

so,

$$C(3) = N_{6,2}(3) \cdot X_6 = X_6$$

Likewise, the internal control point (with C^0 continuity across $\xi = 2$) is interpolatory to the curve because:

$$C(2) = \sum_{i=1}^{n} \left\{ N_{i,2}(2) \cdot X_i \right\}$$

$N_{4,2}(2) = 1$, as this is the only non-zero basis function across $\xi = 2$.

$$N_{i,2}(2) = 0, \; i \neq 4$$

so,

$$C(2) = N_{4,2}(2) \cdot X_4 = X_4$$

Observe that both the form of the curve and the form of the basis functions indicate that this geometry could be represented by two different sets of knot vectors and control points, with absolutely no deflections from the current representation. This will be examined thoroughly later. Interpolation also applies for surfaces and solids, when appropriately reduced continuity is used for all directions at a knot. C^{-1} continuity is required for external knots and C^0 for internal.

2.5.2.5. Convex Hull

T-SPLines curves, as a generalization of NURBS curves, retain convex hull property. The convex hull of the curve is obtained as the sum of the convex hull of p+1 consecutive control points. The curve is always contained in the convex hull.

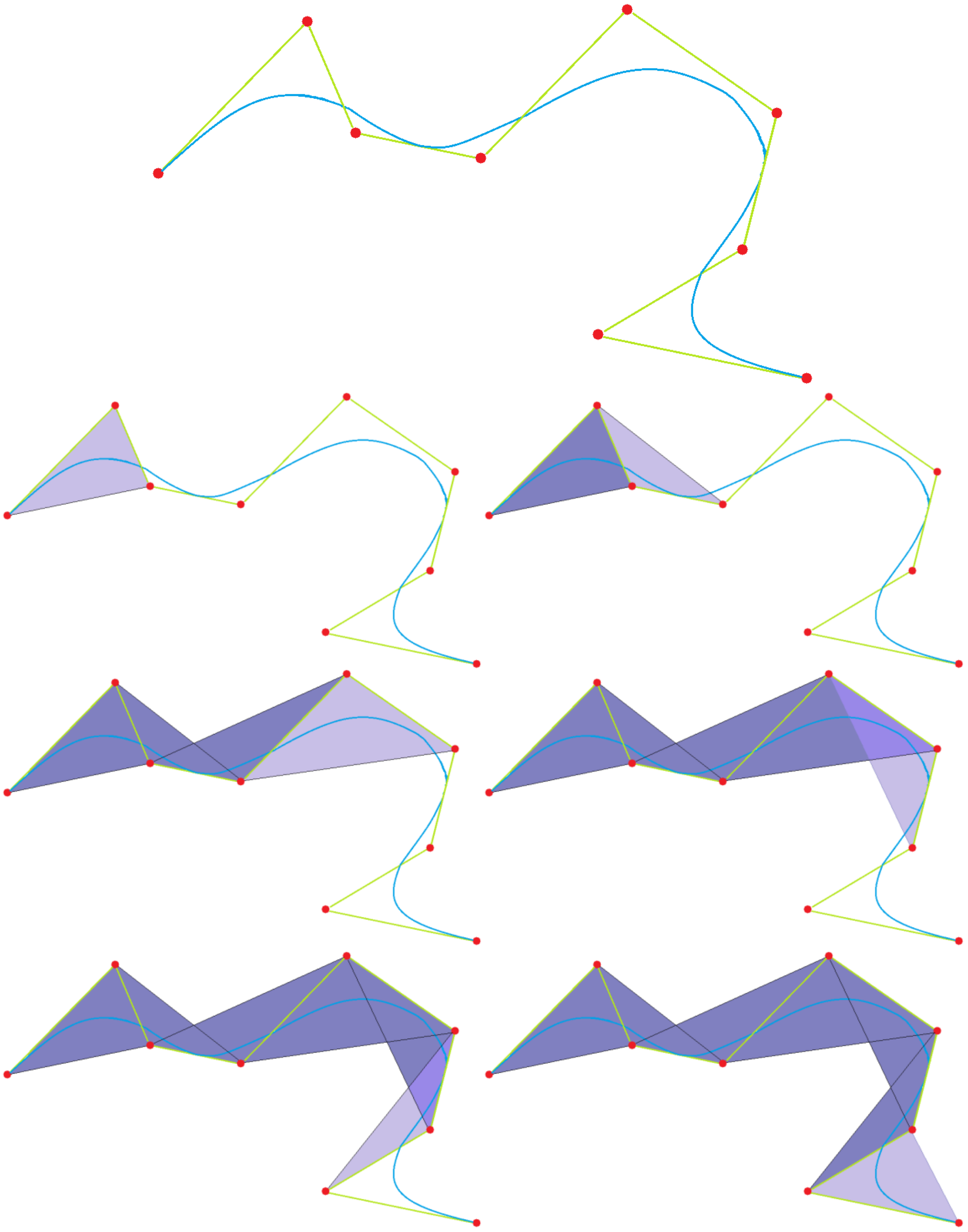
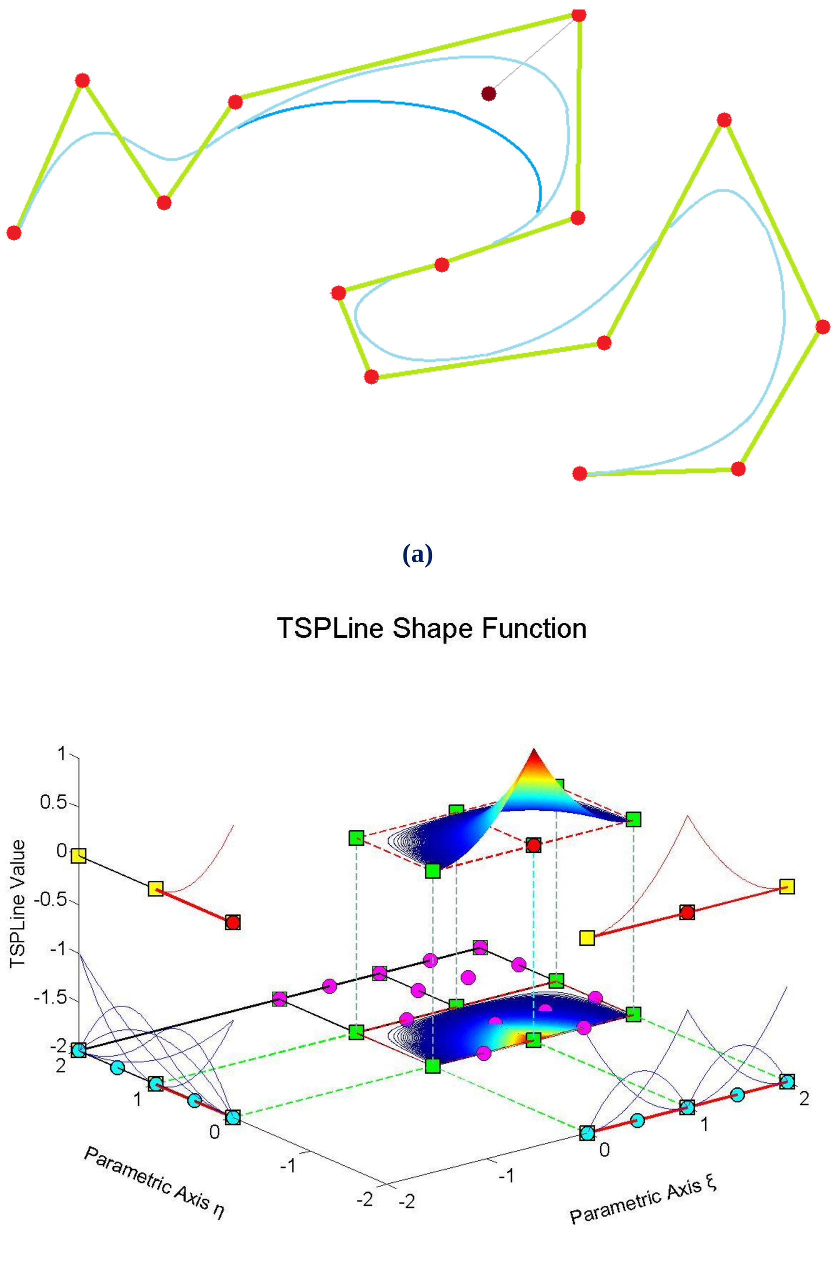


Figure 2.36.
Convex Hull Representation

2.5.2.6. Control Point Local Support

Moving a control point X_i only changes part of the curve, more specifically the part corresponding to one control point is p+2 consecutive spans per axis. This is a result to local character of the corresponding T-SPLine blending functions. Note that in T-SPLines, blending functions do not affect a certain number of knot value spans rather a certain number of spans, as the restricted non tensor product T-Mesh differentiates the span and knot span definitions.



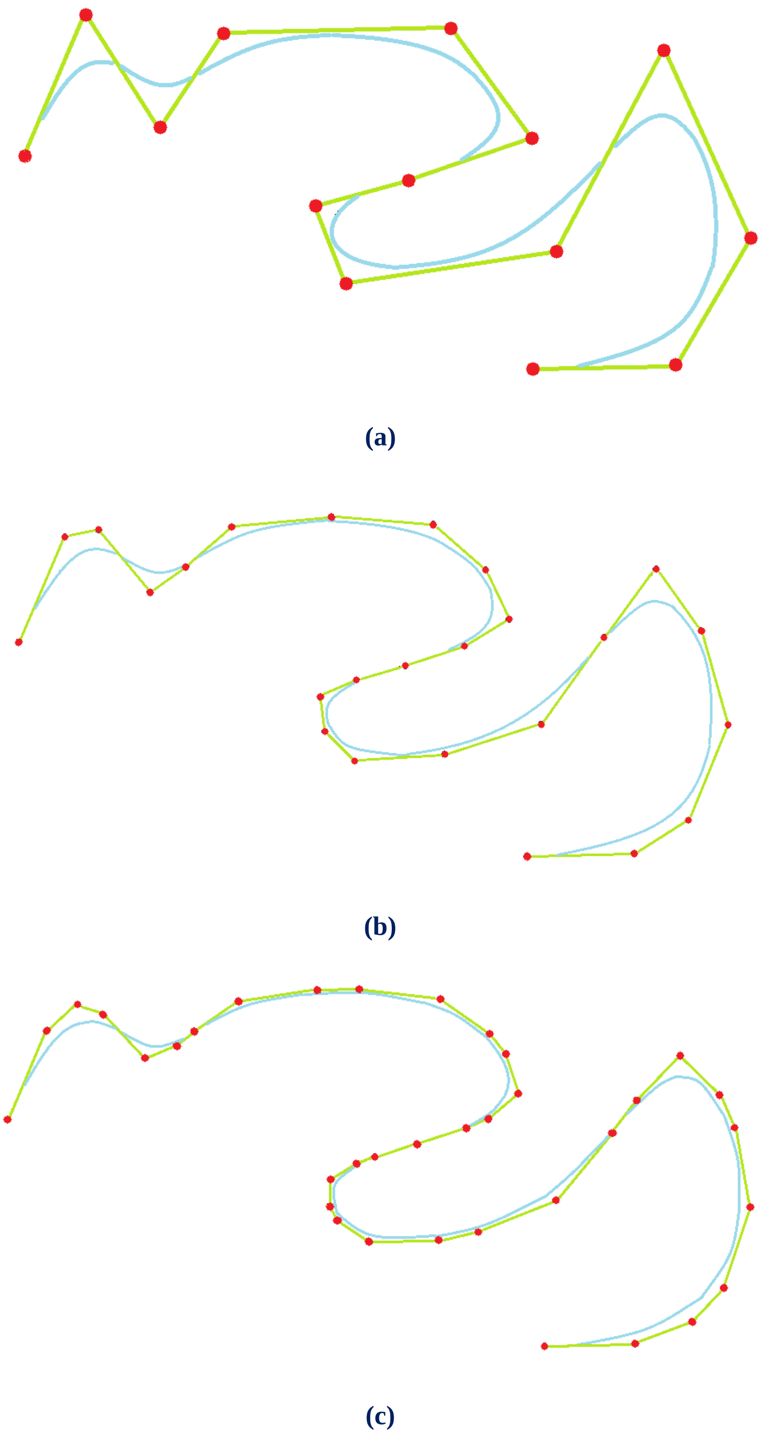
(a)

(b)

Figure 2.37. (Created with GeomIso)
Control point local support.

2.5.2.7. Control Polygon Approximation

The control polygon represents a piecewise linear approximation to the curve. Due to convex hull properties, refinement brings the control polygon closer to the curve.



(a)

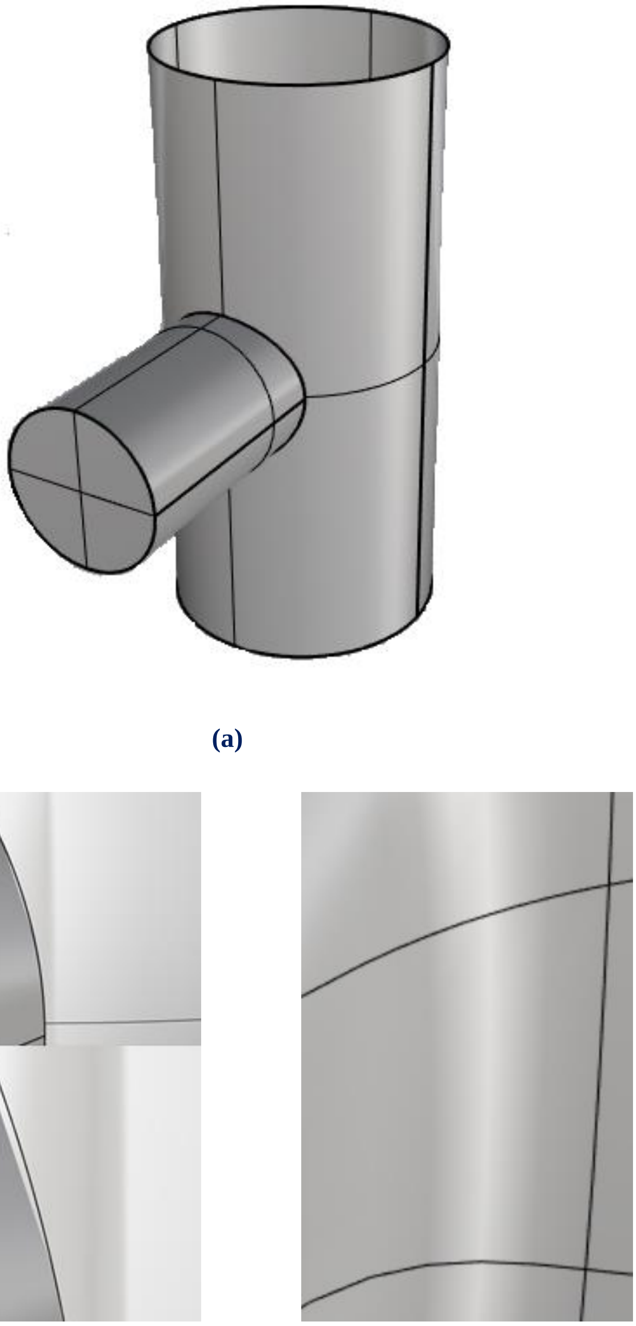
(b)

(c)

Figure 2.38.
Control polygon approximation.

2.5.2.8. Two merged T-SPLines have a water-tight interconnection

The intricate structure of the mesh enables the designer to merge separate meshes with gap free “welding”.



(a)

(b)

(c)

Figure 2.39.
(a) Two pipes and their interconnection.
(b) Non watertight interconnection for NURBS representation.
(c) Watertight interconnection for T-SPLines design.

2.6. Refinement

Isogeometric analysis is a methodology that allows designers to utilize the exact mesh of the geometry. In addition, engineers are able to analyze the model more efficiently and accurately, as the shape functions used for design are the same with the ones used for analysis. Unfortunately, the coarse mesh used by designers does not usually give satisfactory results. So, refinement is introduced in order to improve the solution, while keeping the parameterization and geometry unchanged. Various types of refinement exist, which allow scientists to reach the desired outcome with various ways (each one suitable for specific problems). Note that NURBS refinement strategies require that full rows or columns of control points are inserted in order to refine the needed area. On the other hand, T-SPLines are the first type of SPLines that enable local refinement.

2.6.1. Refinement Types

2.6.1.1. Knot Value Insertion

Knot value insertion is the introduction of new knot values and the enrichment of the existing knot value vector Ξ. A new knot value vector $\overline{\Xi}$ is created, such that$\Xi \subset \overline{\Xi}$. Only internal knot values can be added; while the ones on the boundaries have to remain intact.

2.6.1.2. Degree Elevation

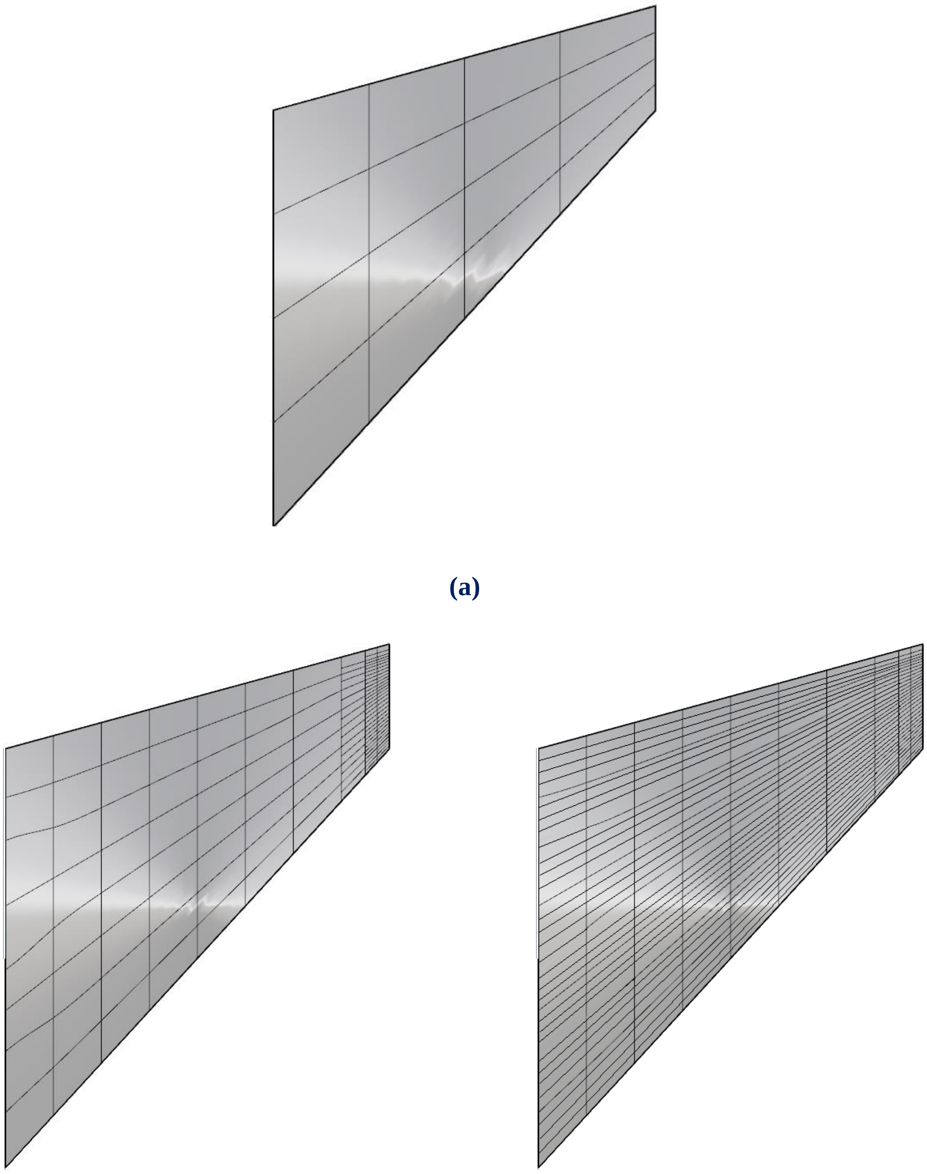
Instead of adding knot values to an existing knot value vector, the raise of the polynomial order can also enrich the basis. Apart from geometry, the mapping from parameter to physical space must also remain unchanged.

2.6.1.3. Degree Elevation and Knot Value Insertion

Increasing polynomial degree by p-refinement is an improvement to the basis, but continuity remains the same as in the coarse mesh. In order to improve this aspect, k-refinement was introduced by Hughes. The basic idea is that, after p-refinement, h-refinement can be applied in order to create basis functions of C^{p-1} Continuity. This is a powerful tool that can lead to greater convergence rates for our models.

2.6.2. Local refinement

T-SPLines are the first type of SPLines that enables true local refinement. The insertion of T-junctions enables refinement to take place on element level. Faces subdivision is the most common refinement strategy. The main process is to divide one of the elements created into four new ones. In addition, T-SPLines allow the insertion of single control points in places, where geometry needs special local handling. Finally, there is an edge insertion possibility that allows the division of element into two and at any point needed by analysis.



(a)

(b) **(c)**

Figure 2.40.
(a) Initial Mesh.
(b) Refinement using T-SPLines.
(c) Refinement using NURBS.

3. Stiffness Matrix

3.1. Preliminary Steps

3.1.1. Degrees of Freedom

In isogeometric analysis, as it is common in any other type of computational analysis, it is crucial to define the modeling type of structure that is going to be analyzed. Given the fact that until now only continuum mechanics elements are applied in isogeometric analysis, the previous dilemma could be transformed into the number of dimensions the simulation elements will have, i.e. the number of displacement degrees of freedom. So, in order to define the analysis type, the first step would be to determine the number of dimensions of the simulation. Three categories are available:

- One displacement degree of freedom (**1D analysis**). This type appeals to problems like truss elements analysis. In these cases, forces and displacements are applied only on the longitude of the structure. Lines and curves are structures that belong to this case, as they can be considered the representation of a given element that its height and thickness are substantially smaller (by orders of magnitude) than its length.

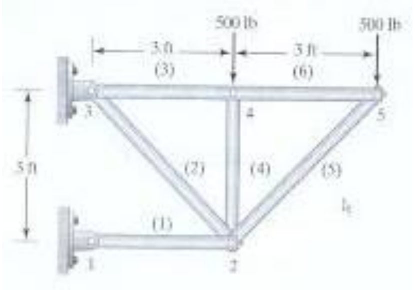


Figure 3.1. 1D analysis structure.
(engineering-inventions.blogspot.com)

- Two displacement degrees of freedom (**2D analysis**), which is suitable for surfaces and plane strain or plane stress structures. Plane strain is considered a structure that its thickness is significantly larger than length and width by orders of magnitude. In that case, we can simplify the real structure by considering a surface that has a certain usually uniform thickness. Plane stress is considered a structure that its thickness is significantly smaller than length and width by order of magnitude. In that case, calculation of stresses is simplified, since they correspond to a 2x2 matrix, rather than 3x3.

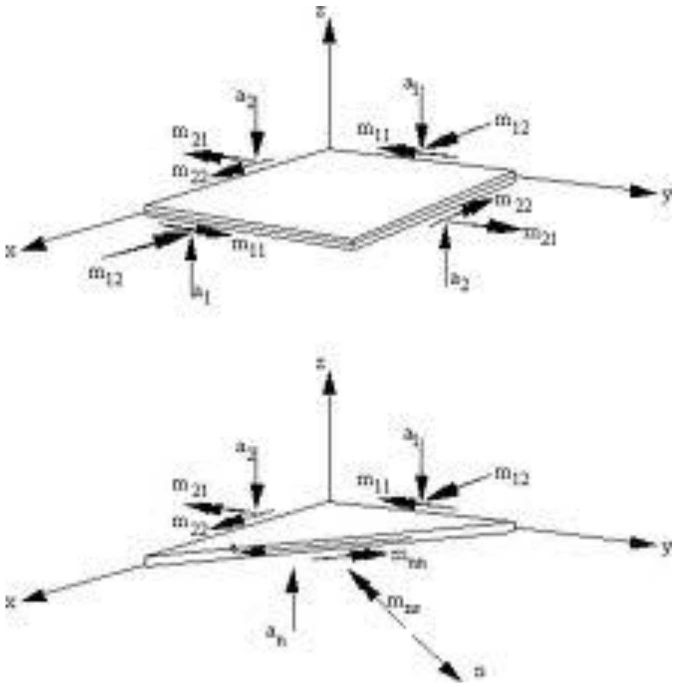


Figure 3.2. 2D analysis structure.
(www.princeton.edu)

- Three displacement degrees of freedom (**3D analysis**). By this approach, any structure or object is simulated in real dimensions, without considering simplifications based on its physical analogy.

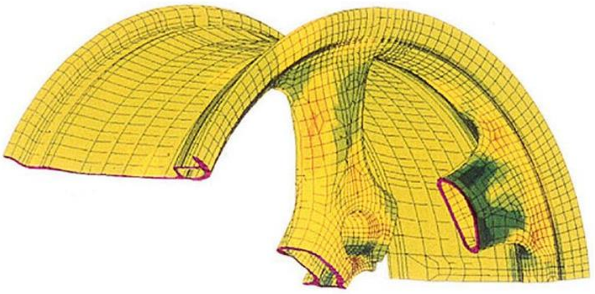


Figure 3.3. 3D analysis structure.
(www.hondatuningmagazine.com)

The analysis type determines the information about the structure's geometry, that should be provided. In 1D analysis, there exists only one parametric axis (ξ) for the representation of the model and therefore data of height and thickness are required. In 2D analysis, one-dimensional control points are given on axes ξ and η and the unknown data is the thickness of the structure. In 3D analysis, the geometry is fully described by the combination of control points on axes ξ, η and ζ, requiring no further information.

3.1.2. Constitutive Law

In order to achieve an analysis that corresponds to the actual behavior of the used material, its properties and constitutional law have to be defined. Properties deal with elasticity modulus, Poisson's ratio and specific weight. The constitutive law defines the response of the material under the action of external forces, depending on whether it is elastic-inelastic, isotropic-anisotropic, homogenous- no homogenous and includes equations which describe this reaction. Elastic materials tend to recover completely from deformation and return to their initial shape after the external load is removed. Steel and iron can usually be simulated as materials with linear elastic behavior, when loads increment. On the other hand, materials, like concrete, have only a short linear elastic behavior, followed by a brittle almost plastic breach. Isotropic materials have identical properties in all directions and the homogenous ones have identical properties at all points in their body.

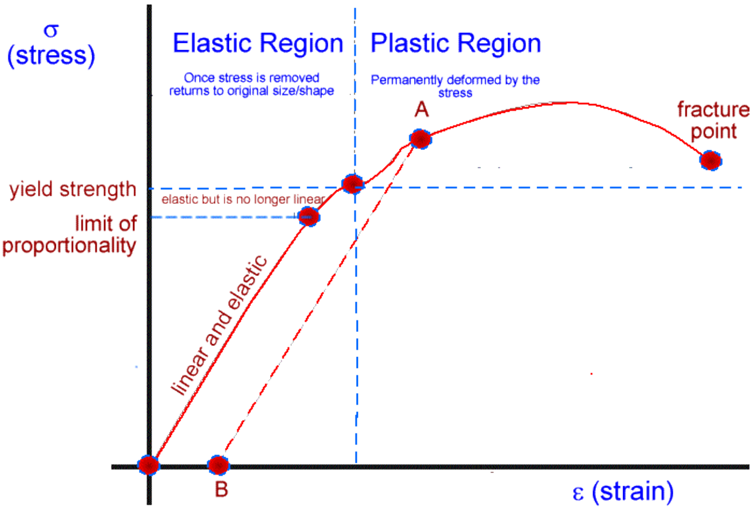


Figure 3.4. Stress-strain chart. Elastic and plastic regions. (www.cyberphysics.co.uk)

So, the constitutive law of the material plays a major role on the analysis type that will be performed. Many types of analysis can be applied to a structure, varying from linear (stress proportional to strain) to nonlinear and static to dynamic. Linear and nonlinear analyses are types imposed by material properties, as explained above. The choice between static and dynamic analysis depends on the acceleration of the applied load in comparison to the natural frequency of the structure. When the load is applied slowly enough, static analysis can be performed. Otherwise, inertia forces should be taken into consideration and the suitable analysis type is the dynamic one. For the needs of this thesis, the selected structures will undergo a linear static analysis and both elasticity modulus and Poisson's ratio must be defined for the selected material.

3.1.3. Elasticity Matrix

The elasticity matrix type is given by the dimension and the type of the simulation used in each case. As mentioned above, the degrees of freedom for each problem may vary between three different cases. Elasticity matrices for these cases are shown below.

1D Elasticity:

$$\underset{(1x1)}{[E]} = E$$

2D Elasticity, Plane Stress:

$$\underset{(3x3)}{[E]} = \frac{E}{1-\nu^2} \cdot \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix}$$

2D Elasticity, Plane Strain:

$$\underset{(3x3)}{[E]} = \frac{E}{(1-\nu)\cdot(1-2\nu)} \cdot \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix}$$

Finally, in three dimensional analysis, elasticity matrix is identical to the one of finite element analysis, as it is shown below:

$$\underset{(6x6)}{[E]} = \frac{E}{(1-\nu)\cdot(1-2\nu)} \cdot \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix}$$

The corresponding stress and strain vectors are:

1D Elasticity:

$$\underset{(1\times1)}{\{\sigma\}} = \{\sigma_X\}$$

$$\underset{(1\times1)}{\{\varepsilon\}} = \{\varepsilon_X\}$$

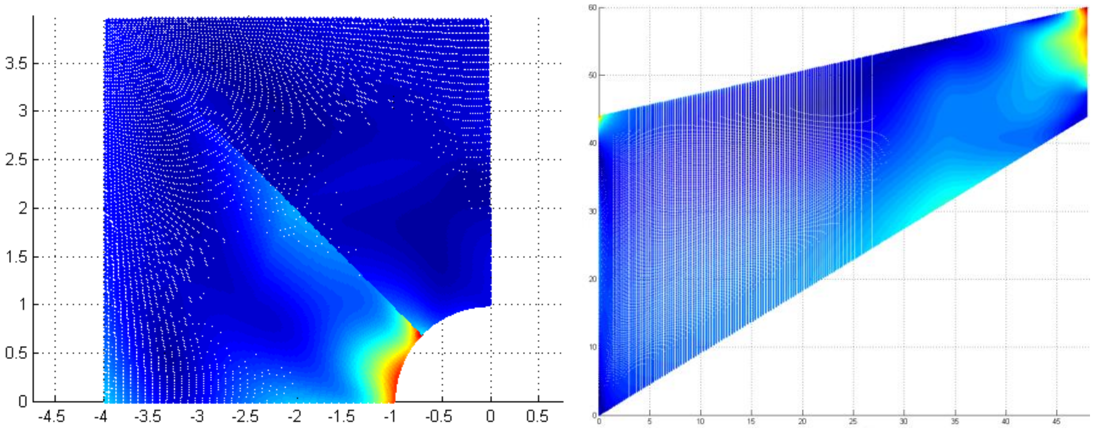
2D Elasticity, Plane Stress and Plane Strain:

$$\underset{(3\times1)}{\{\sigma\}} = \begin{Bmatrix} \sigma_X \\ \sigma_Y \\ \tau_{XY} \end{Bmatrix}$$

$$\underset{(3\times1)}{\{\varepsilon\}} = \begin{Bmatrix} \varepsilon_X \\ \varepsilon_Y \\ \gamma_{XY} \end{Bmatrix}$$

3D Elasticity:

$$\underset{(6\times1)}{\{\sigma\}} = \begin{bmatrix} \sigma_X \\ \sigma_Y \\ \sigma_Z \\ \sigma_{XY} \\ \sigma_{YZ} \\ \sigma_{ZX} \end{bmatrix}, \quad \underset{(6\times1)}{\{\varepsilon\}} = \begin{bmatrix} \varepsilon_X \\ \varepsilon_Y \\ \varepsilon_Z \\ \gamma_{XY} \\ \gamma_{YZ} \\ \gamma_{ZX} \end{bmatrix}$$



(a) **(b)**

Figure 3.5. Stress contour. (Created with GeomIso)
Subfigure **(a)** shows stresses that are discontinuous due to knot multiplicity (C^0 continuity).
In figure **(b),** continuity is greater than C^0 and stresses are smoother.

3.1.4. Mesh

Since isogeometric analysis is a computational analysis method, its main concern is to approximate as accurately as possible the desired geometry. Mesh is the array of isogeometric elements used to represent the geometry of a structure. FEM node is now divided into two different notions, the notion of knot and the notion of control point, which have complementary properties. Knots are the boundaries of isogeometric elements, while control points determine the degrees of freedom. The number of basis functions used is equal to the number of control points. The unknowns of the resulting algebraic equations correspond to the displacements of the control points for each axis.

In IGA method, the exact geometry is represented (even in cases of very coarse meshes), unlike FEM, where approximations can only be achieved. Thus, the fact that the same shape functions are used for both analysis and geometry (solution field) assures accurate and reliable results. Better quality of a NURBS-mesh is also pursued through multiple patches, which are subdomains with the varying material and geometry type and consist of a full tensor product grid of elements.

The basic feature of IGA basis functions is their tensor product nature. Functions of each parametric axis are combined in order to form the corresponding shape functions. Due to the higher regularity between inter-element boundaries, they exhibit greater overlapping in comparison with the shape functions of FEA. Attention is recommended at knots where basis functions are interpolatory (connection of patches), since that indicates discontinuity (C^{-1}) and consequently loss in accuracy.

The order of curves that will verge on geometry should be defined appropriately. For instance, if a straight line needs to be modeled, a minimum degree of one must be given to the polynomial which is called upon to fit it. So, every dimension of the problem must acquire a certain degree. Due to the isoparametric concept applied in IGA, these degrees are usually applied to parametric space rather than to the dimensions of physical space. It is wise to always apply a degree greater than the minimum one in order to improve the accuracy of the integration scheme.

In case of B-SPLines, each object is divided into orthogonal-like pieces and then each one of them is divided into elements that are utilized for integration. This division requires only the definition of global knot value vector.

On the other hand, TSPLines, as a generalization of B-SPLines and NURBS, have a much more complex mesh definition, which enables easy design and local refinement, but, on the other hand, increases analysis complexity. As far as T-SPLines are concerned, there is also a distinction between T-Mesh and integration mesh that will also be explained. These two major aspects of T-SPLines will be thoroughly analyzed in the following paragraphs.

3.1.5. T-Mesh - Integration Mesh

As mentioned in Chapter 2, integration mesh is different from a T-Mesh. A T-Mesh might have a variety of junction combinations, making it almost impossible to define the elements where integration will take place. The integration mesh is a product of the initial mesh and the supports of anchors are depicted in the figure below. Figure 3.6 depicts two anchors of the T-Mesh and how their local knot value vectors are defined.

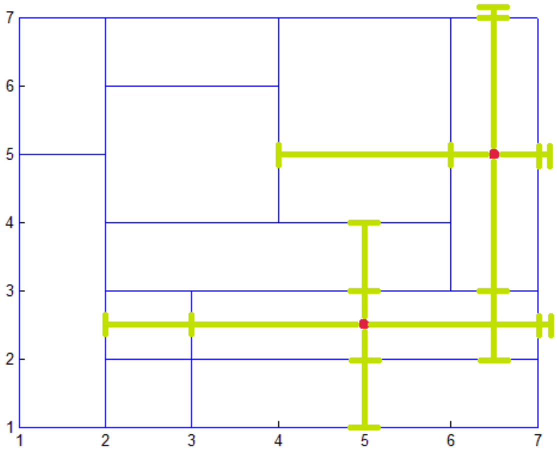


Figure 3.6. T-Mesh. (Created with GeomIso)
Anchors are shown as red dots and their supports as green lines.

For the anchor, that has index coordinates {ξ_α, η_α}={5,2.5}, its knot value vector consists of p+2 values for both axes. This support per direction creates an influence domain for each anchor, as shown below.

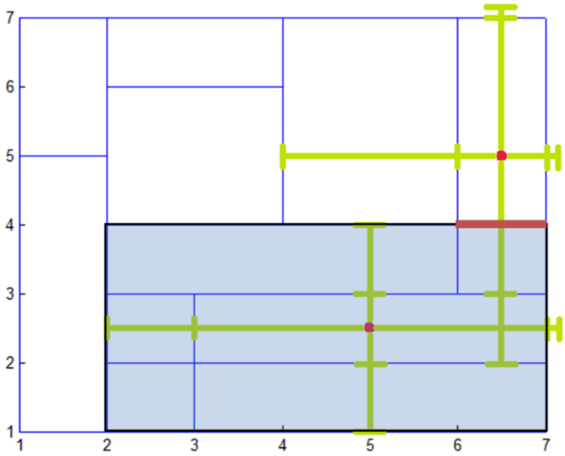


Figure 3.7. (Created with GeomIso)
T-Mesh. Influence domain of anchor with index coordinates {ξ_α, η_α}={5,2.5}.
Red bold line is a continuity reduction line, as created by this anchor's support.

The support domain of this function is defined by the blue transparent area. It is obvious that the defined support area creates lines that do not exist in the real mesh. One of these lines is depicted above with a deep red color. By creating all these non-existing lines in a T-Mesh, we take as a product the so called continuity reduction lines. These lines divide the index space into elements, where shape functions are C^{∞} continuous. On the verge of these lines, the continuity is limited due to the connection of different polynomials.

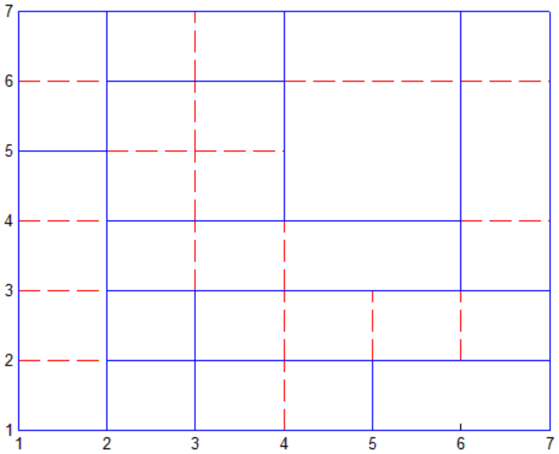


Figure 3.8. (Created with GeomIso)
T-Mesh. Continuity reduction lines as red dashed ones.

These continuity reduction lines form the final integration mesh, on which integration will be performed after the placement of Gauss Points.

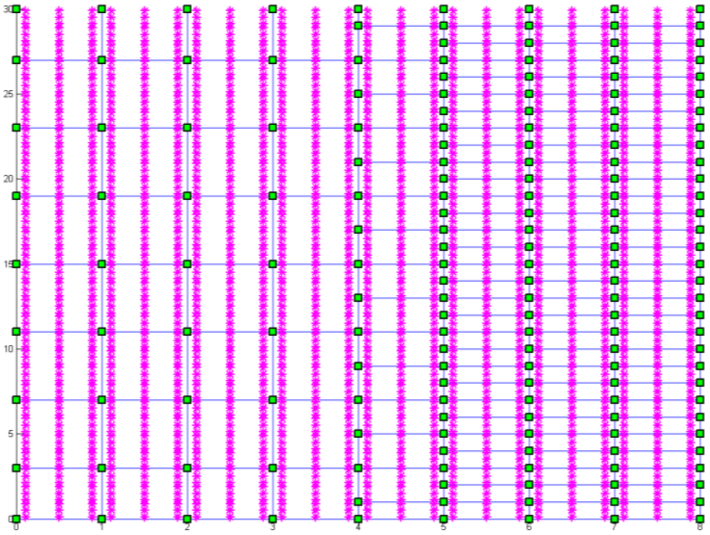


Figure 3.9. (Created with GeomIso)
Parameter space with Gauss points for integration elements.

3.1.6. Gauss Points

Gauss points should be placed in the elements defined by the continuity reduction lines as above. Since blending functions, as described in Chapter 2, are piecewise polynomials in each span, Gauss points are chosen to be applied in each and every knot span. Gauss points for the mesh above are shown in Figure 3.10.

As it was previously mentioned, Gauss points are chosen per parametric direction for each element. Their coordinates are obtained on a reference element spanning [-1, 1] as the roots of the Legendre polynomials. The next step is to transform the coordinates and weights from the reference knot span ξ^R to the desired knot span $[\xi_i, \xi_{i+1})$.

$$\xi = \frac{(\xi_{i+1} - \xi_i) \cdot \xi^R + (\xi_{i+1} + \xi_i)}{2}$$

$$w^{GP\xi} = \frac{(\xi_{i+1} - \xi_i)}{2} \cdot w_\xi^R$$

Full tensor product properties apply in the analyzed elements, leading in similar equations for the other two parametric directions.

$$\eta = \frac{(\eta_{j+1} - \eta_j) \cdot \eta^R + (\eta_{j+1} + \eta_j)}{2}$$

$$w^{GP\eta} = \frac{(\eta_{j+1} - \eta_j)}{2} \cdot w_\eta^R$$

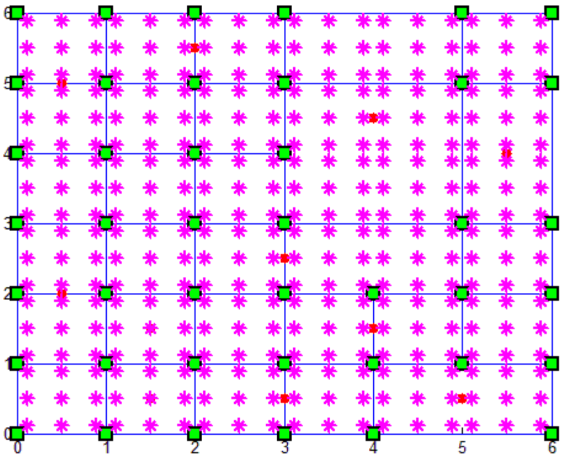


Figure 3.10. T-Mesh. (Created with GeomIso)
Knots are shown as green squares.
Gauss points (magenta stars) are placed with respect to tensor product knot spans.

The minimum number of Gauss points, where a certain function has to be evaluated, is p+1, where p is equal to the degree of the polynomial, as previously described. This statement leads us to the conclusion that different number of integration (Gauss) points has to be defined and applied to each and every different axis.

Gauss integration is preferred, because its ability to have integration points at varied distances makes integration much more accurate than other schemes. Standard positions and weights of Gauss points for an interval [-1,1] are the following:

n	ξi	αi
1	0	2
2	-0,57735	1
	0,57735	1
3	-0,77459	0,55555
	0	0,88888
	0,77459	0,55555
4	-0,86113	0,34785
	-0,33998	0,65214
	0,33998	0,65214
	0,86113	0,34785
5	-0,90617	0,23692
	-0,53846	0,47862
	0	0,56888
	0,53846	0,47862
	0,90617	0,23692

Table 3.1.
Positions and weights of Gauss points for the interval [-1,1].

3.1.6.1. Gauss Point Number

Gauss point's coordinates and weights are evaluated for every element created by a tensor product knot span. According to Hughes et al, for the exact integration of a polynomial of degree q, $\frac{q+1}{2}$ or $\frac{q+2}{2}$ Gauss points are required per knot span in the case of q odd and even respectively.

For 1D problems, the maximum degree of the deformation matrix is defined from the derivation of piecewise polynomial shape functions of polynomial degree q and is consequently q-1.

$\left[B(\xi)\right]^{T}\cdot[E]\cdot\left[B(\xi)\right]$ yields the product of polynomials of maximum order p-1 resulting in a polynomial of maximum order $(p-1)+(p-1)=2p-2$. Thus, the minimum number of gauss points per knot span required for exact integration is $\frac{(2p-2)+2}{2}=p$.

Inductively, for 2D and 3D problems, the exact order of deformation matrix is determined by partial derivation of piecewise polynomial shape functions. Therefore, the maximum degree is p for derivation in the remaining directions.

The order of the product $\left[B(\xi)\right]^{T}\cdot[E]\cdot\left[B(\xi)\right]$ is $p+p=2p$. In order to achieve exact integration, the minimum number of Gauss points required per knot span is $\frac{2p+2}{2}=p+1$.

To sum up:

- For 1D problems, $\mathbf{p}$ Gauss points per knot span are required.
- For 2D and 3D problems, $\mathbf{p+1}$ Gauss points per knot span are required.

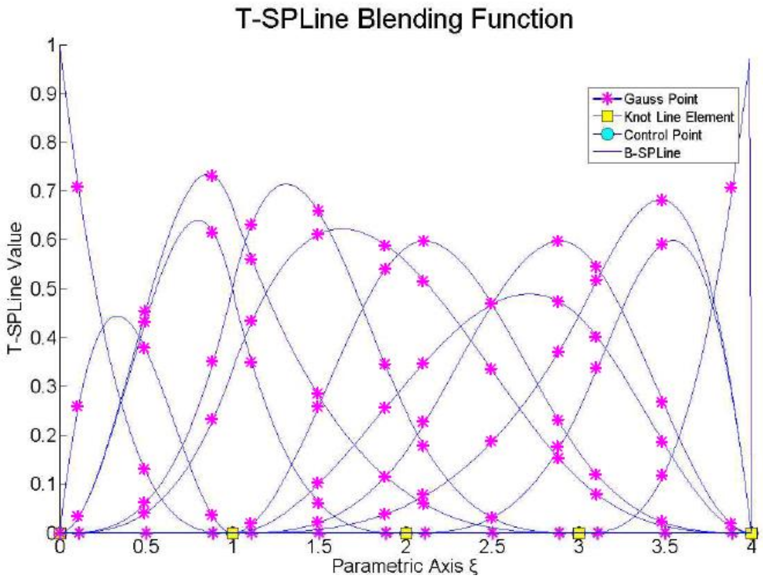


Figure 3.11. (Created with GeomIso)
Quadratic T-SPLine blending functions.
p+1 or three Gauss points are required per knot span for accurate integration.

3.2. Patch Merging

In computation science, it is of crucial importance to divide a structure into simple segments, that can be separately analyzed and then all combined can give the final solution. This idea became the motive that urged the development of patches. Patches are used in case of specific elements that have different material properties and thus do not share the same constitutive law. Unfortunately, this option is not available as far as T-SPLines are concerned. T-SPLines do not support the so called patch, like NURBS do. T-Meshes cannot be torn apart and then combined due to their intricate element interaction. This might seem a major drawback of T-SPLines, but once again T-Mesh's intricate interconnection allows the watertight merging of NURBS patches. Specifically, a uniform mesh can be created for one or more patches with different knot value vectors. This is a solution to the previously mentioned problem.

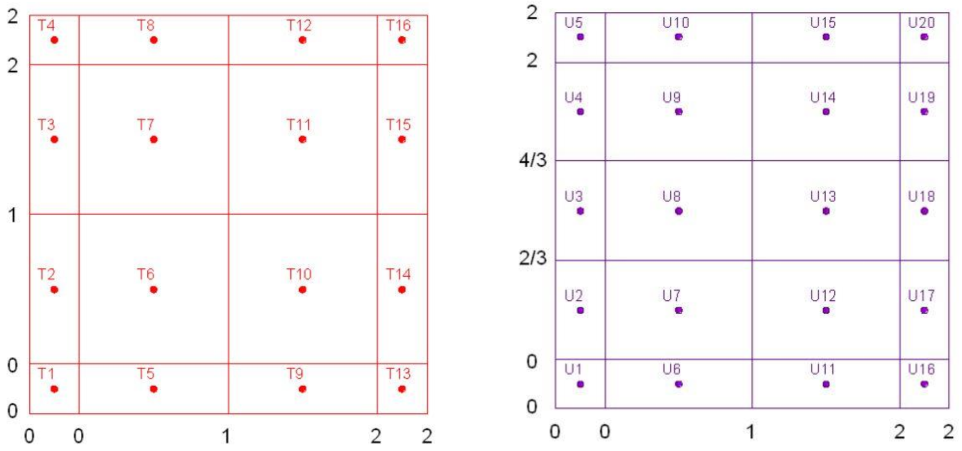


Figure 3.12. Quadratic TSPLines.
Parameter knot value spaces with different knot value vectors on axis η will be merged.

Figure 3.12 shows two separate NURBS patches. Apparently, these patches are created by orthogonal domains that have different global knot value vectors. Until now, the connection of these patches with NURBS was impossible. Refinement strategies, such as knot insertion, had to be implied in order to create domains whose control points would coincide. This obstacle can be easily overcome with T-SPLines. With the appropriate manipulation, only the boundary control points and elements of both patches need to be adjusted so as to coincide. In Figure 3.12, left patch has four elements on the connection edge, while the right one has five. It would be expected that the connected boundaries should both have five elements (the maximum number of boundary elements). The fact that the knot value vectors do not have the same values leads to the creation of a new knot value vector, which is the Boolean union of the previous ones. This procedure is depicted in Figure 3.13, where the two domains are prepared for a C^0 continuity connection. Since the two knot value vectors are now identical, the patch interconnection becomes now possible, as shown in Figure 3.14.

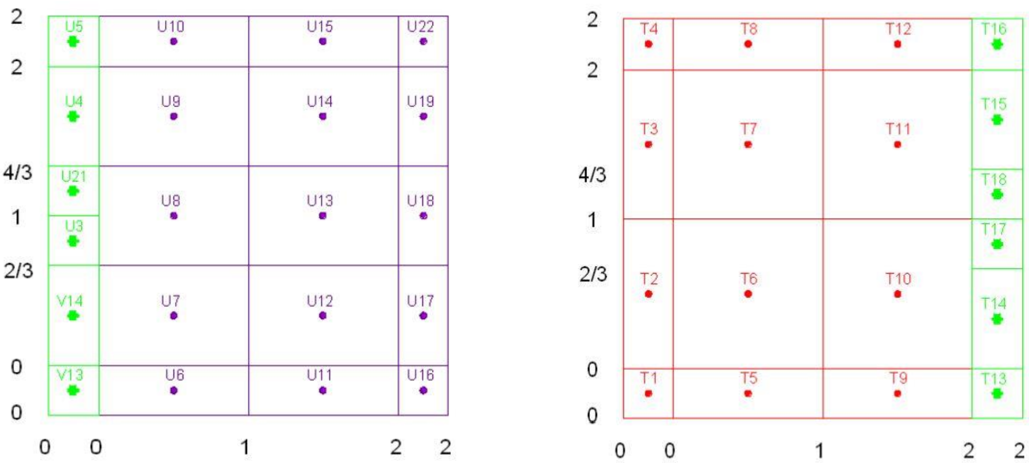


Figure 3.13.
Quadratic T-SPLines.
Parameter knot value spaces are prepared for integration.
The final column of the first mesh and the initial of the second are divided,
in order to anchors coincide after merging.

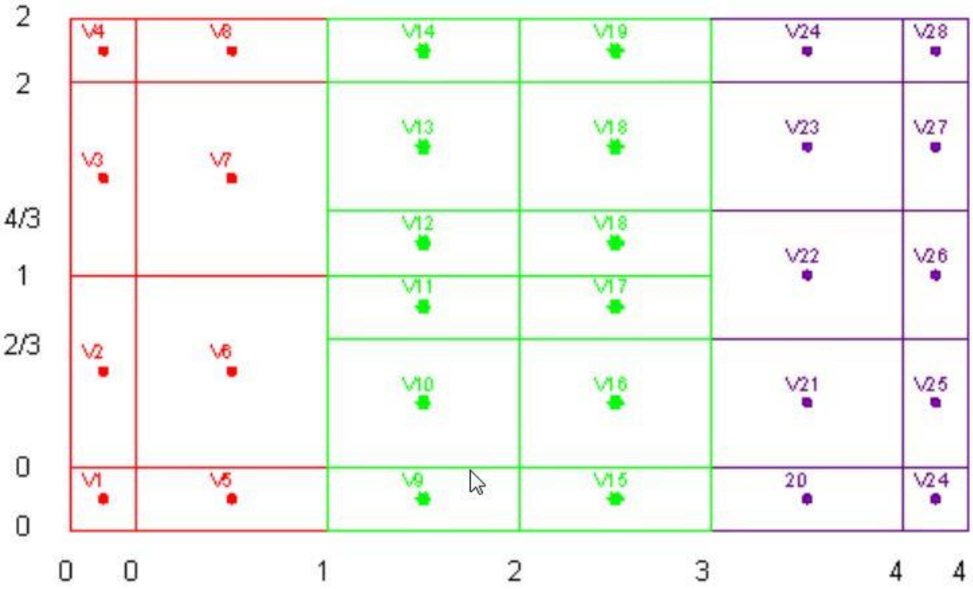


Figure 3.14.
Parameter knot value space of the merged NURBS meshes.
B-SPLines have now unified in one watertight T-SPLine surface.
T-junction have now appeared and
the multiplicity of the merged parts creates a C^0 continuity.

The above procedure indicates how T-SPLines are able to overcome the problem of patch interconnection, providing the sought structure division.

3.3. Stiffness Matrix Assembly

The general process for the total stiffness matrix assembly, as obtained from finite element method, is shown in the following flow chart.

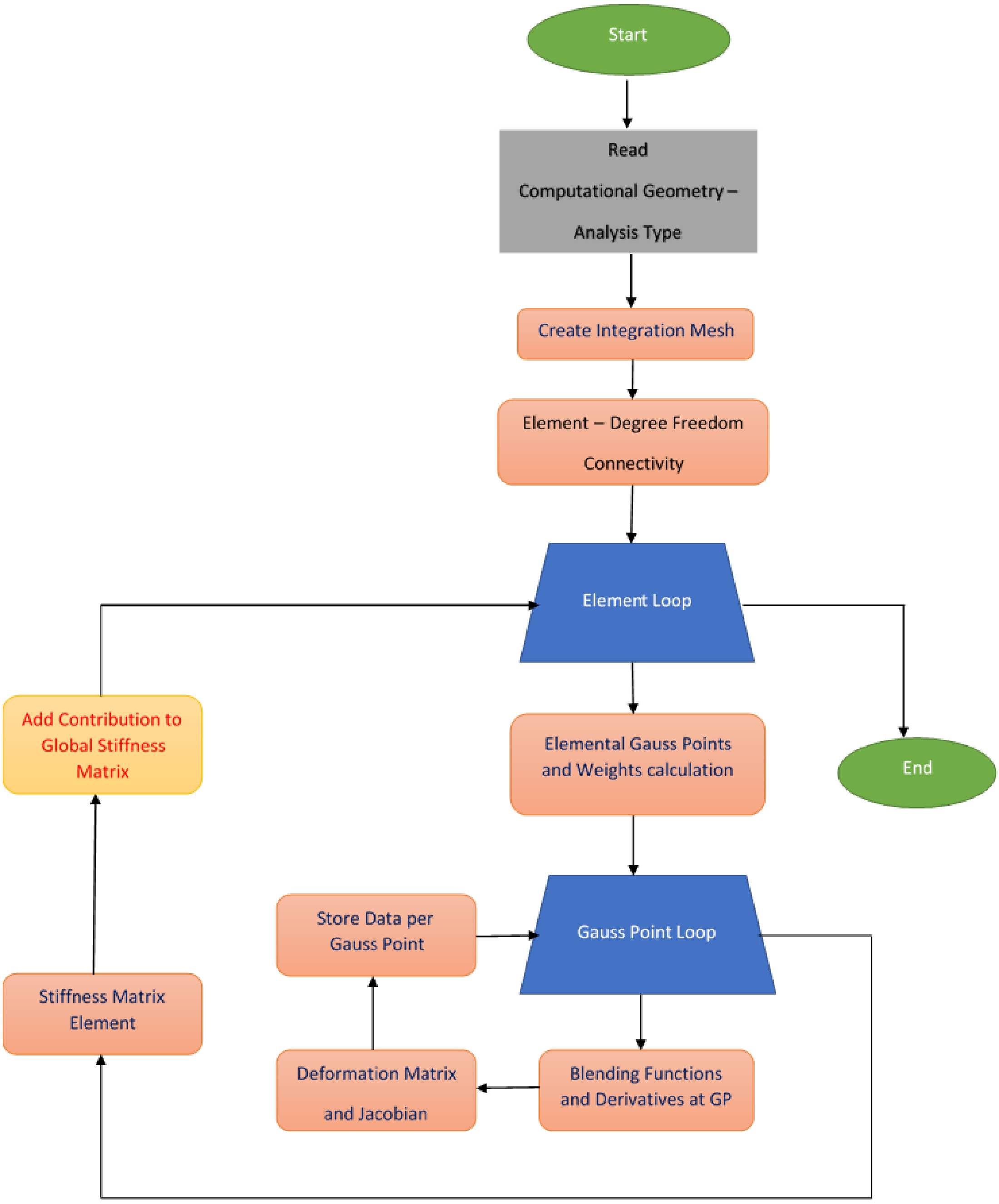
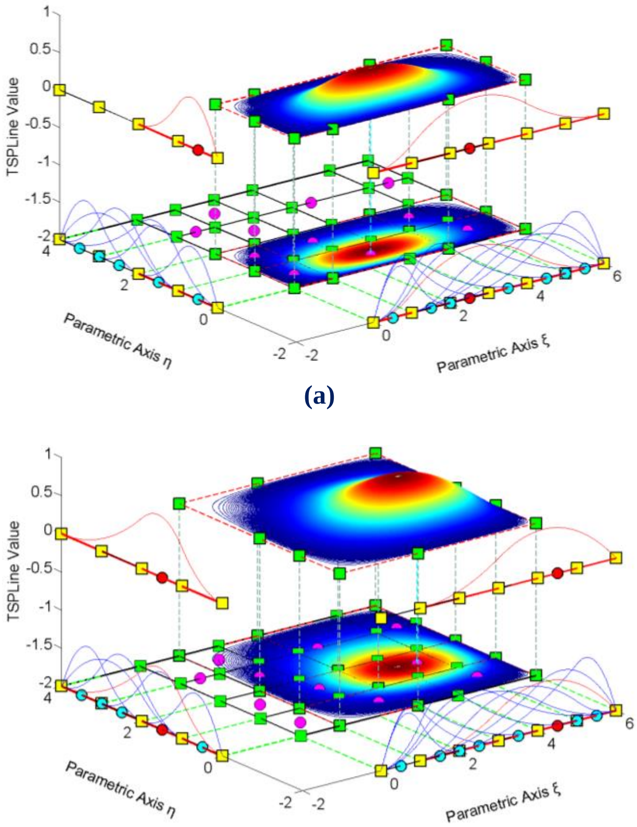


Figure 3.15.
Stiffness matrix assembly in finite element method.

There are two loops:

- Element loop
- Gauss point loop

It is worth mentioning that the element loop in IGA using T-SPLines cannot be avoided due to the intriguing structure of the mesh. As described above, the local knot value vector consists of p+2 values for each axis. In case of NURBS, the distance between the consecutive values remains constant. This is a result of the full tensor product of the mesh, which creates a stable number of functions, that influnces each element. This property is no longer valid in T-SPLines, as the spans do not correspond any more to one knot value span. So, each element is affected by a different number of functions, compared to NURBS, where this number remains the same for all elements.



(a)

(b)

Figure 3.16. (Created with GeomIso)
Shape functions with varied support.

The essential information for the formulation of stiffness matrix is divided into:

- Structural analysis and material
- Computational geometry

3.4. Stiffness Matrix

3.4.1. 1D

One dimensional problem can be compared to a uniaxial truss element under longitudinal forces. It is not possible to apply other types of loads, as degrees of freedom are parallel to axis ξ. Thus, these types of elements are trivial in their examination, yet it is crucial to study them to make the transition to greater dimensions easier. In such case, only axial deformation for each point of the truss exists. This deformation is $u(x) = u(C(\xi)) = u(\xi)$. The strain matrix consists of only one value:

$$\underset{(1\times1)}{\{\varepsilon\}} = [\varepsilon_x] = \left[\frac{\partial u}{\partial x}\right]$$

In order to calculate the derivative of deformation, we must first establish a transformation between physical space and parameter space, i.e. the Jacobian one.

$$\frac{\partial\varphi}{\partial x} = \frac{\partial\varphi}{\partial\xi}\frac{\partial\xi}{\partial x}$$
$$\frac{\partial\varphi}{\partial\xi} = \frac{\partial\varphi}{\partial x}\frac{\partial x}{\partial\xi}$$

$$\left[\frac{\partial\varphi}{\partial\xi}\right] = [J]\cdot\left[\frac{\partial\varphi}{\partial x}\right]$$

where $[J]$ is the Jacobian matrix, which enables transition from physical to parameter space and vice-versa. It can be evaluated with the help of blending functions $R_i(\xi)$ and control points' Cartesian coordinates X_i as shown:

$$\underset{(1\times1)}{[J(\xi)]} = \underset{(1\times n)}{\left[R_{1,\xi}(\xi) \quad R_{2,\xi}(\xi) \quad \ldots \quad \ldots \quad \ldots \quad R_{n,\xi}(\xi)\right]}\cdot\underset{(n\times1)}{\begin{bmatrix} X_1 \\ X_2 \\ \cdot \\ \cdot \\ \cdot \\ X_n \end{bmatrix}}$$

where $R_{i,\xi}(\xi) = \frac{\partial}{\partial\xi}R_i(\xi)$.

In finite element analysis, the reverse transformation is applied. This is why the inverse matrix $[\mathrm{J}]^{-1}$ is needed.

$$\underset{(1\mathrm{x}1)}{[\mathrm{J}]^{-1}} = \frac{1}{\underset{(1\times1)}{[\mathrm{J}]}}$$

Special care has to be taken to assure the correct calculation of the Jacobian. The positive direction of axes in parameter and physical space must coincide, or else the determinant of the Jacobian will be negative and the matrix $[\mathrm{J}]$ irreversible. Numerical integration on points of singularity, such as two points on parameter space mapped into the same point on physical space, has to be avoided as well as far as NURBS are concerned. In case of T-SPLines, this obstacle does not exist, as the complex T-Mesh allows the same representation, without using singularity points.

The next step is to calculate the matrices $[\mathrm{B}_1]$ and $[\mathrm{B}_2]$. Matrix $[\mathrm{B}_1]$ transfers the strains of the element from parameter to physical space and matrix $[\mathrm{B}_2]$ transfers the nodal displacements of the elements to the strains at the parameter space. Therefore, the matrices $[\mathrm{B}_1]$and $[\mathrm{B}_2]$ can be calculated from the following equations:

$$\underset{(1\times1)}{\{\varepsilon\}} = \left[\frac{\partial \mathrm{u}}{\partial \mathrm{x}}\right] = [\mathrm{J}]^{-1} \cdot \left[\frac{\partial \mathrm{u}}{\partial \xi}\right]$$

$$\underset{(1\times1)}{[\mathrm{B}_1(\xi)]} = \left[\frac{1}{\mathrm{J}_{11}}\right]$$

$$\underset{(1\times1)}{\{\varepsilon\}} = \underset{(1\times1)}{[\mathrm{B}_1]} \cdot \underset{(1\times1)}{\{\mathrm{u}_\xi\}}$$

$$\left[\frac{\partial \mathrm{u}}{\partial \xi}\right] = \left[\mathrm{R}_{1,\xi}(\xi) \quad \mathrm{R}_{2,\xi}(\xi) \quad \ldots \quad \ldots \quad \ldots \quad \mathrm{R}_{\mathrm{n},\xi}(\xi)\right] \cdot \begin{bmatrix} \mathrm{u}_1 \\ \mathrm{u}_2 \\ \cdot \\ \cdot \\ \cdot \\ \mathrm{u}_\mathrm{n} \end{bmatrix}$$

$$\underset{(1\times\mathrm{n})}{[\mathrm{B}_2(\xi)]} = \left[\mathrm{R}_{1,\xi}(\xi) \quad \mathrm{R}_{2,\xi}(\xi) \quad \ldots \quad \ldots \quad \ldots \quad \mathrm{R}_{\mathrm{n},\xi}(\xi)\right]$$

The vector $\underset{(1\times1)}{\{u_\xi\}}$ is equal to the product between [B2] and the control point displacement vector $\underset{(n\times1)}{\{d\}}$.

$$\underset{(1\times1)}{\{u_\xi\}} = \underset{(1\times n)}{[B_2]} \cdot \underset{(n\times1)}{\{d\}}$$

As deformation sub-matrices $\underset{(1\times1)}{\left[B_1(\xi)\right]}$ and $\underset{(1\times n)}{\left[B_2(\xi)\right]}$ are known, deformation matrix $\underset{(1\times n)}{\left[B(\xi)\right]}$ is given by:

$$\underset{(1\times n)}{\left[B(\xi)\right]} = \underset{(1\times1)}{\left[B_1(\xi)\right]} \cdot \underset{(1\times n)}{\left[B_2(\xi)\right]}$$

Deformation matrix produces strain values anywhere in the model, by utilizing nodal displacements.

$$\underset{(1\times1)}{\left\{\varepsilon(\xi)\right\}} = \underset{(1\times n)}{\left[B(\xi)\right]} \cdot \underset{(n\times1)}{\{d\}}$$

Stiffness matrix for a patch is evaluated as shown:

$$\underset{(n\times n)}{[K]} = \int_{\xi_0}^{\xi_{n+p+1}} \underset{(n\times1)}{\left[B(\xi)\right]^T} \cdot \underset{(1x1)}{[E]} \cdot \underset{(1\times n)}{\left[B(\xi)\right]} \cdot A \cdot \det[J]\, d\xi$$

Direct integration is almost never applicable. Numerical integration is used instead, looping through all the Gauss points of a patch and their respective weights:

$$\underset{(n\times n)}{[K]} = \sum_{i=1}^{GP\xi} \left\{ \underset{(n\times1)}{\left[B(\xi_i)\right]^T} \cdot \underset{(1x1)}{[E]} \cdot \underset{(1\times n)}{\left[B(\xi_i)\right]} \cdot A \cdot \det[J] \cdot w_i^{GP\xi} \right\}$$

where:

- A : the area of the cross-section
- GP_ξ : the total number of Gauss points for the specific patch
- $\xi_{\ i}$: the parametric coordinates of Gauss points
- $w_i^{GP\xi}$: the weights of Gauss points

3.4.2. 2D

Two dimensional problems can be seen anywhere around us. Both plane strain and plane stress cases can be seen in dams, slabs, cantilevers and cells. Generally all cases that one dimension is significantly smaller (or larger) than the other two can be described as two dimensional. The main difference is, obviously, the utilization of one more dimension. Parameter space is defined on (ξ,η) and physical space on (x,y). Displacements per x, y at any point in the entire domain are defined as $u(x,y)=u(S(\xi,\eta))=u(\xi,\eta)$ and $v(x,y)=v(\xi,\eta)$, respectively.

The strain vector is defined as:

$$\underset{(3\times1)}{\{\varepsilon\}}=\begin{bmatrix}\varepsilon_x\\ \varepsilon_y\\ \gamma_{xy}\end{bmatrix}=\begin{bmatrix}\frac{\partial u}{\partial x}\\ \frac{\partial v}{\partial y}\\ \frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\end{bmatrix}\Rightarrow\underset{(3\times1)}{\{\varepsilon\}}=\begin{bmatrix}\frac{\partial}{\partial x} & 0\\ 0 & \frac{\partial}{\partial y}\\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x}\end{bmatrix}\cdot\begin{bmatrix}u\\ v\end{bmatrix}$$

The transformation of a function φ between parameter and physical space yields:

$$\frac{\partial\varphi}{\partial x}=\frac{\partial\varphi}{\partial\xi}\frac{\partial\xi}{\partial x}+\frac{\partial\varphi}{\partial\eta}\frac{\partial\eta}{\partial x}$$

$$\frac{\partial\varphi}{\partial y}=\frac{\partial\varphi}{\partial\xi}\frac{\partial\xi}{\partial y}+\frac{\partial\varphi}{\partial\eta}\frac{\partial\eta}{\partial y}$$

$$\frac{\partial\varphi}{\partial\xi}=\frac{\partial\varphi}{\partial x}\frac{\partial x}{\partial\xi}+\frac{\partial\varphi}{\partial y}\frac{\partial y}{\partial\xi}$$

$$\frac{\partial\varphi}{\partial\eta}=\frac{\partial\varphi}{\partial x}\frac{\partial x}{\partial\eta}+\frac{\partial\varphi}{\partial y}\frac{\partial y}{\partial\eta}$$

Thus, the 2D Jacobian matrix can be defined as:

$$\begin{bmatrix}\frac{\partial\varphi}{\partial\xi}\\ \frac{\partial\varphi}{\partial\eta}\end{bmatrix}=\begin{bmatrix}\frac{\partial x}{\partial\xi} & \frac{\partial y}{\partial\xi}\\ \frac{\partial x}{\partial\eta} & \frac{\partial y}{\partial\eta}\end{bmatrix}\cdot\begin{bmatrix}\frac{\partial\varphi}{\partial x}\\ \frac{\partial\varphi}{\partial y}\end{bmatrix}\Rightarrow\begin{bmatrix}\frac{\partial\varphi}{\partial\xi}\\ \frac{\partial\varphi}{\partial\eta}\end{bmatrix}=\underset{(2\times2)}{[J]}\cdot\begin{bmatrix}\frac{\partial\varphi}{\partial x}\\ \frac{\partial\varphi}{\partial y}\end{bmatrix}$$

and the inverse mapping:

$$\begin{bmatrix} \dfrac{\partial\varphi}{\partial x} \\ \dfrac{\partial\varphi}{\partial y} \end{bmatrix} = \underset{(2\times2)}{[J]}^{-1} \cdot \begin{bmatrix} \dfrac{\partial\varphi}{\partial\xi} \\ \dfrac{\partial\varphi}{\partial\eta} \end{bmatrix}$$

The Jacobian matrix can be calculated as follows:

$$\underset{(2\times2)}{[J]} = \begin{bmatrix} R_{1,\xi}(\xi,\eta) & R_{2,\xi}(\xi,\eta) & \dots & \dots & \dots & R_{N,\xi}(\xi,\eta) \\ R_{1,\eta}(\xi,\eta) & R_{2.\eta}(\xi,\eta) & \dots & \dots & \dots & R_{N,\eta}(\xi,\eta) \end{bmatrix} \cdot \begin{bmatrix} X_1 & Y_1 \\ X_2 & Y_2 \\ . & . \\ . & . \\ . & . \\ X_N & Y_N \end{bmatrix}$$

where N is the total number of control points.

The inverse Jacobian matrix is used in stiffness matrix calculation:

$$\underset{(2\times2)}{[J]}^{-1} = \begin{bmatrix} J^*_{11} & J^*_{12} \\ J^*_{21} & J^*_{22} \end{bmatrix} = \frac{1}{\det[J]} \cdot \begin{bmatrix} J_{22} & -J_{12} \\ -J_{21} & J_{11} \end{bmatrix}$$

The determinant of the Jacobian matrix is also required and is equal to:

$$\det[J] = J_{11} \cdot J_{22} - J_{21} \cdot J_{12}$$

In order to calculate the deformation matrix for 2D problems, $[B_1]$ and $[B_2]$ have to be evaluated as usual.

To obtain matrix $[B_1]$:

$$\underset{(3\times1)}{\{\varepsilon\}} = \begin{bmatrix} \dfrac{\partial u}{\partial x} \\ \dfrac{\partial v}{\partial y} \\ \dfrac{\partial u}{\partial y} + \dfrac{\partial v}{\partial x} \end{bmatrix} = \frac{1}{\det[J]} \cdot \begin{bmatrix} J_{22} & -J_{12} & 0 & 0 \\ 0 & 0 & -J_{21} & J_{11} \\ -J_{21} & J_{11} & J_{22} & -J_{12} \end{bmatrix} \cdot \begin{bmatrix} \dfrac{\partial u}{\partial\xi} \\ \dfrac{\partial u}{\partial\eta} \\ \dfrac{\partial v}{\partial\xi} \\ \dfrac{\partial v}{\partial\eta} \end{bmatrix}$$

Hence,

$$\underset{(3\times4)}{\left[B_1(\xi,\eta)\right]}=\frac{1}{\det[J]}\cdot\begin{bmatrix} J_{22} & -J_{12} & 0 & 0 \\ 0 & 0 & -J_{21} & J_{11} \\ -J_{21} & J_{11} & J_{22} & -J_{12} \end{bmatrix}$$

To calculate matrix $[B_2]$:

$$\begin{bmatrix} \frac{\partial u}{\partial \xi} \\ \frac{\partial u}{\partial \eta} \\ \frac{\partial v}{\partial \xi} \\ \frac{\partial v}{\partial \eta} \end{bmatrix}=\begin{bmatrix} R_{1,\xi} & 0 & R_{2,\xi} & 0 & \dots & \dots & \dots & R_{N,\xi} & 0 \\ R_{1,\eta} & 0 & R_{2,\eta} & 0 & \dots & \dots & \dots & R_{N,\eta} & 0 \\ 0 & R_{1,\xi} & 0 & R_{2,\xi} & \dots & \dots & \dots & 0 & R_{N,\xi} \\ 0 & R_{1,\eta} & 0 & R_{2,\eta} & \dots & \dots & \dots & 0 & R_{N,\eta} \end{bmatrix}\cdot\begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ . \\ . \\ . \\ u_N \\ v_N \end{bmatrix}$$

Hence,

$$\underset{(4\times2N)}{\left[B_2(\xi,\eta)\right]}=\begin{bmatrix} R_{1,\xi} & 0 & R_{2,\xi} & 0 & \dots & \dots & \dots & R_{N,\xi} & 0 \\ R_{1,\eta} & 0 & R_{2,\eta} & 0 & \dots & \dots & \dots & R_{N,\eta} & 0 \\ 0 & R_{1,\xi} & 0 & R_{2,\xi} & \dots & \dots & \dots & 0 & R_{N,\xi} \\ 0 & R_{1,\eta} & 0 & R_{2,\eta} & \dots & \dots & \dots & 0 & R_{N,\eta} \end{bmatrix}$$

Having determined $[B_1]$ and $[B_2]$, the deformation matrix is calculated as:

$$\underset{(3\times2N)}{\left[B(\xi,\eta)\right]}=\underset{(3\times4)}{\left[B_1(\xi,\eta)\right]}\cdot\underset{(4\times2N)}{\left[B_2(\xi,\eta)\right]}$$

In order to evaluate stiffness matrix, integration is required.

$$\underset{(2N\times2N)}{[K]}=\int_{\xi_0}^{\xi_{n+p+1}}\int_{\eta_0}^{\eta_{m+q+1}}\underset{(2N\times3)}{\left[B(\xi,\eta)\right]^T}\cdot\underset{(3x3)}{[E]}\cdot\underset{(3\times2N)}{\left[B(\xi,\eta)\right]}\cdot t\cdot\det[J]\,d\eta d\xi$$

Numerical integration procedures for ξ,η lead to integration for tensor product Gauss points.

$$\underset{(2N\times2N)}{[K]}=\sum_{i=1}^{GP\xi}\sum_{j=1}^{GP\eta}\underset{(2N\times3)}{\left[B(\xi_i,\eta_j)\right]^T}\cdot\underset{(3x3)}{[E]}\cdot\underset{(3\times2N)}{\left[B(\xi_i,\eta_j)\right]}\cdot t\cdot\det[J]\cdot w_i^{GP\xi}\cdot w_j^{GP\eta}$$

where:

- t: the thickness of the cross section
- GP_{ξ}: the total number of Gauss points per ξ for the specific patch
- GP_{η}: the total number of Gauss points per η for the specific patch
- ξ_i, η_j: the parametric coordinates of the tensor product Gauss point (i, j)
- $w_i^{GP\xi}$, $w_j^{GP\eta}$: the weights of the tensor product Gauss point (i, j)

The only difference, at this point, between plane stress and plane strain is the elasticity matrix, which is the result of the utilized constitutive law, which connects stress and strain field.

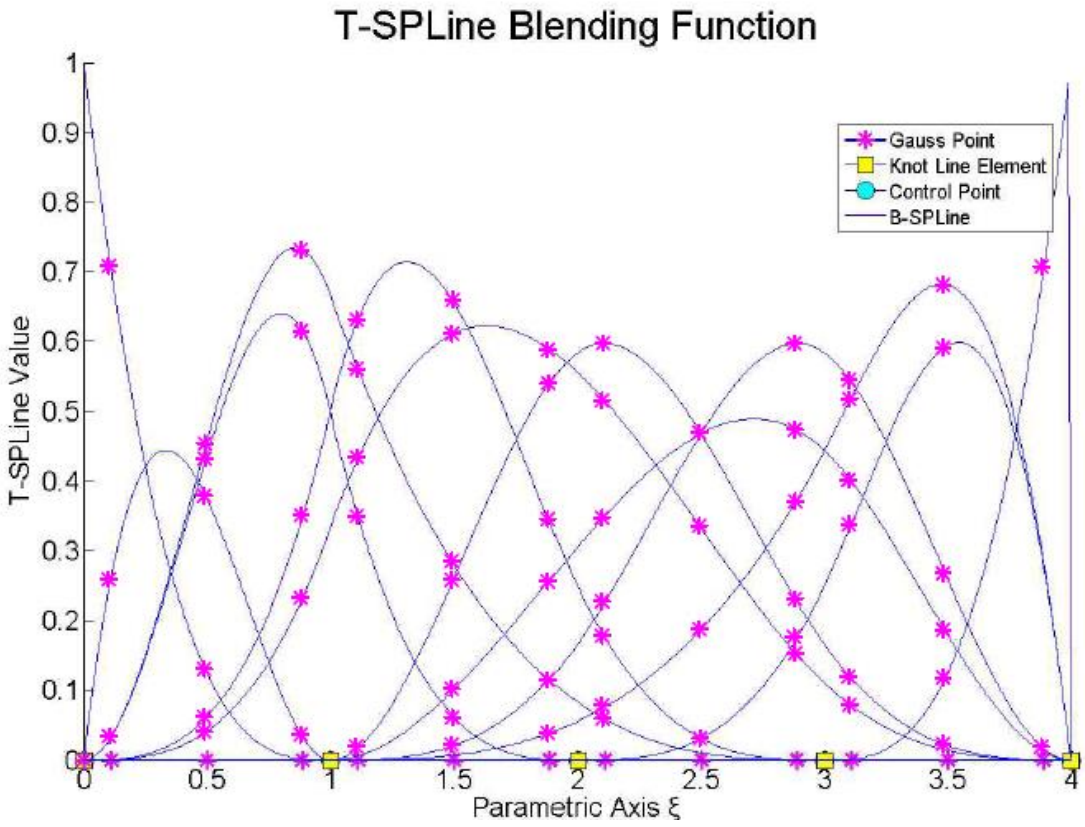


Figure 3.17.
Parameter space with blending functions as blue lines,
knots as yellow squares and Gauss points as magenta stars.
Blending functions evaluated at Gauss points.
(Created with GeomIso)

3.4.3. 3D

3D elasticity is merely the extension of 2D elasticity in all directions, with a complete stress field. Every other problem can be created by downgrading 3D problems into 2D and 1D problems. The displacement field for each point in physical space is now defined for x, y, z by $u(x,y,z)=u(S(\xi,\eta,\zeta))=u(\xi,\eta,\zeta)$, $v(\xi,\eta,\zeta)$, $w(\xi,\eta,\zeta)$ respectively. The strain field can now be defined as:

$$\underset{(6\times1)}{\{\varepsilon\}}=\begin{bmatrix}\varepsilon_x\\ \varepsilon_y\\ \varepsilon_z\\ \gamma_{xy}\\ \gamma_{yz}\\ \gamma_{zx}\end{bmatrix}=\begin{bmatrix}\frac{\partial u}{\partial x}\\ \frac{\partial v}{\partial y}\\ \frac{\partial w}{\partial z}\\ \frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\\ \frac{\partial v}{\partial z}+\frac{\partial w}{\partial y}\\ \frac{\partial w}{\partial x}+\frac{\partial u}{\partial z}\end{bmatrix}=\begin{bmatrix}\frac{\partial}{\partial x} & 0 & 0\\ 0 & \frac{\partial}{\partial y} & 0\\ 0 & 0 & \frac{\partial}{\partial z}\\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0\\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y}\\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x}\end{bmatrix}\cdot\begin{bmatrix}u\\ v\\ w\end{bmatrix}$$

which leads to the definition of the Jacobian matrix for 3D:

$$\begin{bmatrix}\frac{\partial\varphi}{\partial\xi}\\ \frac{\partial\varphi}{\partial\eta}\\ \frac{\partial\varphi}{\partial\zeta}\end{bmatrix}=\begin{bmatrix}\frac{\partial x}{\partial\xi} & \frac{\partial y}{\partial\xi} & \frac{\partial z}{\partial\xi}\\ \frac{\partial x}{\partial\eta} & \frac{\partial y}{\partial\eta} & \frac{\partial z}{\partial\eta}\\ \frac{\partial x}{\partial\zeta} & \frac{\partial y}{\partial\zeta} & \frac{\partial z}{\partial\zeta}\end{bmatrix}\cdot\begin{bmatrix}\frac{\partial\varphi}{\partial x}\\ \frac{\partial\varphi}{\partial y}\\ \frac{\partial\varphi}{\partial z}\end{bmatrix}\Rightarrow\begin{bmatrix}\frac{\partial\varphi}{\partial\xi}\\ \frac{\partial\varphi}{\partial\eta}\\ \frac{\partial\varphi}{\partial\zeta}\end{bmatrix}=\underset{3\times3}{[J]}\cdot\begin{bmatrix}\frac{\partial\varphi}{\partial x}\\ \frac{\partial\varphi}{\partial y}\\ \frac{\partial\varphi}{\partial z}\end{bmatrix}$$

and the inverse Jacobian matrix as well:

$$\begin{bmatrix}\frac{\partial\varphi}{\partial x}\\ \frac{\partial\varphi}{\partial y}\\ \frac{\partial\varphi}{\partial z}\end{bmatrix}=\underset{(3\times3)}{[J]}^{-1}\cdot\begin{bmatrix}\frac{\partial\varphi}{\partial\xi}\\ \frac{\partial\varphi}{\partial\eta}\\ \frac{\partial\varphi}{\partial\zeta}\end{bmatrix}$$

Jacobian matrix can be calculated from the derivatives of the shape functions. It is a square matrix.

$$\underset{(3\times3)}{\left[J\right]}=\begin{bmatrix} R_{1,\xi}(\xi,\eta,\zeta) & R_{2,\xi}(\xi,\eta,\zeta) & \dots & \dots & \dots & R_{N,\xi}(\xi,\eta,\zeta) \\ R_{1,\eta}(\xi,\eta,\zeta) & R_{2.\eta}(\xi,\eta,\zeta) & \dots & \dots & \dots & R_{N,\eta}(\xi,\eta,\zeta) \\ R_{1,\zeta}(\xi,\eta,\zeta) & R_{2,\zeta}(\xi,\eta,\zeta) & \dots & \dots & \dots & R_{N,\zeta}(\xi,\eta,\zeta) \end{bmatrix}\begin{bmatrix} X_1 & Y_1 & Z_1 \\ X_2 & Y_2 & Z_2 \\ . & . & . \\ . & . & . \\ . & . & . \\ X_N & Y_N & Z_N \end{bmatrix}$$

The inverse of the Jacobian matrix is:

$$\underset{(3\times3)}{\left[J\right]^{-1}}=\begin{bmatrix} J^*_{11} & J^*_{12} & J^*_{13} \\ J^*_{21} & J^*_{22} & J^*_{23} \\ J^*_{31} & J^*_{32} & J^*_{33} \end{bmatrix}$$

$$\underset{(6\times9)}{\left[B_1\left(\xi,\eta,\zeta\right)\right]}=\begin{bmatrix} J^*_{11} & J^*_{12} & J^*_{13} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & J^*_{21} & J^*_{22} & J^*_{23} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & J^*_{31} & J^*_{32} & J^*_{33} \\ J^*_{21} & J^*_{22} & J^*_{23} & J^*_{11} & J^*_{12} & J^*_{13} & 0 & 0 & 0 \\ 0 & 0 & 0 & J^*_{31} & J^*_{32} & J^*_{33} & J^*_{21} & J^*_{22} & J^*_{23} \\ J^*_{31} & J^*_{32} & J^*_{33} & 0 & 0 & 0 & J^*_{11} & J^*_{12} & J^*_{13} \end{bmatrix}$$

$$\underset{(9\times3N)}{\left[B_2\left(\xi,\eta,\zeta\right)\right]}=\begin{bmatrix} R_{1,\xi} & 0 & 0 & R_{2,\xi} & 0 & 0 & \dots & \dots & \dots & R_{N,\xi} & 0 & 0 \\ R_{1,\eta} & 0 & 0 & R_{2,\eta} & 0 & 0 & \dots & \dots & \dots & R_{N,\eta} & 0 & 0 \\ R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & 0 & 0 & \dots & \dots & \dots & R_{N,\zeta} & 0 & 0 \\ 0 & R_{1,\xi} & 0 & 0 & R_{2,\xi} & 0 & \dots & \dots & \dots & 0 & R_{N,\xi} & 0 \\ 0 & R_{1,\eta} & 0 & 0 & R_{2,\eta} & 0 & \dots & \dots & \dots & 0 & R_{N,\eta} & 0 \\ 0 & R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & 0 & \dots & \dots & \dots & 0 & R_{N,\zeta} & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & R_{1,\xi} & 0 & 0 & R_{2,\xi} & \dots & \dots & \dots & 0 & 0 & R_{N,\xi} \\ 0 & 0 & R_{1,\eta} & 0 & 0 & R_{2,\eta} & \dots & \dots & \dots & 0 & 0 & R_{N,\eta} \\ 0 & 0 & R_{1,\zeta} & 0 & 0 & R_{2,\zeta} & \dots & \dots & \dots & 0 & 0 & R_{N,\zeta} \end{bmatrix}$$

As a result, the deformation matrix for 3D elasticity is calculated as:

$$\underset{(6\times 3N)}{\left[B(\xi,\eta,\zeta)\right]} = \underset{(6\times 9)}{\left[B_1(\xi,\eta,\zeta)\right]} \cdot \underset{(9\times 3N)}{\left[B_2(\xi,\eta,\zeta)\right]}$$

The corresponding stiffness matrix is produced by integration:

$$\underset{(3N\times 3N)}{[K]} = \int_{\xi_0}^{\xi_{n+p+1}} \int_{\eta_0}^{\eta_{m+q+1}} \int_{\zeta_0}^{\zeta_{l+r+1}} \underset{(3N\times 6)}{\left[B(\xi,\eta,\zeta)\right]^T} \cdot \underset{(6x6)}{[E]} \cdot \underset{(6\times 3N)}{\left[B(\xi,\eta,\zeta)\right]} \cdot \det[J]\, d\zeta d\eta d\xi$$

Numerical integration is used in 3D, as well:

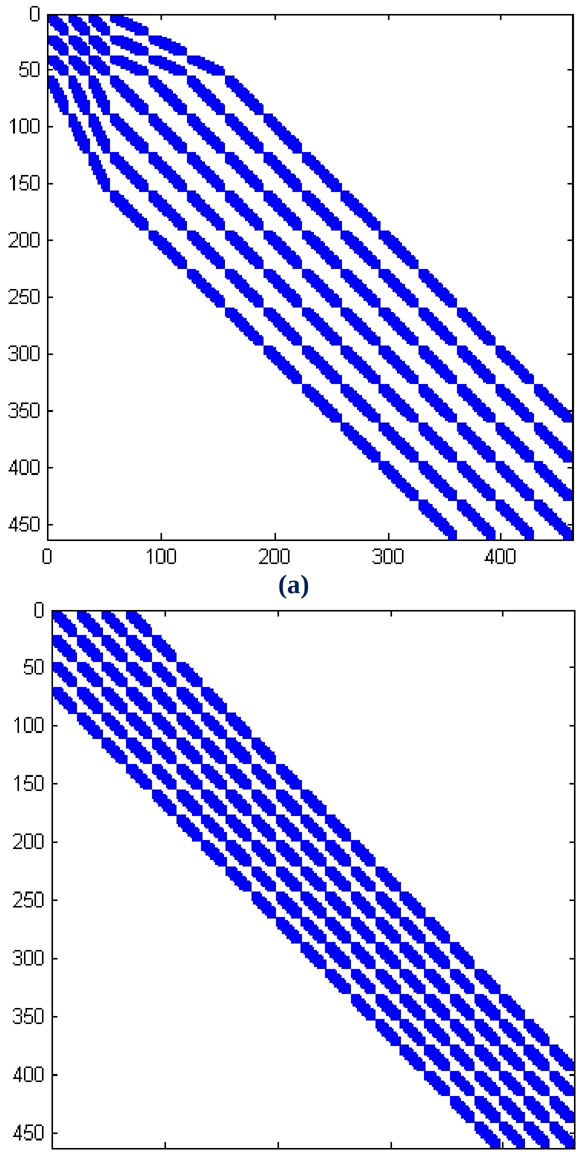
$$\underset{(3N\times 3N)}{[K]} = \sum_{i=1}^{GP\xi} \sum_{j=1}^{GP\eta} \sum_{k=1}^{GP\zeta} \underset{(3N\times 6)}{\left[B\left(\xi_i,\eta_j,\zeta_k\right)\right]^T} \cdot \underset{(6x6)}{[E]} \cdot \underset{(6\times 3N)}{\left[B\left(\xi_i,\eta_j,\zeta_k\right)\right]} \cdot \det[J] \cdot w_i^{GP\xi} \cdot w_j^{GP\eta} \cdot w_k^{GP\zeta}$$

where:

- GP_{ξ} : the total number of Gauss points per ξ for the specific patch
- GP_{η} : the total number of Gauss points per η for the specific patch
- GP_{ζ} : the total number of Gauss points per ζ for the specific patch
- ξ_i : the parametric coordinate ξ of the tensor product Gauss point ijk
- η_j : the parametric coordinate η of the tensor product Gauss point ijk
- ζ_k : the parametric coordinate ζ of the tensor product Gauss point ijk
- $w_i^{GP\xi}$: the univariate weight (ξ) of the tensor product Gauss point ijk
- $w_j^{GP\eta}$: the univariate weight (η) of the tensor product Gauss point ijk
- $w_k^{GP\zeta}$: the univariate weight (ζ) of the tensor product Gauss point ijk

3.4.4. Stiffness Matrix Examples

For two different meshes of the same structure with the same degrees of freedom, when NURBS and T-SPLines are used, significant differences are spotted. The main difference is the bandwidth of the stiffness matrix. NURBS have a constant bandwidth, while in T-SPLines its range varies. The maximum bandwidth is greater than the one in NURBS and this leads to an increased computational cost, but analyses have shown that it provides much greater accuracy. Figure 3.18 depicts two such stiffness matrices. T-SPLine stiffness matrix has a bandwidth of 120 degrees of freedom, while the same stiffness matrix for NURBS has a bandwidth of 70 degrees of freedom.

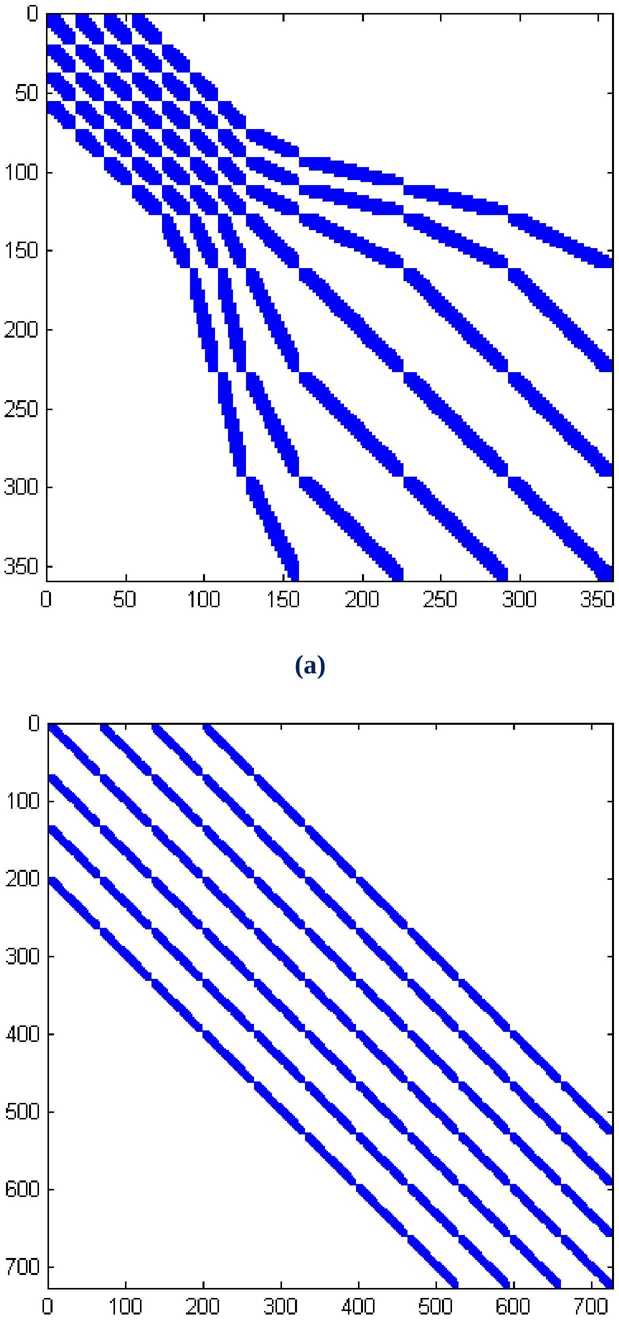


(a)

(b)

Figure 3.18. Stiffness matrix for **(a)** T-SPLines and **(b)** NURBS. (Created with GeomIso)

A possible process could be the transformation of a given T-Mesh into a NURBS one. This is achieved by extending the lines of index space, so as to create a full tensor product structure. These two meshes give the same geometry, but the number of control points becomes automatically larger. In this case, the constant bandwidth of the NURBS stiffness matrix is equal to the maximum bandwidth of the corresponding one in T-SPLines. Figure 3.18 shows the results of the process described. NURBS control points are more and so the scale of the axes is different in these two illustrations and the maximum bandwidth is the same. This procedure seems worthless, as it leads to more unknown degrees of freedom. The process usually utilized is the reverse one for the sake of simplification of calculations.

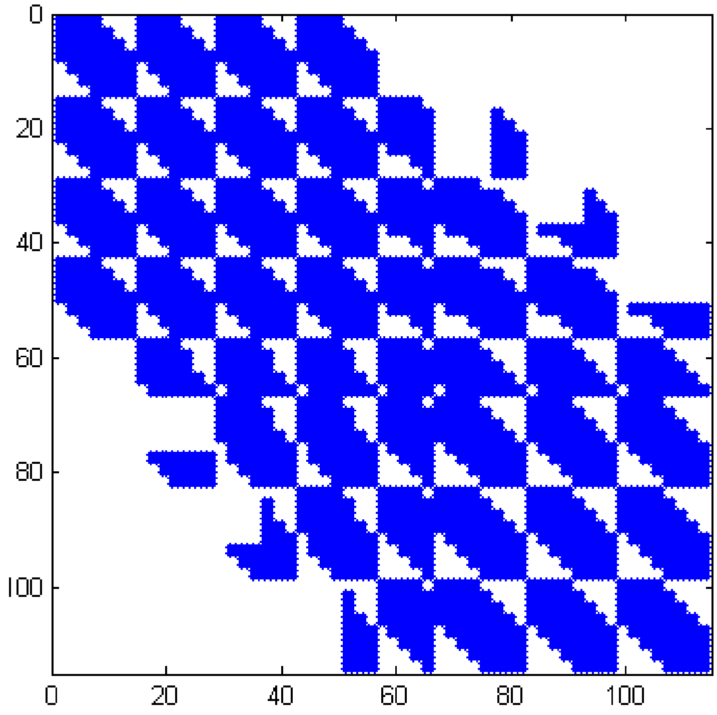


(a)

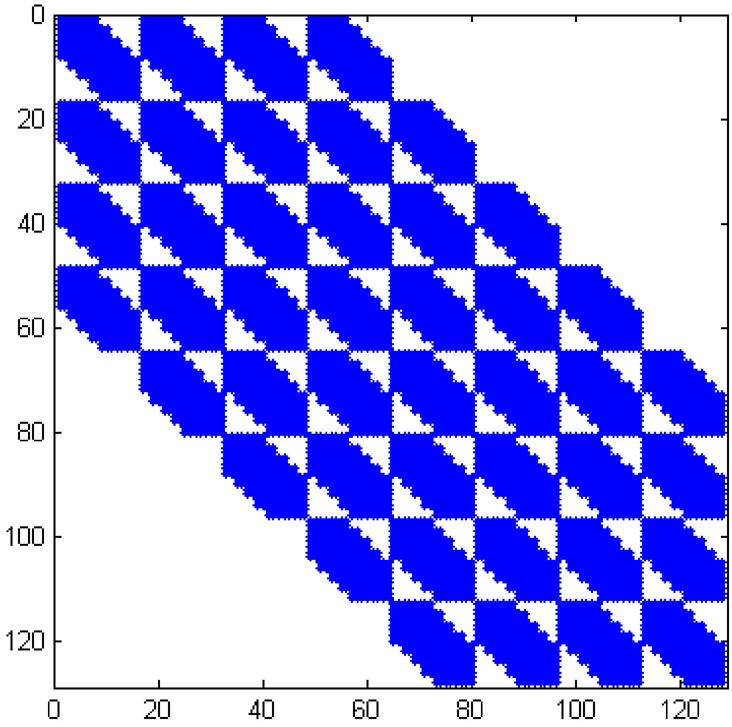
(b)

Figure 3.19.
Stiffness matrix for (a) T-SPLines and (b) NURBS.
(Created with GeomIso)

The property of overlapping behaves similarly to the one of bandwidth. This means that it remains constant in case of NURBS as the local support of shape function is always fixed. In T-SPLines, the local support differs due to the existence of T-junctions and thus the continuity and shape function overlapping changes throughout the domain.



(a)



(b)

Figure 3.20. (Created with GeomIso)
Stiffness matrix for **(a)** T-SPLines and **(b)** NURBS.

3.5. External Loads & Boundary Conditions

3.5.1. External Loads

Having created the stiffness matrix with the aforementioned procedure, it is now imperative to define loads, which comply with the actual load situation of the examined structure. In finite element analysis (FEA), external loads act on the nodes of the structure, which are at the same time material points. In isogeometric analysis (IGA), external loads are imposed not on material points, but on the control net. It's obvious that this handling is far away from a young engineer's perception, who is used to act loads directly on points belonging to the model geometry. The only case in IGA where a load imposed at a control point and a material point at the same time is at interpolatory control points. We should keep in mind that, regardless the type of the external loads chosen for the analysis, the final forces will be implemented on interpolatory control points, due to the fact that only interpolatory control points can be loaded. In case a f(ξ,η,ζ) load has to be distributed on control points, that are not interpolatory to the curve, it has to be transformed into equivalent concentrated loads by integration :

$$\underset{(N\times 1)}{\{F\}} = \int_{\xi_0}^{\xi_{n+p+1}} \int_{\eta_0}^{\eta_{m+q+1}} \int_{\zeta_0}^{\zeta_{l+r+1}} \underset{(N\times 1)}{\{R(\xi,\eta,\zeta)\}} \cdot \underset{(1\times 1)}{f(\xi,\eta,\zeta)} \cdot \det[J] \, d\zeta d\eta d\xi$$

More specifically, for each case:

1D:

$$\underset{(N\times 1)}{\{F\}} = \int_{\xi_0}^{\xi_{n+p+1}} \underset{(N\times 1)}{\{R(\xi)\}} \cdot \underset{(1\times 1)}{f(\xi)} \cdot \det[J] d\xi$$

2D:

$$\underset{(N\times 1)}{\{F\}} = \int_{\xi_0}^{\xi_{n+p+1}} \int_{\eta_0}^{\eta_{m+q+1}} \underset{(N\times 1)}{\{R(\xi,\eta)\}} \cdot \underset{(1\times 1)}{f(\xi,\eta)} \cdot \det[J] d\eta d\xi$$

3D:

$$\underset{(N\times 1)}{\{F\}} = \int_{\xi_0}^{\xi_{n+p+1}} \int_{\eta_0}^{\eta_{m+q+1}} \int_{\zeta_0}^{\zeta_{l+r+1}} \underset{(N\times 1)}{\{R(\xi,\eta,\zeta)\}} \cdot \underset{(1\times 1)}{f(\xi,\eta,\zeta)} \cdot \det[J] d\zeta d\eta d\xi$$

This way, the load vector $\{F\}$ is assembled.

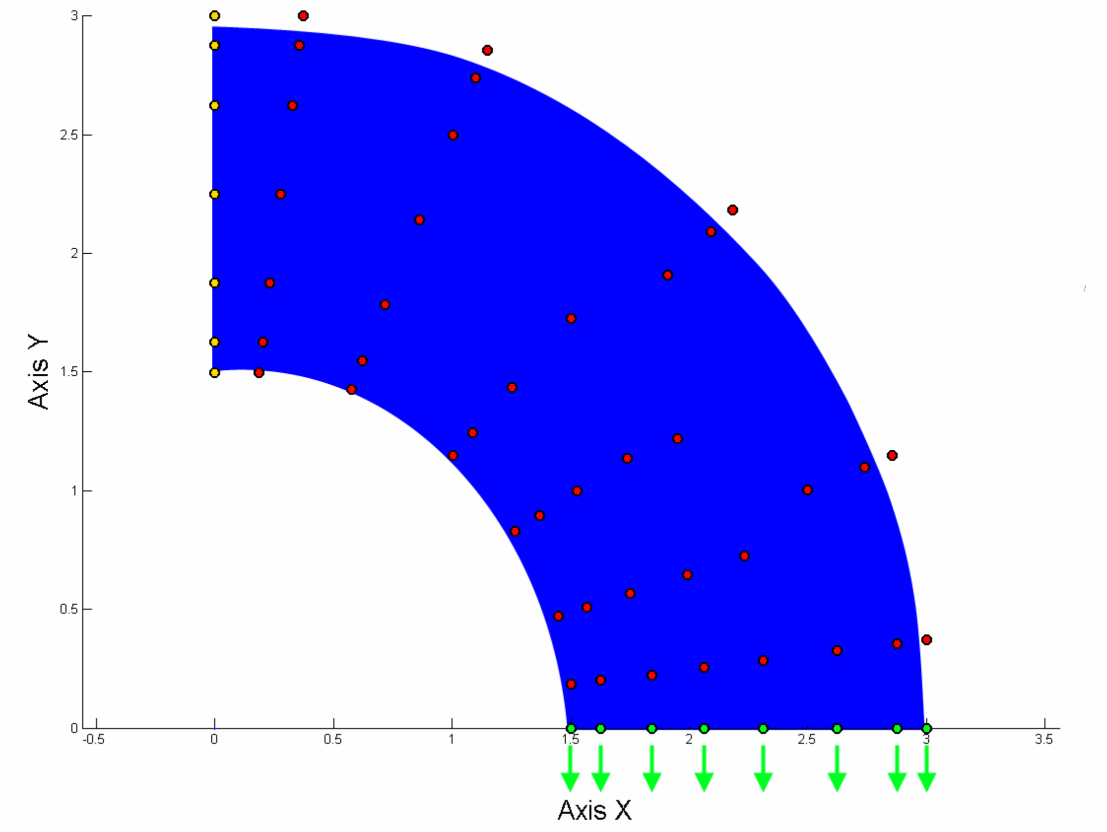


Figure 3.21. (Created with GeomIso)
Physical space. Quarter of an annulus. 2D Analysis.
Red color refers to free control points and yellow to the supported ones.
Concentrated loads have been acted at green control points.

3.5.2. Boundary Conditions

A crucial step, that allows analysis to be done, is the enforcement of the boundary conditions, as structures cannot be analyzed when they are not stable. This means that some degrees of freedom will be free and some have to be fixed, i.e. their displacements will be zero. These degrees of freedom are called stationary and their corresponding rows and columns are deleted from the stiffness matrix and the load vector. The stiffness matrix and the load vector having only free degrees of freedom are $\left[K_{ff}\right]$ and $\left\{F_f\right\}$ respectively. The solution of the equation is the final step in analysis:

$$\left\{F_f\right\} = \left[K_{ff}\right] \cdot \left\{D_f\right\} \Rightarrow \left\{D_f\right\} = \left[K_{ff}\right]^{-1} \cdot \left\{F_f\right\}$$

The (zero) displacements for the stationary degrees of freedom are added back to the result creating the displacement vector $\left\{D\right\}$.

The necessity of applying boundary conditions and the inability to apply strong formulation leads to the application of a weak one. The weak formulation of such boundary conditions demands that they are applicable on a finite number of points. The enforcement of the boundary conditions takes place at the control points of the structure. Each control point has one, two or three displacement degrees of freedom for 1D, 2D and 3D analysis respectively. In this reference, we deal with the enforcement of 2D boundary conditions (plane stress/strain), which are enforced at the corners of the domain's boundaries. At these points, basis functions have C^{-1} continuity and one component of them refers to corner control point. We aim to these control points' commitment only when their corresponding basis function has non-zero value. Having set zero corner control point's displacements, we ensure that all the material points of the specific boundary side have zero displacements, too. For example, on parametric axis ξ:

$$\sum_{i=1}^{n+p+1} N_i D_i = 0 \Rightarrow N_1 D_1 + \sum_{i=2}^{n+p} N_i D_i + N_{n+p+1} D_{n+p+1} = 0 \Rightarrow \sum_{i=2}^{n+p} N_i D_i = 0$$

The procedure of the boundary condition enforcement is getting complicated, because the domain of influence of each control point overlaps the domains of its adjoining control points. Compared with finite element analysis, in which control points are simultaneously material points, in isogeometric analysis control points may be points out of the structure's body. So, another difficulty is ahead, concerning the enforcement of boundary conditions in IGA. Enforcing boundary conditions in this way, we do not deal with a general case of mixed conditions on a boundary domain. Every researcher should give special attention to its implementation and dare with all the above difficulties.

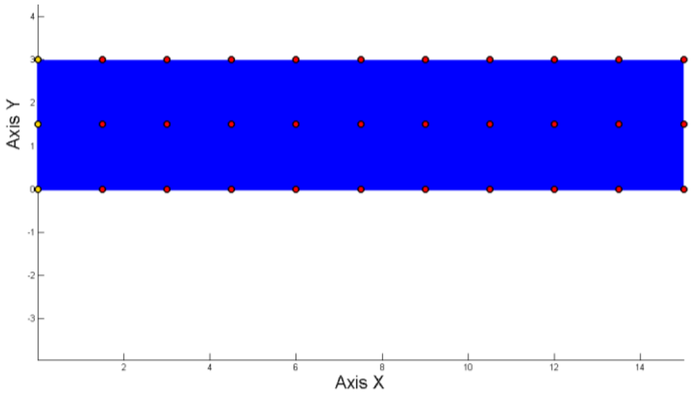


Figure 3.22. (Created with GeomIso)
Physical space. Cook's 2D cantilever.
Red color refers to free control points and yellow to the supported ones.

The displacement vector $\{\mathrm{D}\}$ allows us to calculate the equivalent actions at the control points and to estimate the support reactions of the structure.

3.6. Displacement, Strain and Stress Field

3.6.1. Displacement

After solving the equation, control points' displacements are obtained. Unlike classical FEM, control points are usually located outside the area of the model. The displacements of the model's material points differ from the displacements of the corresponding control points. Conclusively, these analysis results are considered "pseudo-displacements" and play an auxiliary role in calculating the real ones. As mentioned before, the distribution of the displacement field is achieved via shape functions.

1D:

$$d(\xi) = \sum_{i=1}^{N} \left\{ R_i(\xi) \cdot D_i \right\} = \underset{(1\times N)}{\left\{ R_i(\xi) \right\}^T} \cdot \underset{(N\times 1)}{\left\{ D \right\}}$$

2D:

$$d(\xi,\eta) = \sum_{i=1}^{N} \left\{ R_i(\xi,\eta) \cdot D_i \right\} = \underset{(1\times N)}{\left\{ R_i(\xi,\eta) \right\}^T} \cdot \underset{(N\times 1)}{\left\{ D \right\}}$$

3D:

$$d(\xi,\eta,\zeta) = \sum_{i=1}^{N} \left\{ R_i(\xi,\eta,\zeta) \cdot D_i \right\} = \underset{(1\times N)}{\left\{ R_i(\xi,\eta,\zeta) \right\}^T} \cdot \underset{(N\times 1)}{\left\{ D \right\}}$$

where:

- R_i is the shape function i
- D_i is the displacement of the corresponding control point i
- N is the total number of control points

If a control point c is interpolatory to the curve at (ξ_c, η_c, ζ_c), it follows that:

$$d(\xi_c,\eta_c,\zeta_c) = \sum_{i=1}^{N} \left\{ R_i(\xi_c,\eta_c,\zeta_c) \cdot D_i \right\} = 1 \cdot D_c = D_c$$

Displacements of interpolatory control points are material points' displacements as well.

3.6.2. Stress and Strain

The strain vector can be evaluated at any point in the field with the help of control point displacements and the deformation matrix $[B]$:

1D

$$\underset{(1\times1)}{\{\varepsilon(\xi)\}} = \underset{(1\times N)}{\left[B(\xi)\right]}\cdot\underset{(N\times1)}{\{D\}}$$

2D

$$\underset{(3\times1)}{\{\varepsilon(\xi,\eta)\}} = \underset{(3\times2N)}{\left[B(\xi,\eta)\right]}\cdot\underset{(2N\times1)}{\{D\}}$$

3D

$$\underset{(6\times1)}{\{\varepsilon(\xi,\eta,\zeta)\}} = \underset{(6\times3N)}{\left[B(\xi,\eta,\zeta)\right]}\cdot\underset{(3N\times1)}{\{D\}}$$

Applying Hooke’s constitutive law leads to:

1D

$$\underset{(1\times1)}{\{\sigma(\xi)\}} = \underset{(1\times1)}{[E]}\cdot\underset{(1\times1)}{\{\varepsilon(\xi)\}} = \underset{(1\times1)}{[E]}\cdot\underset{(1\times N)}{\left[B(\xi)\right]}\cdot\underset{(N\times1)}{\{D\}}$$

2D

$$\underset{(3\times1)}{\{\sigma(\xi,\eta)\}} = \underset{(3\times3)}{[E]}\cdot\underset{(3\times1)}{\{\varepsilon(\xi,\eta)\}} = \underset{(3\times3)}{[E]}\cdot\underset{(3\times2N)}{\left[B(\xi,\eta)\right]}\cdot\underset{(2N\times1)}{\{D\}}$$

3D

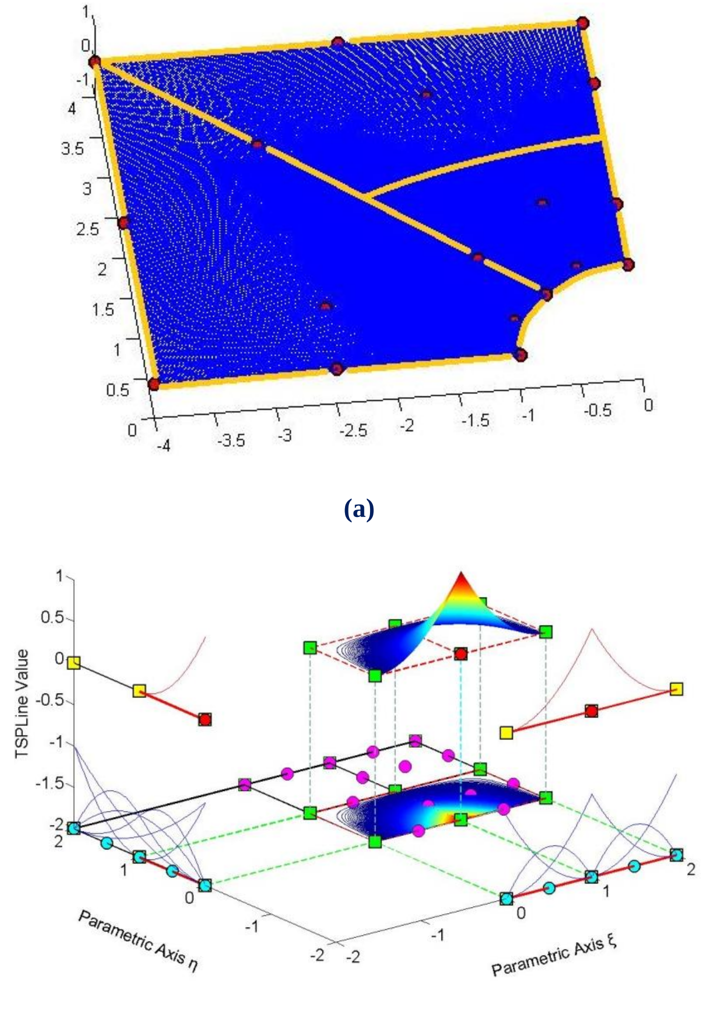
$$\underset{(6\times1)}{\{\sigma(\xi,\eta,\zeta)\}} = \underset{(6\times6)}{[E]}\cdot\underset{(6\times1)}{\{\varepsilon(\xi,\eta,\zeta)\}} = \underset{(6\times6)}{[E]}\cdot\underset{(6\times3N)}{\left[B(\xi,\eta,\zeta)\right]}\cdot\underset{(3N\times1)}{\{D\}}$$

Note that stress and strain vectors are evaluated via the derivatives of the shape functions. This means that their distribution is going to be one order less than the displacement distribution. This is why stress and strain continuity cannot be achieved in FEM models, where shape functions are always C^{-1} continuous. This problem is solved when the derivatives of the shape functions are also continuous, which means that used shape functions have C^{1} continuity or higher.

4. Applications

4.1. Plate with a hole

The plate with a hole presented here is a structure subjected to plane stress, as its thickness is significantly smaller than the other two dimensions (height, length). Thiw model subjected to uniaxial tension, due to symmetrical geometry, boundary and load conditions, can be divided into four identical plates for the sake of simplicity. The examined model is one quarter of the real one.



(a)

(b)

Figure 4.1. Plate with a hole. (Created with GeomIso)

(a) Physical space.

(b) Parameter space. Basis functions for plate with a hole.

In Figure 4.1(a), the physical space of a plate with a hole is shown. Twelve control points are depicted. It is obvious that only control points on the boundary are interpolatory to surface. A question, that can arise now, is which is the difference between interpolatory control points and all the others, as all control points seem to be material points (but they are not). A control point is interpolatory, when it is moved and still remains a point of the surface. This occurs only when the shape function of the particular anchor-control point is of reduced continuity and thus of magnitude C^0 or C^{-1}.

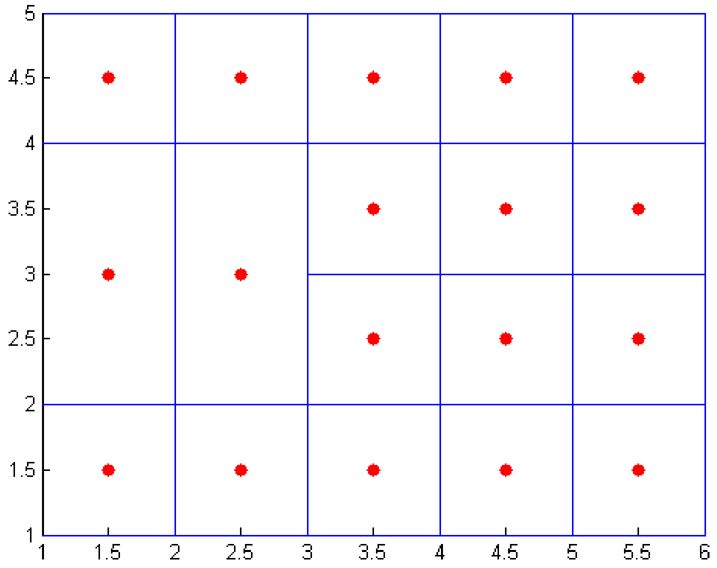
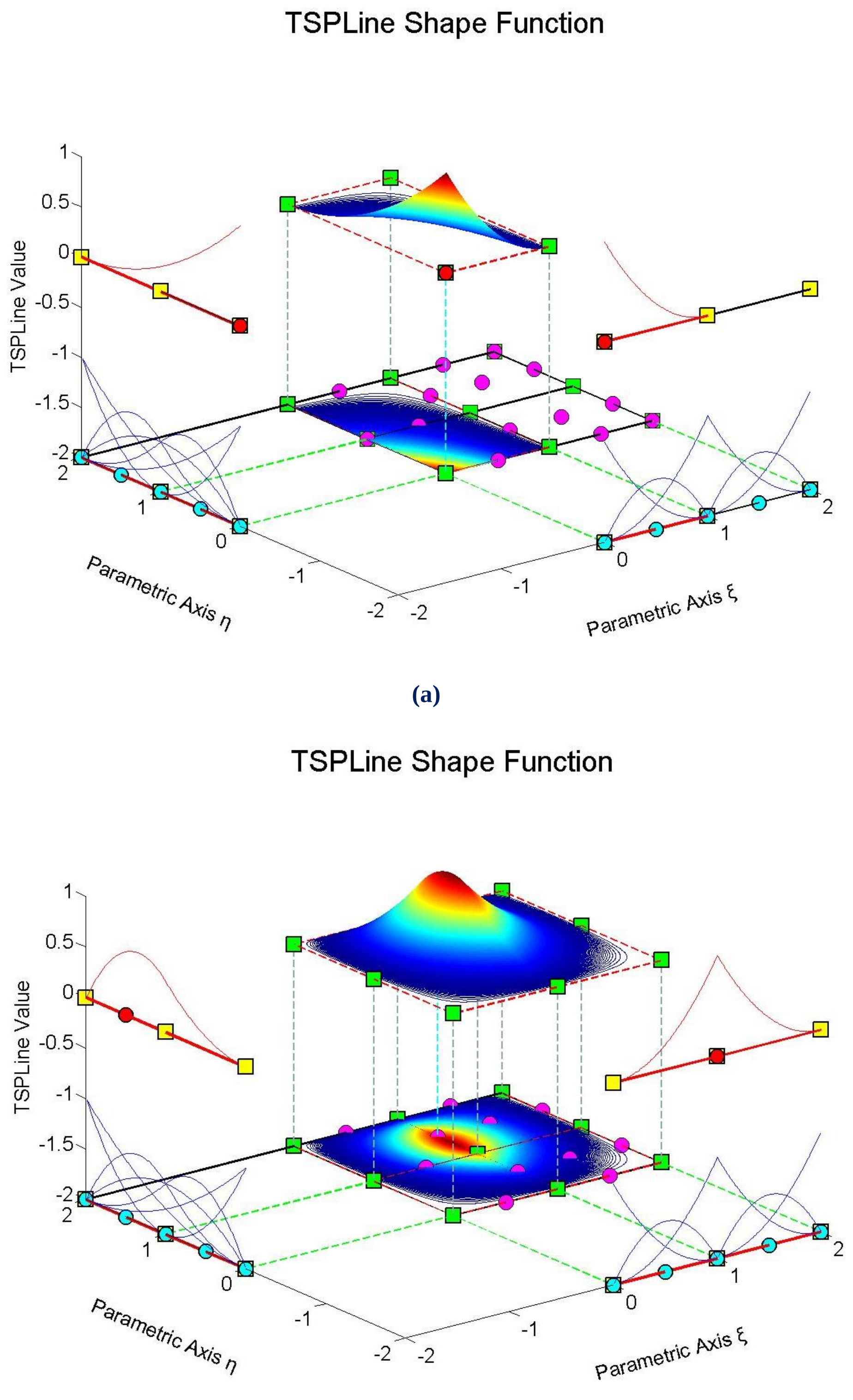


Figure 4.2.
Index space. (Created with GeomIso)
Anchors are drawn as red circles.

Index space is chosen above, so that the quarter plate with the hole is depicted with the less possible control points. The global knot value vector for axis ξ is Ξ={0,0,1,1,2,2} and for axis η H={0,0,1,2,2}. We can see that, even for the coarsest mesh, a T-SPLine representation needs much less control points. This will become more obvious, when refined meshes will be shown as applications later on. For the inexperienced eye, the global knot value vectors might seem not to be open. Note that an open knot value vector is the one that whose boundary knot values are repeated p+1 times. So, if open knot value vector does not exist, the continuity on the boundary seems to be greater than C^{-1} and thus the control points of the boundary are not interpolatory to the object. In fact, this is not true, as this important aspect is no longer defined by global knot value vectors. It is the degree combination for both axes, which implies where anchors will lie and consequently their local knot value vectors define the continuity. Note that certain knot values are repeated. These ones create the reduced continuity effect. The following figures depict shape functions with varying local support due to T-junctions and shape functions with reduced continuity.



(a)

(b)

Figure 4.3. Shape functions. (Created with GeomIso)
(a) Biquadratic shape function has continuity of C^{-1} for both axes.
Local knot value vectors are Ξ_1={0,0,0,1} and H_1={0,0,0,2}.
(b) Eighth biquadratic shape function is of C^0 Continuity along ξ and greater along η.
Its domain of influence is much greater than the first function.
This difference is created due to the T-Junction.

T- Splines do not own the full tensor product property. In Figure 4.4, parameter space is divided into three elements, which have different size. This is a property, which does not exist in NURBS, where all elements have the same size in parameter space. In T-SPLines, the various junction types allow this intricate junction interlocking. This procedure permits geometry design with less control points than in NURBS.

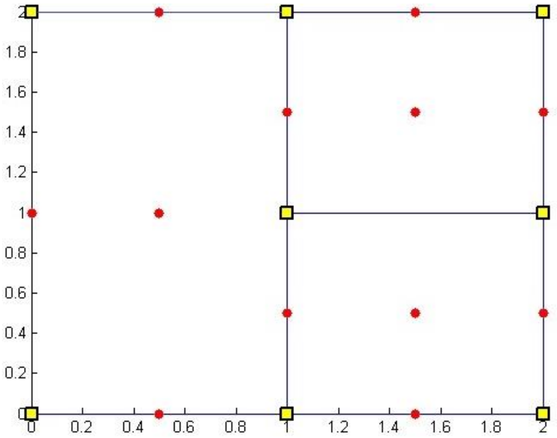
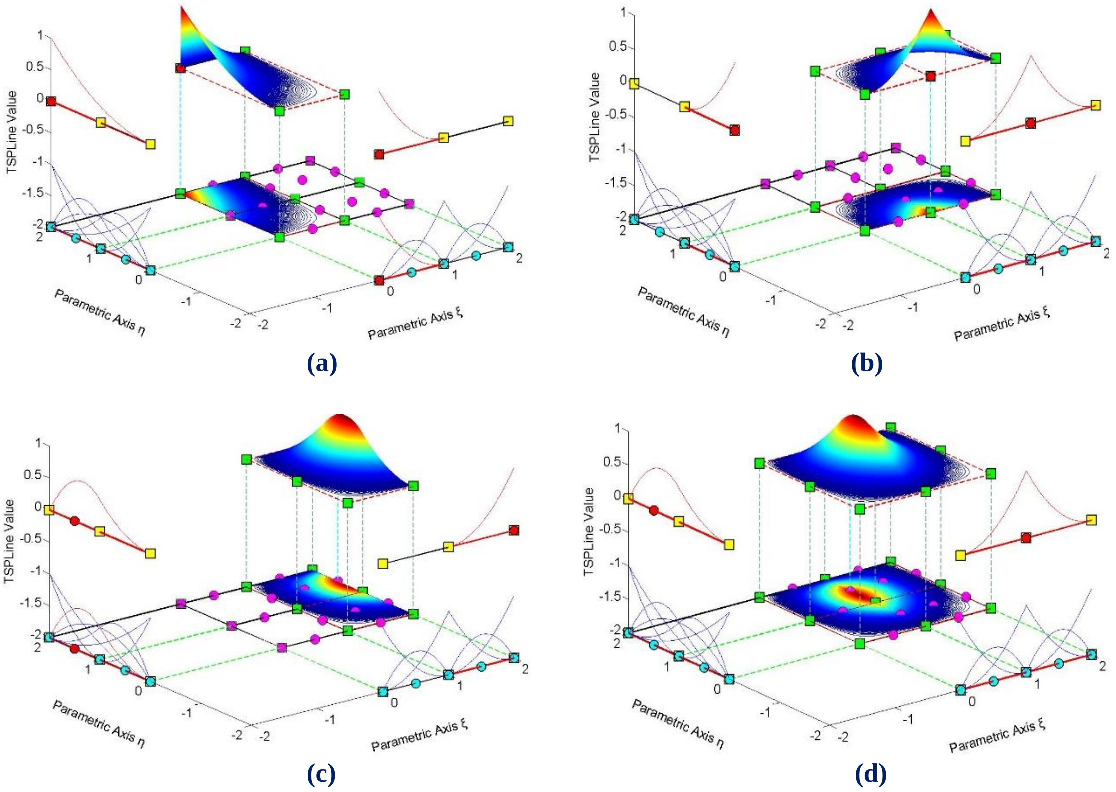


Figure 4.4. Parameter Space. (Created with GeomIso)
Anchors red circles and knots as yellow squares.
Anchors that lie on the knot lines have reduced continuity C^{-1} or C^{0}.



(a)

(b)

(c)

(d)

Figure 4.5. Parameter space. (Created with GeomIso)
Shape function on the boundary have reduced continuity.

Figure 4.6 represents stress σ_{xx} contour. Model is subjected to tensional axial stress. The applied constraints bound displacement Y for the edge parallel to ξ and X for the edge parallel to X. Note that, stresses are not continuous everywhere in the model, as a result of C^0 continuity blending functions.

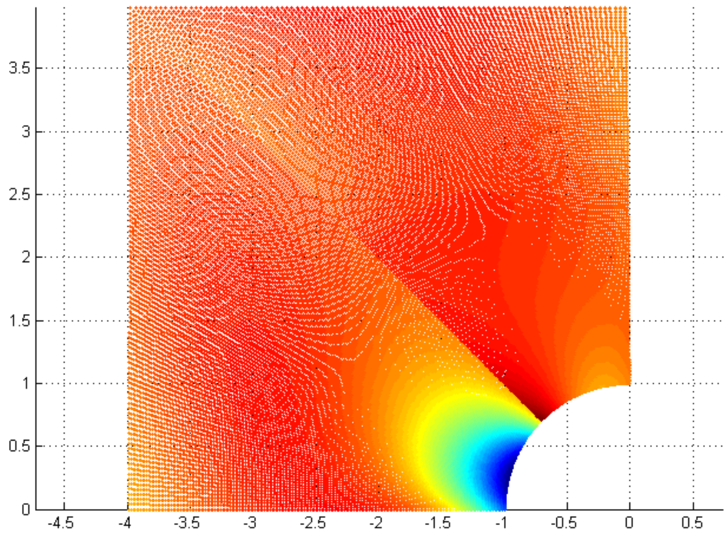
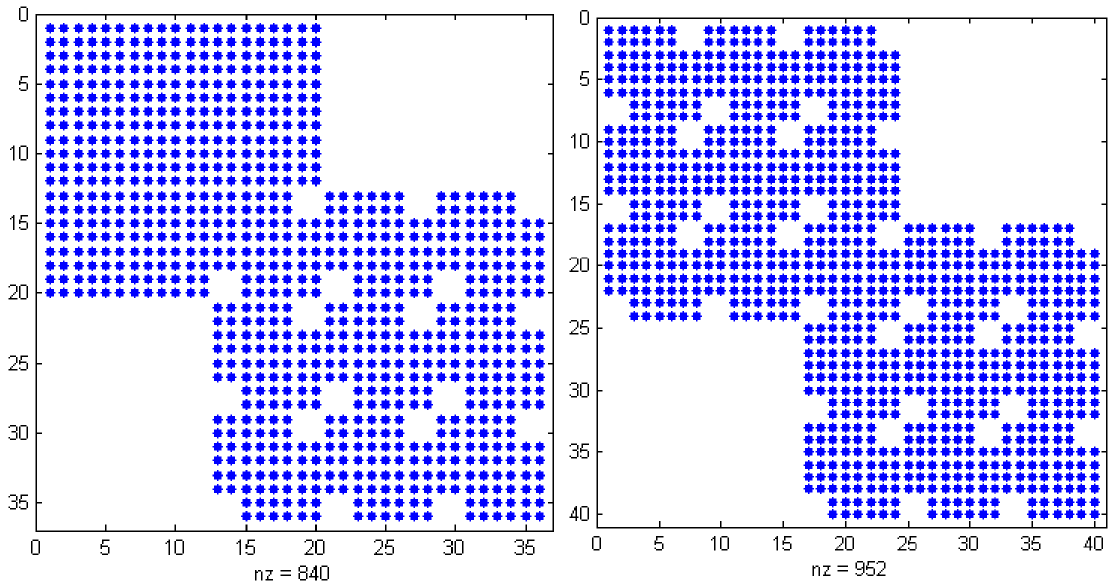


Figure 4.6. Physical space. Stress $\sigma_{\chi\chi}$ contour. (Created with GeomIso)
Shape function of reduced continuity separates the domain in two subdomains, that have continuous stress field.



(a) **(b)**

Figure 4.7. Plate with a hole. Stiffness matrix. (Created with GeomIso)
(a) Coarsest mesh (analyzed with T-SPLines). **(b)** Analyzed with NURBS.

Figure 4.7 depicts a comparison of these coarse meshes for NURBS and T-SPLines. Note that, even for these meshes, there is a difference between degrees of freedom. T-SPLine stiffness matrix is denser in the part where one element exists. Overlapping exists only in the C^0-continuous function.

4.2. Cook's Cantilever

On contrary to simple cantilever, cross section area of Cook's cantilever is not constant. This type of cantilever is quite popular in computational mechanics, because its shape proves ideal for testing numerical methods. Figure 4.8 shows this model in physical space, where its four edgy material points have coordinates (0,0), (0,48), (44,48) and (44,60).

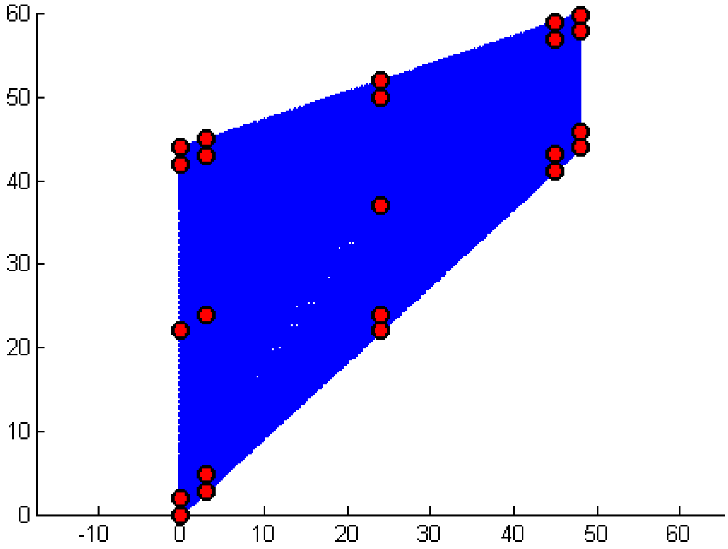
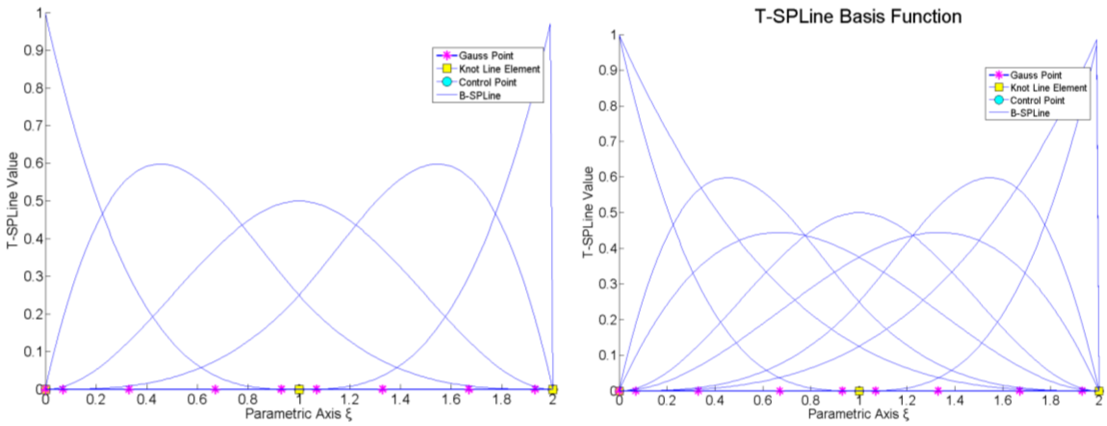


Figure 4.8. Cook's cantilever. Physical space. (Created with GeomIso)

Cook's cantilever is subjected to transverse bending. Loads are applied on the right edge and have direction towards the negative Cartesian axis Y. The left vertical edge is bounded. In this way, all displacements along this edge are limited to zero. All shape functions are smooth and there are no patches in the internal of the domain.



(a) **(b)**

Figure 4.9. T SPLine blending functions.
(a) Axis ξ **(b)** Axis η
(Created with GeomIso)

Figure 4.10 shows the initial index mesh of Cook's cantilever. It contains 14 index cells. T-SPLine blending functions are bi-cubic, so anchors lie on the vertices and their number is equal to 13.

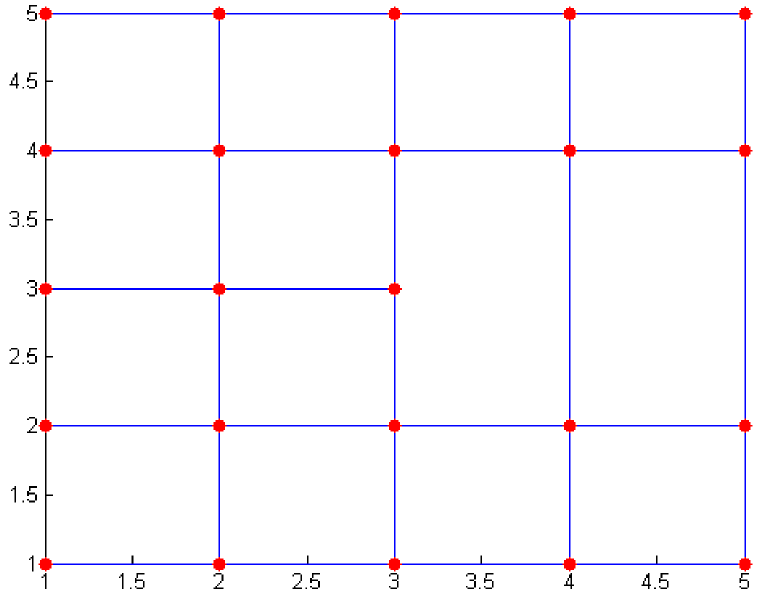


Figure 4.10. Cook's cantilever. Index space. (Created with GeomIso) Anchors (red circles) lie on the vertices due to odd polynomial degree.

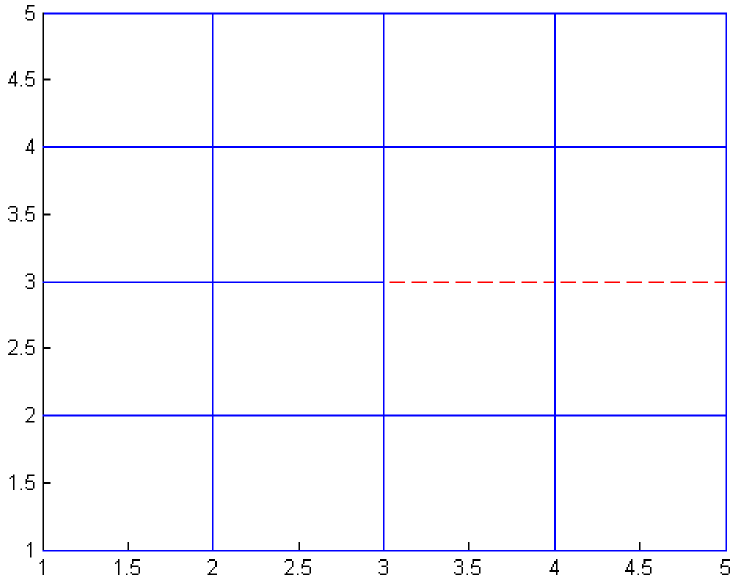


Figure 4.11. Cook's cantilever. (Created with GeomIso) Continuity reduction lines (red dashed) divide further index space in order to envelop new integration mesh.

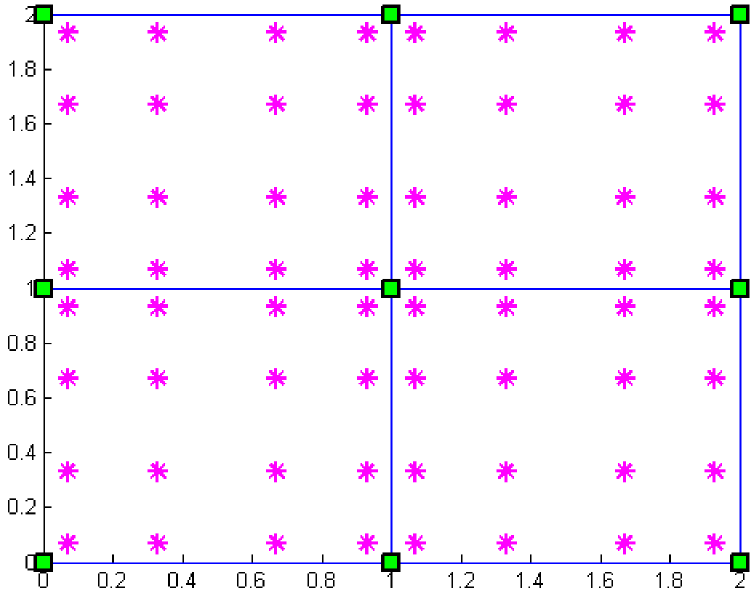


Figure 4.12. Cook's Cantilever.
Parameter space constitutes of 3 elements and 4 integration cells.
Gauss points are located within the integration cells.
(Created with GeomIso)

Figure 4.12 depicts stress filed of $\sigma_{\chi\chi}$. Significant stress concentration occurred on the right edge. In order to imporve the results' accurasy, the right domain of the model was refined.

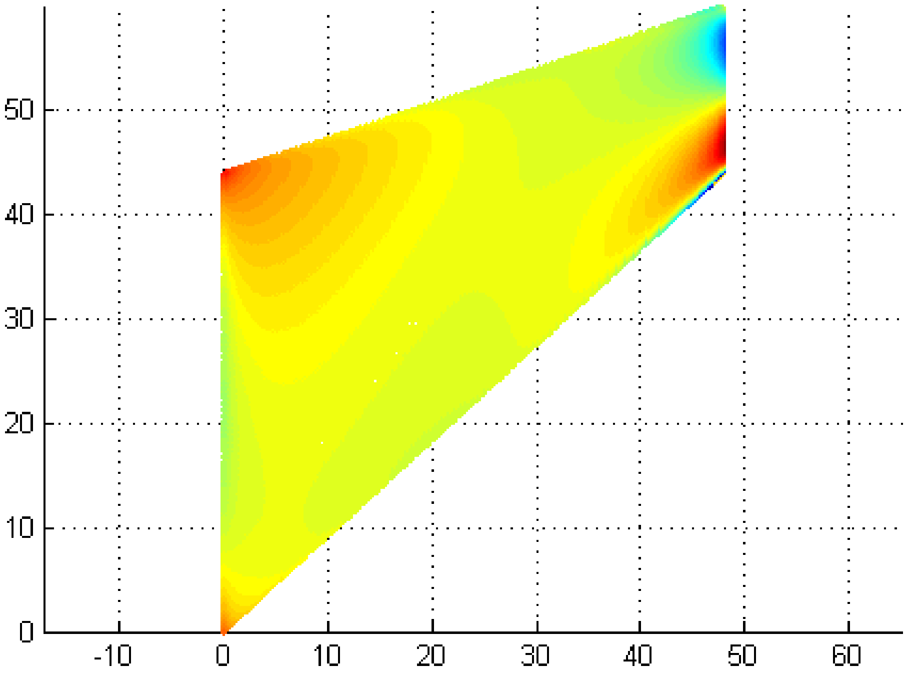
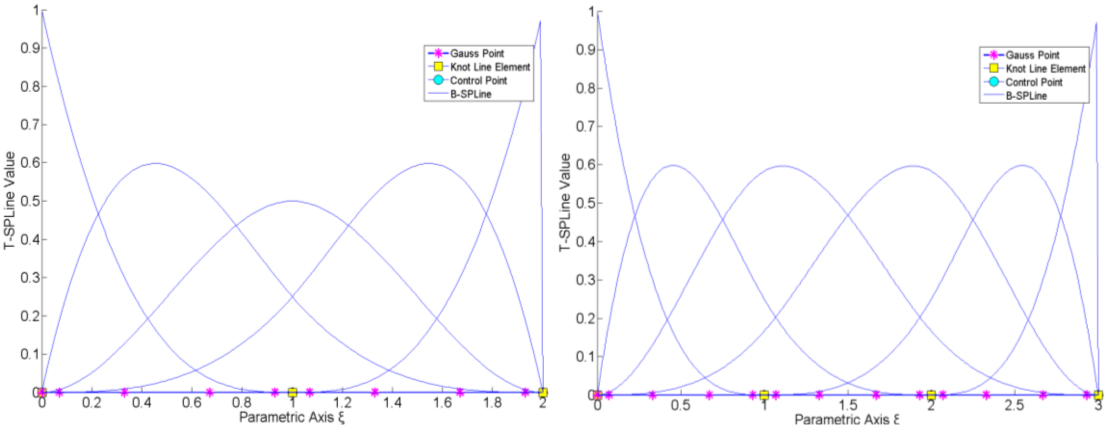


Figure 4.13. Cook's cantilever. (Created with GeomIso)
Stress $\sigma_{\chi\chi}$ contour.

The mesh is refined (from 23 initial control points to 30 final), by subdividing the initial left element to four new ones.

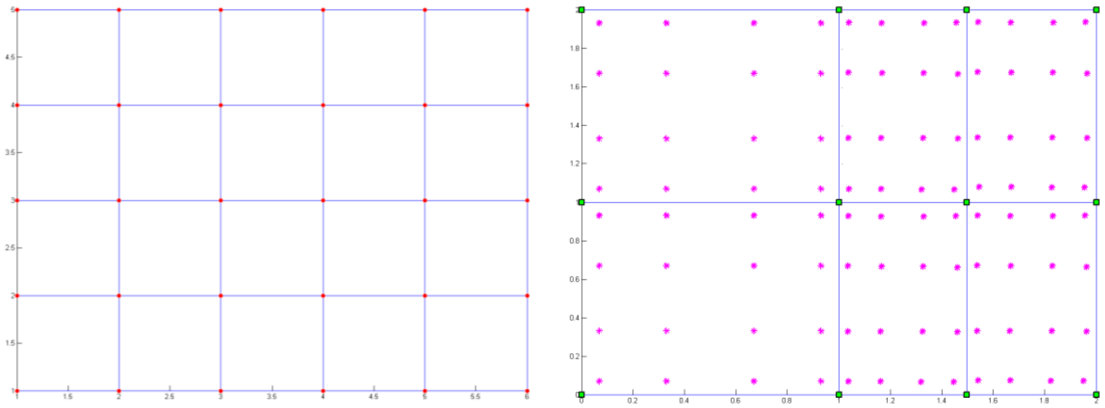


(a)

(b)

Figure 4.14. Cook’s cantilever. Parameter space.
(a) Refined T-SPLine blending functions (axis ξ).
(b) Refined T-SPLines blending functions (axis η).
(Created with GeomIso)

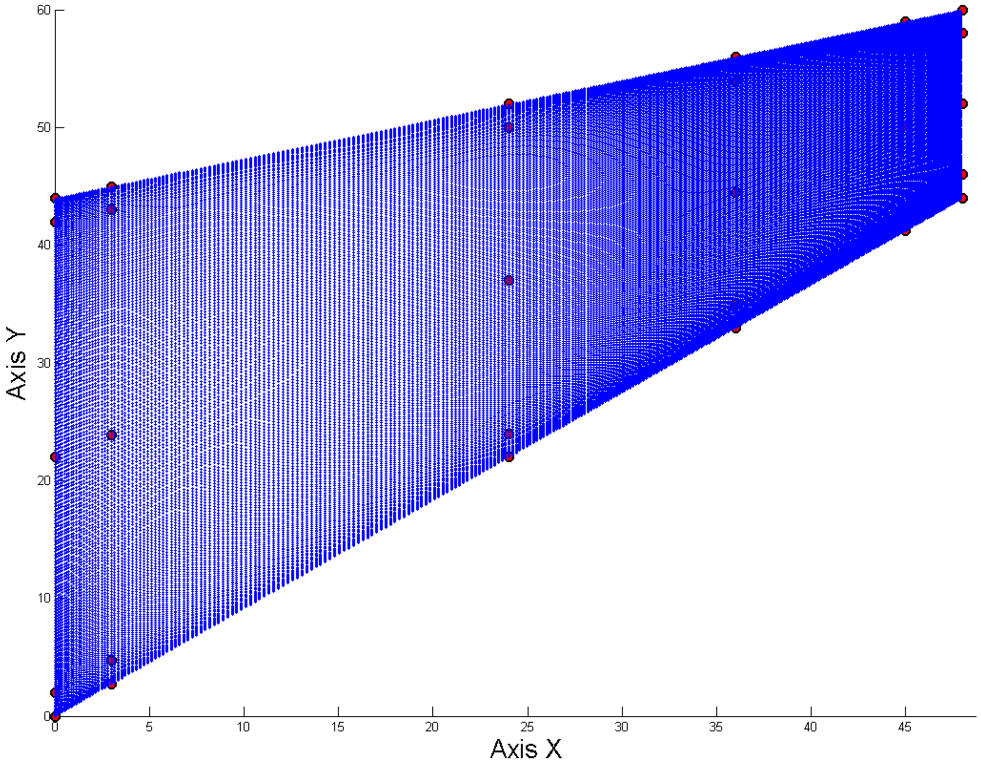
Blending functions of the refined mesh are similar to those of B-SPLines due to mesh interconnection. This refinement was chosen, as the previous one did not result in accurate stress field on the loaded edge and contained quite few elements. Index space is shown below with 20 index cells and parameter space with six integration elements. Stresses become now smoother and displacements have increased as expected.



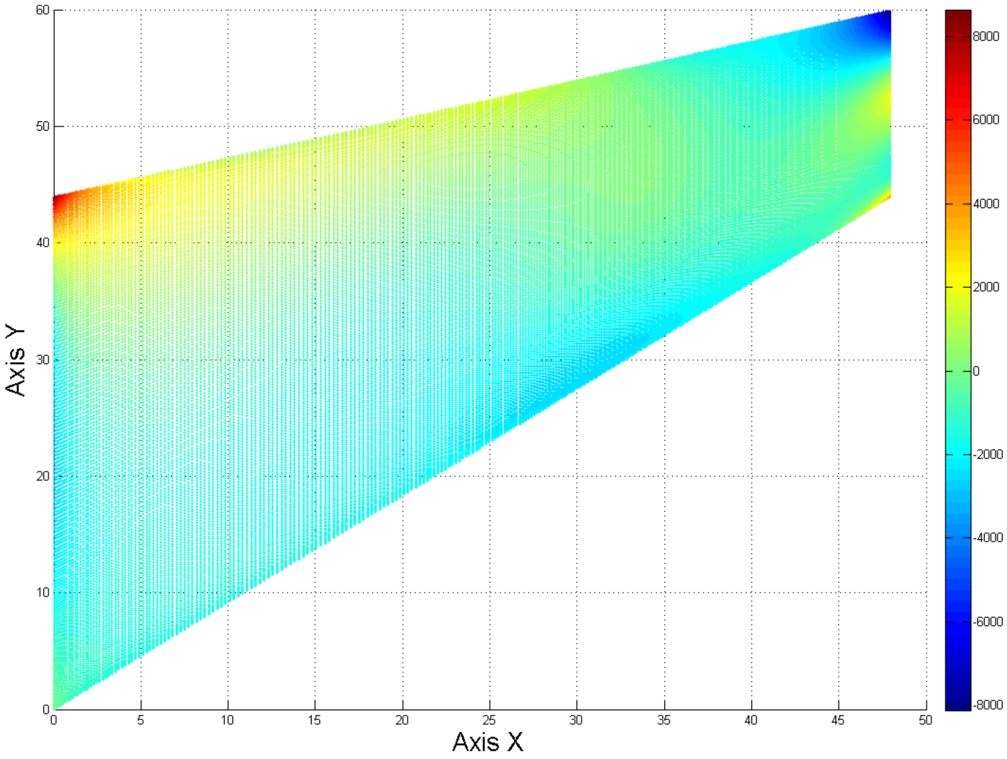
(a)

(b)

Figure 4.15. Cook’s cantilever.
(a) Index space with anchors as red circles.
(b) Parameter space with Gauss points as magenta stars and knots as green squares.
The initial left element is divided into four new ones.
(Created with GeomIso)



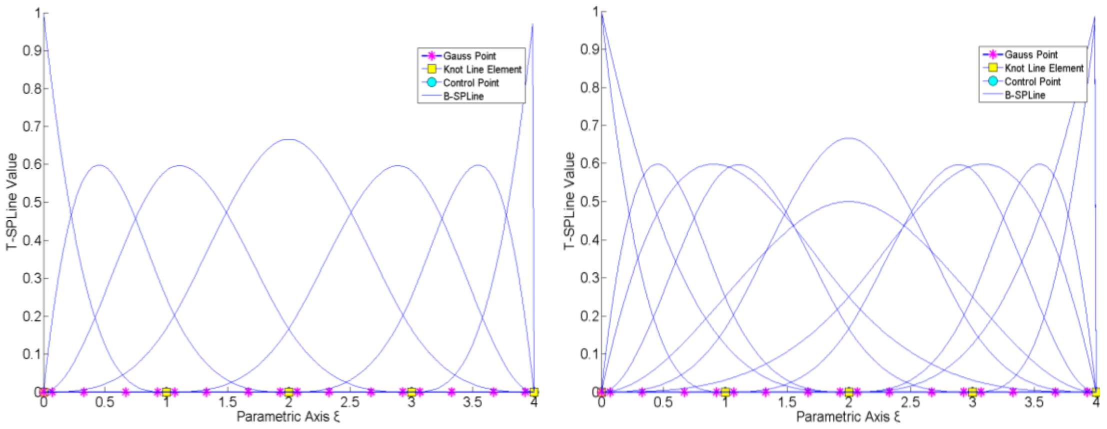
(a)



(b)

Figure 4.16. Physical space. (Created with GeomIso)
(a) Physical space with control points as red circles (after first subdivision).
(b) Stress σ_{xx} contour (after first subdivision).

The mesh is refined again (from 30 control points to 43 final), by subdividing each of the four new elements of previous refinement to four smaller ones.

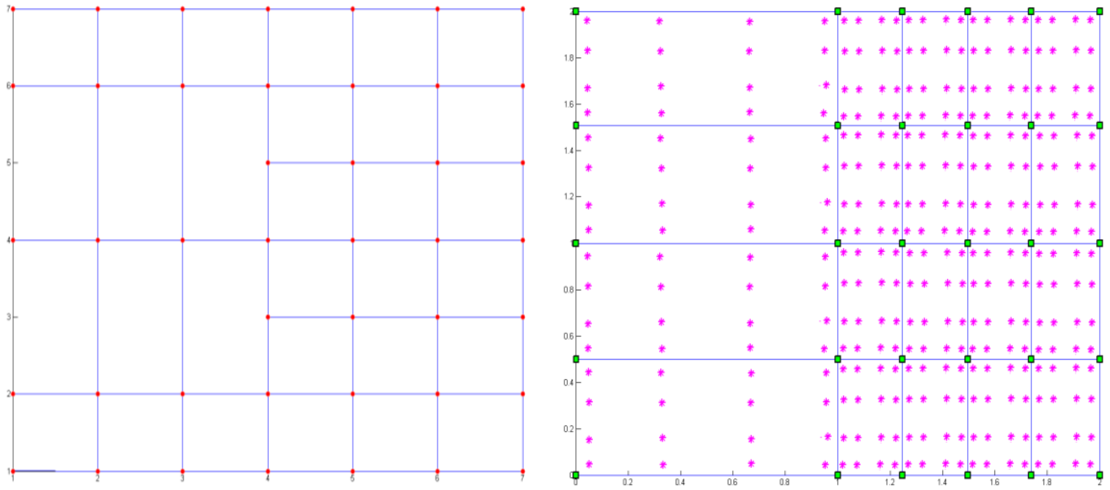


(a)

(b)

Figure 4.17. Cook’s cantilever.
(a) Refined T-SPLine blending functions (axis ξ).
(b) Refined T-SPLine blending functions (axis η).
(Created with GeomIso)

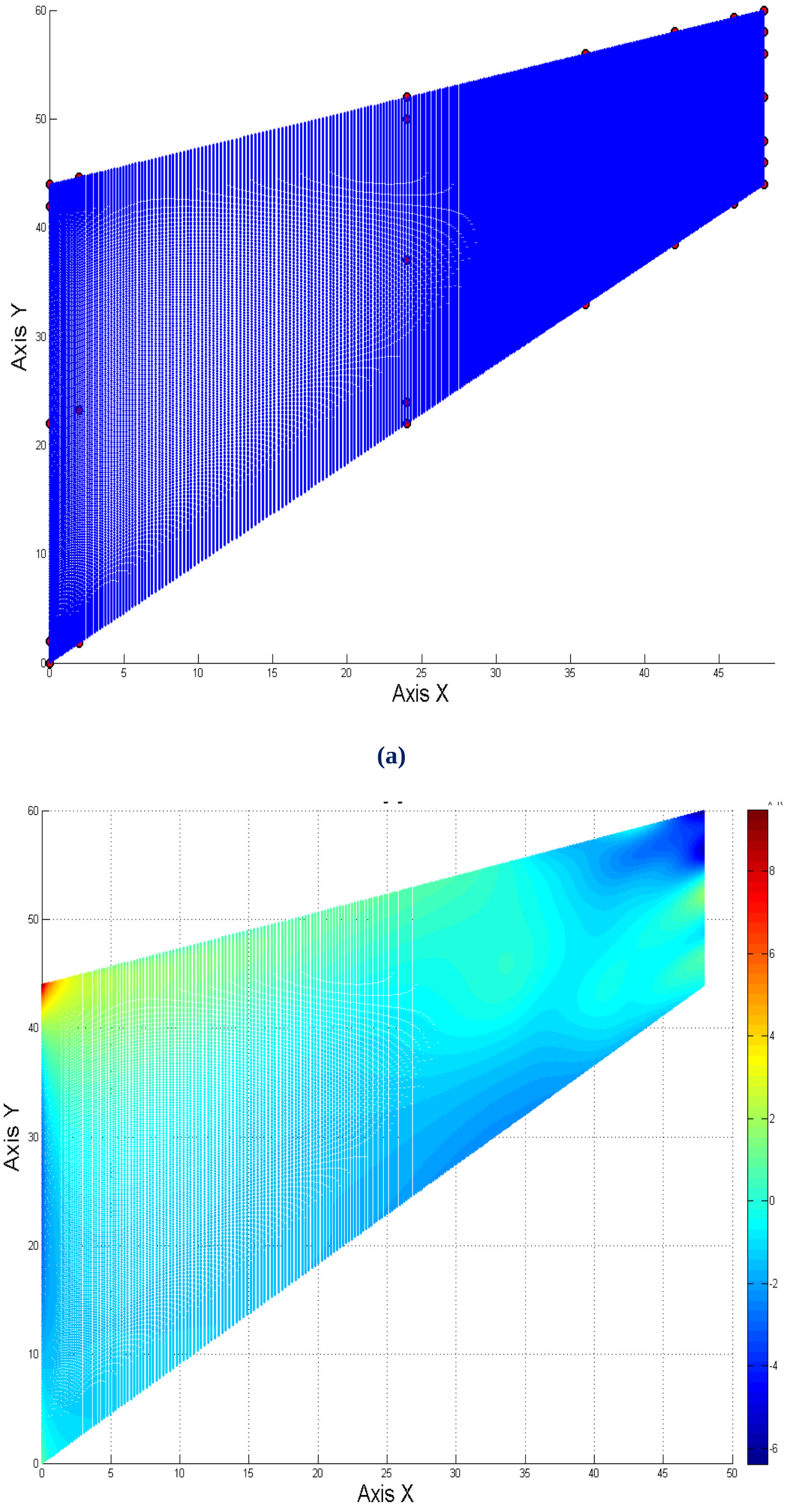
Blending functions of the refined mesh are similar to those of B-SPLines due to mesh interconnection. This refinement was chosen, as the previous one did not result in accurate stress field on the loaded edge and contained quite few elements. Index space is shown below with 30 index cells and parameter space with 20 integration elements. Stresses become now smoother and displacements have increased.



(a)

(b)

Figure 4.18. Cook’s cantilever.
(a) Index space with anchors are red circles.
(b) Parameter space with 20 elements as blue rectangles, 30 knots as green squares and 320 Gauss points as magenta stars.
(Created with GeomIso)



(a)

(b)

Figure 4.19. Physical space. (Created with GeomIso)
(a) Physical space with control points as red circles (after second refinement level).
(b) Stess σ_{xx} contour (after second refinement level).

It is crucial to remind the fact that a refined T-Mesh requires much less computations than the equivalent NURBS one. The following mesh is suitable for both NURBS and T-SPLines. This mesh was refined in order to examine the efficiency of local T-SPLine refinement.

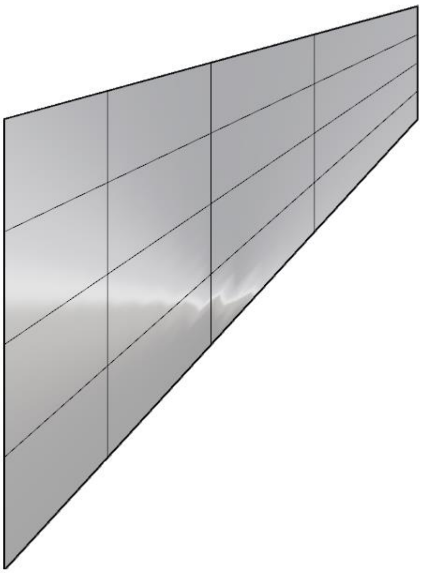
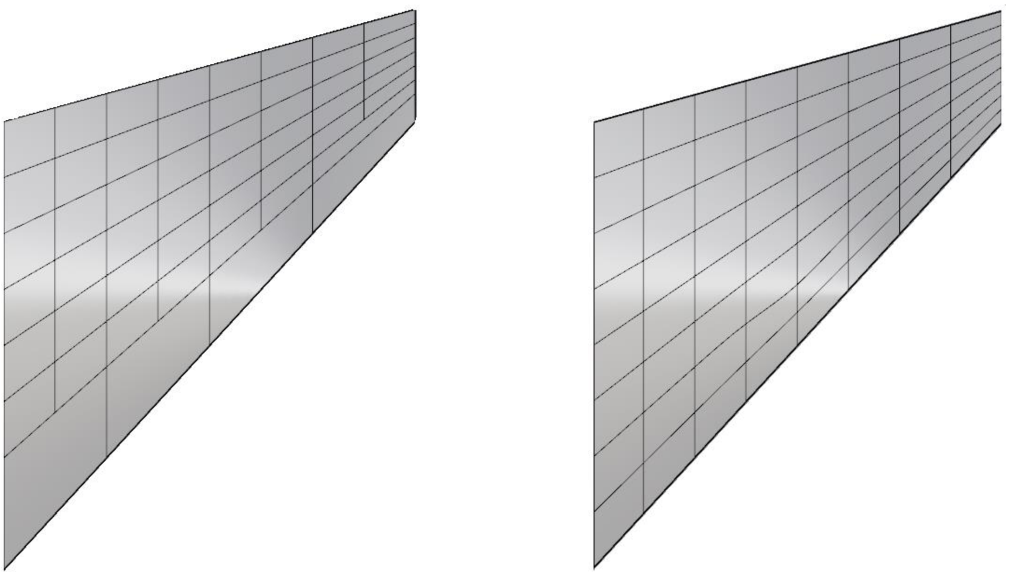


Figure 4.20. Physical space. Exact geometry mesh.
(Designed by Tsapetis Dimitrios)

The initial number of control points is 25 in both case (NURBS, T-SPLines). The area, that requires refinement, seems to be the loaded edge, where consecutive refinements are applied.



(a) **(b)**

Figure 4.21. First refinement level.
(Designed by Tsapetis Dimitrios and analyzed with GeomIso)

After first refinement level, it is obvious that there are major differences between NURBS and T-SPLine meshes with the same odd degree. T-Mesh requires 67 control points, while the equivalent NURBS one is increased by 20% to 81.

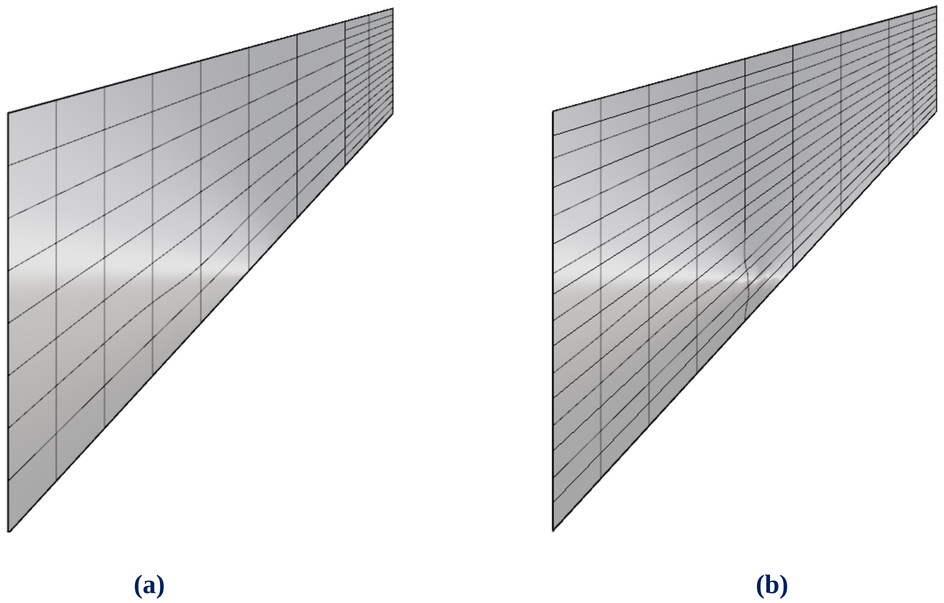


Figure 4.22. Second refinement level.
(Designed by Tsapetis Dimitrios and analyzed with GeomIso)

For the second refinement level (same polynomial degree for both axes), benefits of T-SPLines are more clear. T-Mesh requires 114 control points, while the equivalent NURBS ones is increased by 35% to 153. It is decided to continue for one last refinement step and then compare the results.

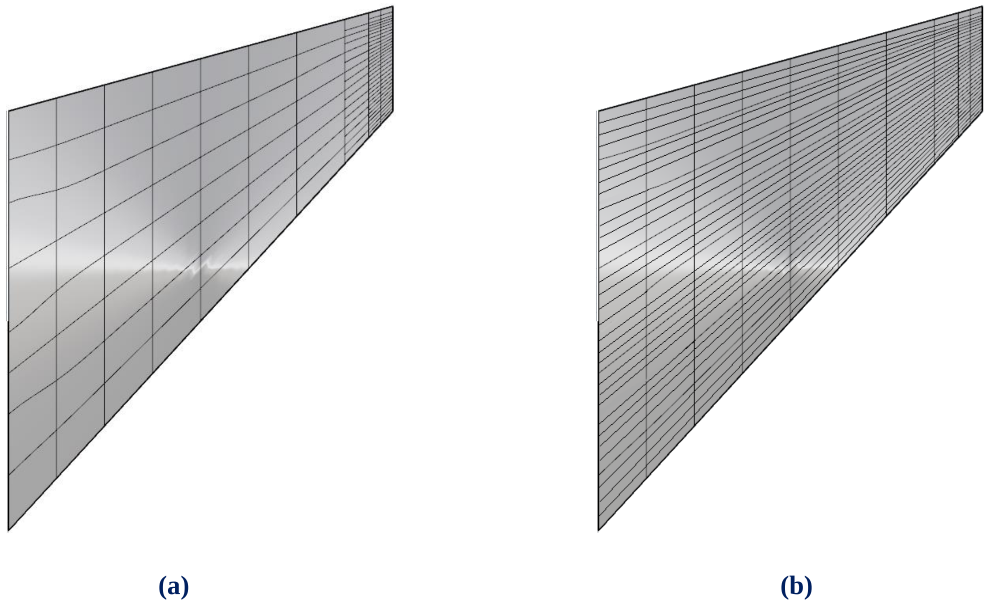


Figure 4.23. Third refinement level.
(Designed by Tsapetis Dimitrios and analyzed with GeomIso)

For the third refinement level, the major "gap" between T-SPLines and NURBS is obvious. T-Mesh requires 179 control points, while the equivalent NURBS mesh 363. The increased percentage has now skyrocketed to 100%.

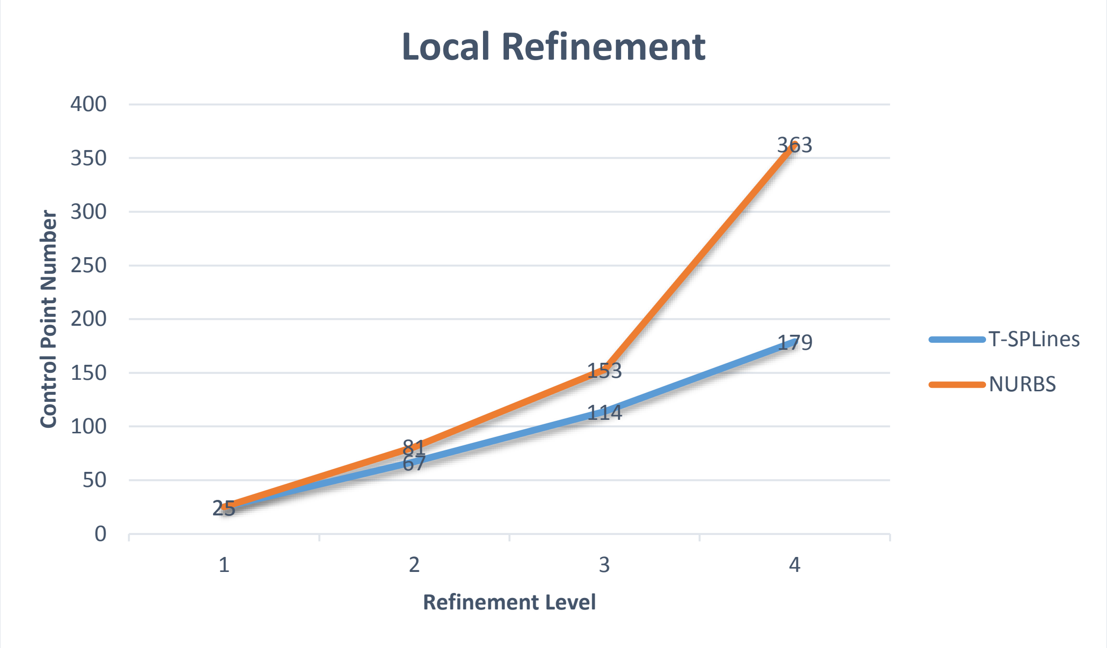


Figure 4.24. Local refinement.
Control point number per refinement level.
Comparison between NURBS and T-SPLines.
(GeomIso Results)

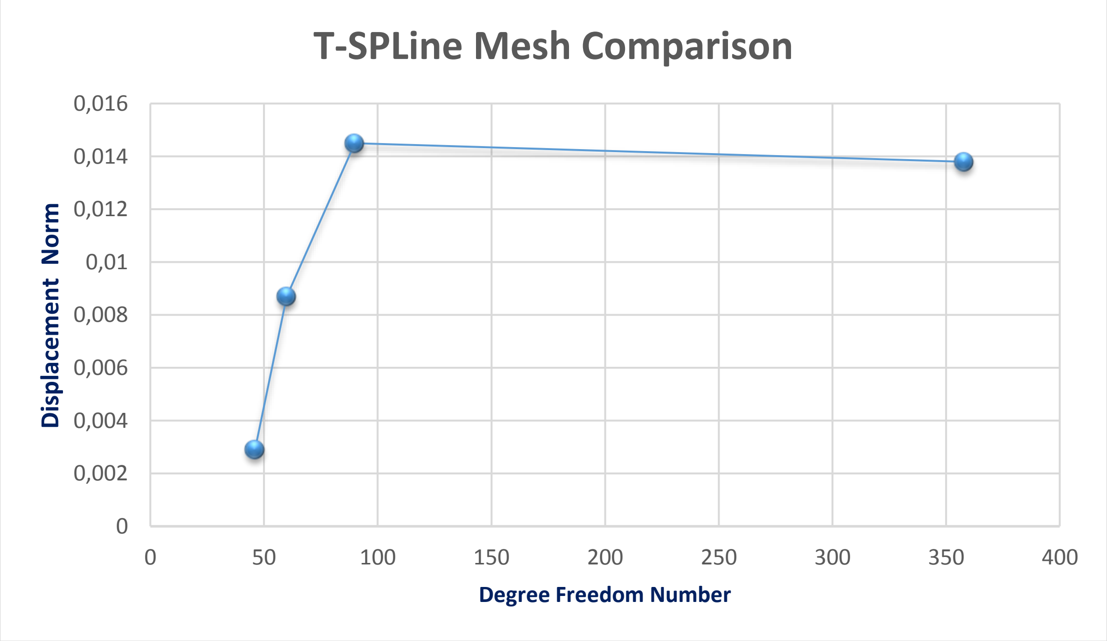


Figure 4.25. Local refinement.
Displacement norm according to degree freedom number.
Parametric investigation.
(GeomIso Results)

Figure 4.26 depicts the comparison between two different T-SPLine meshes. As shown, the real displacement of the examined control point is reached with only few control points. Specifically, if the mesh with 358 degrees of freedom is considered as the accurate solution, then, with only one third of the degrees of freedom required the desired solution was reached. The error of the solution with 90 degrees of freedom is only 5% compared with the accurate one. Below, the corresponding features of fine mesh are available.

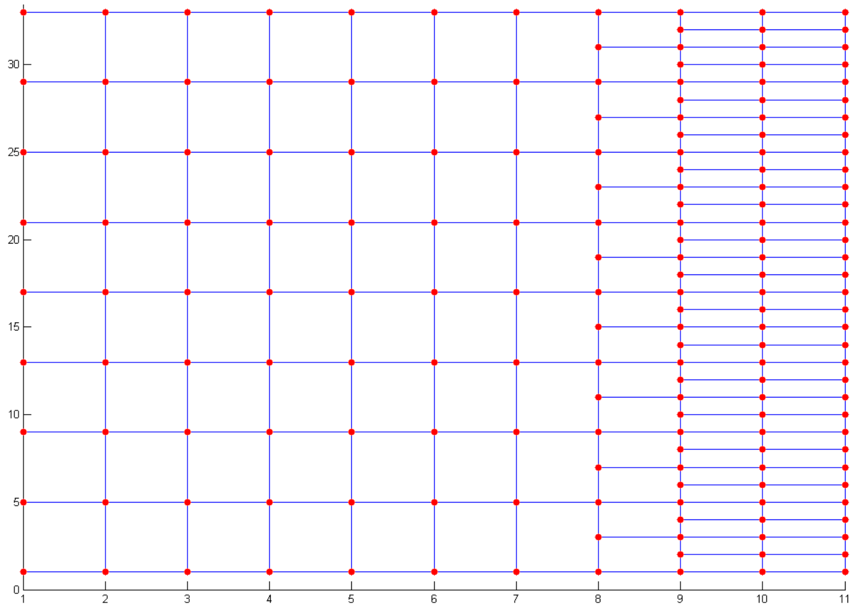


Figure 4.26. Index space.
(Created with GeomIso)

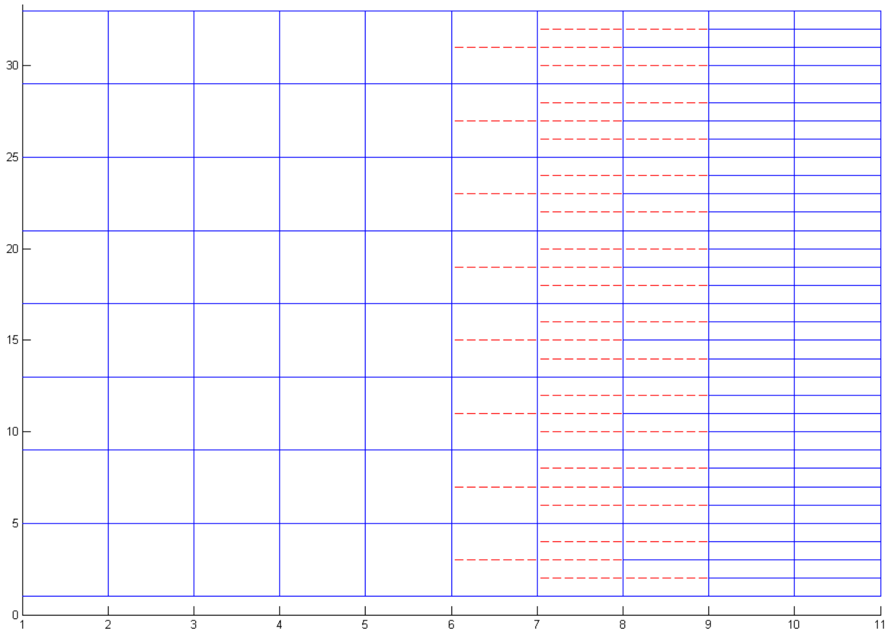


Figure 4.27. Continuity reduction lines.
(Created with GeomIso)

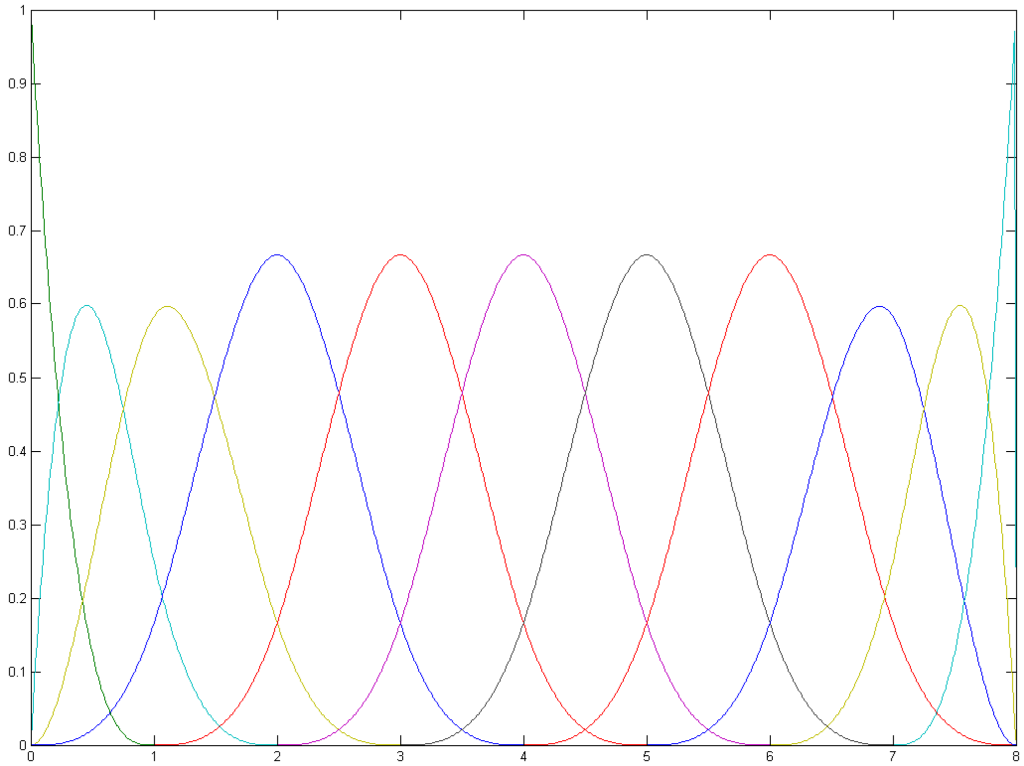


Figure 4.28. T-SPLine blending functions (ξ).
(Created with GeomIso)

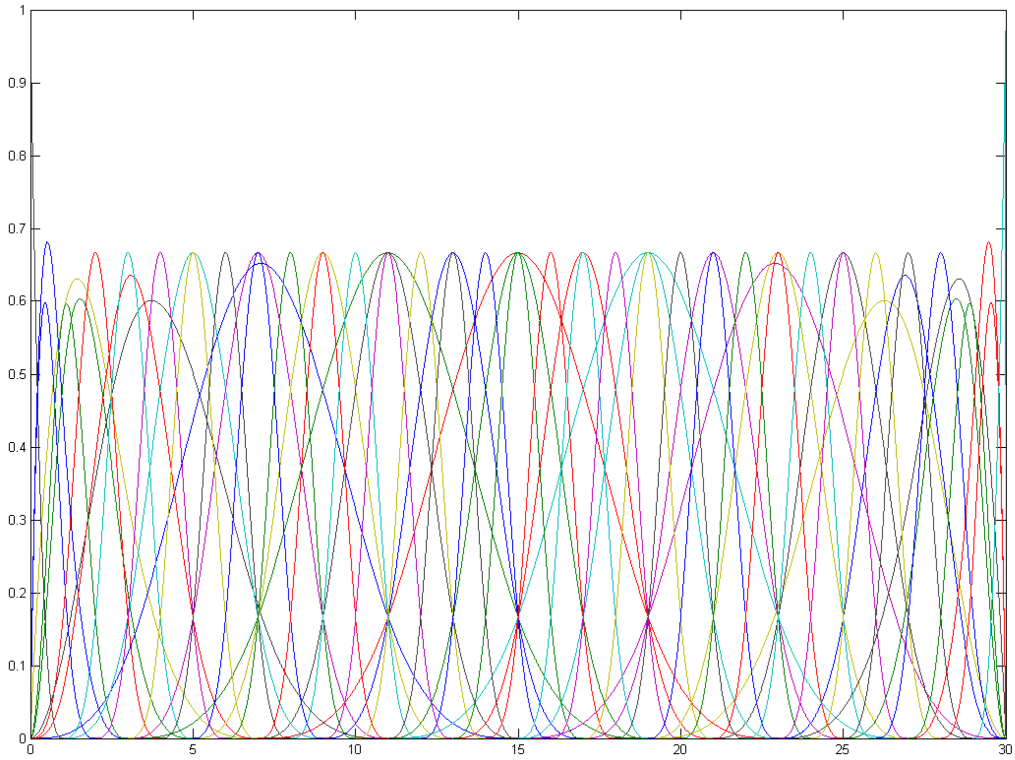
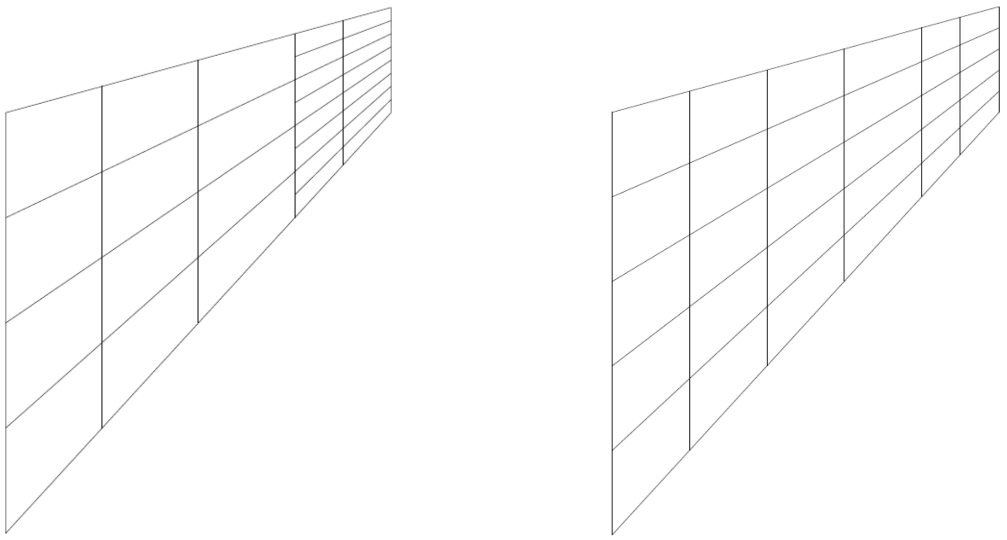


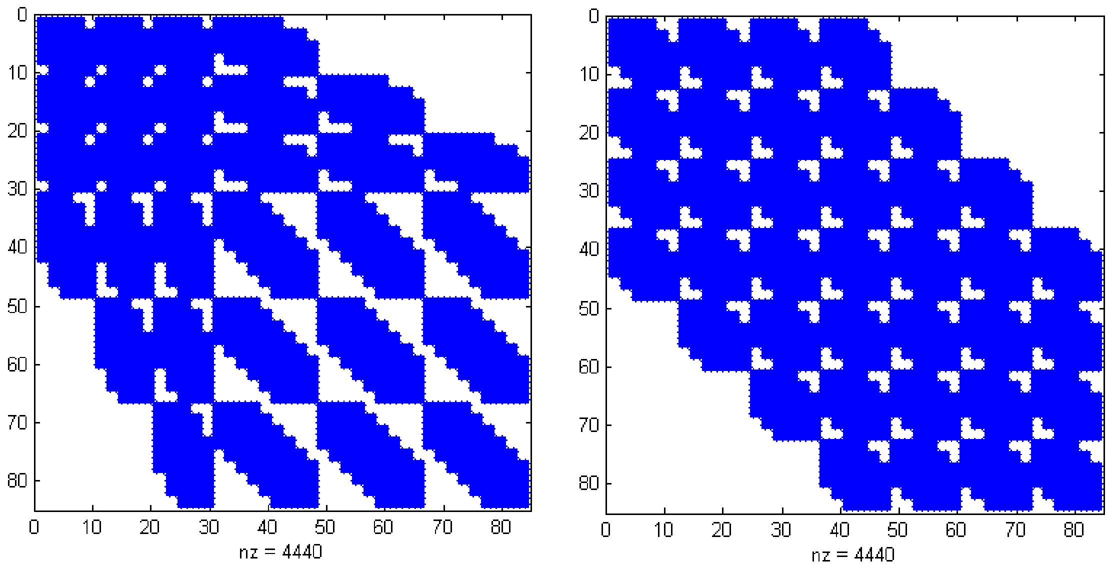
Figure 4.29. T-SPLine blending functions (η).
(Created with GeomIso)

As far as Cook's cantilever is concerned, T-SPLine and NURBS models were compared in terms of accuracy, degree freedom number and stiffness matrix's bandwidth.



(a) **(b)**

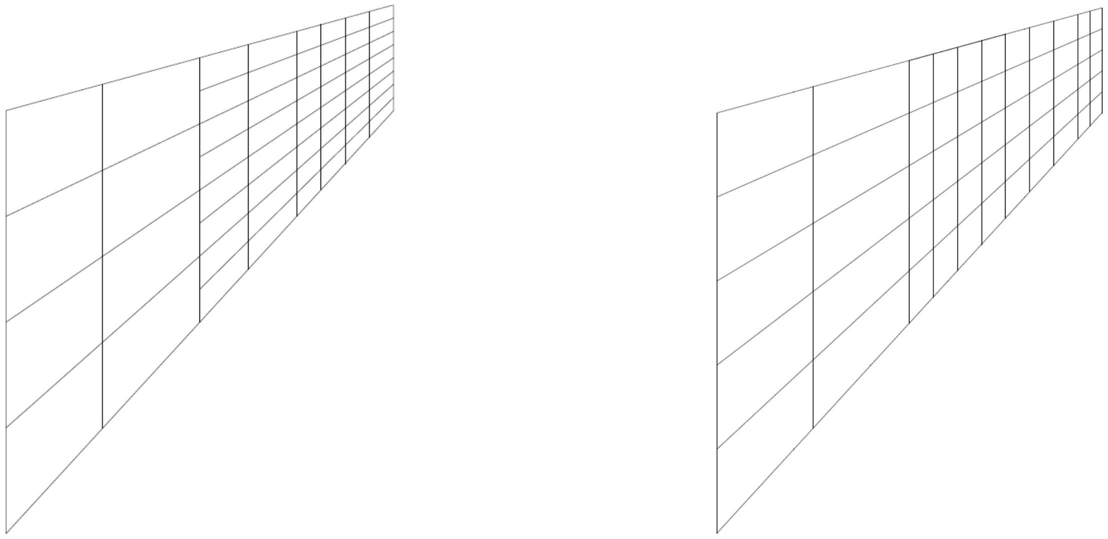
Figure 4.30. Fist refinement level.
Comparison between **(a)** T-SPLines and **(b)** NURBS.
Equivalent meshes in terms of refinement strategy
with the same degree freedom (84) and control point (42) number.
(Created with GeomIso)



(a) **(b)**

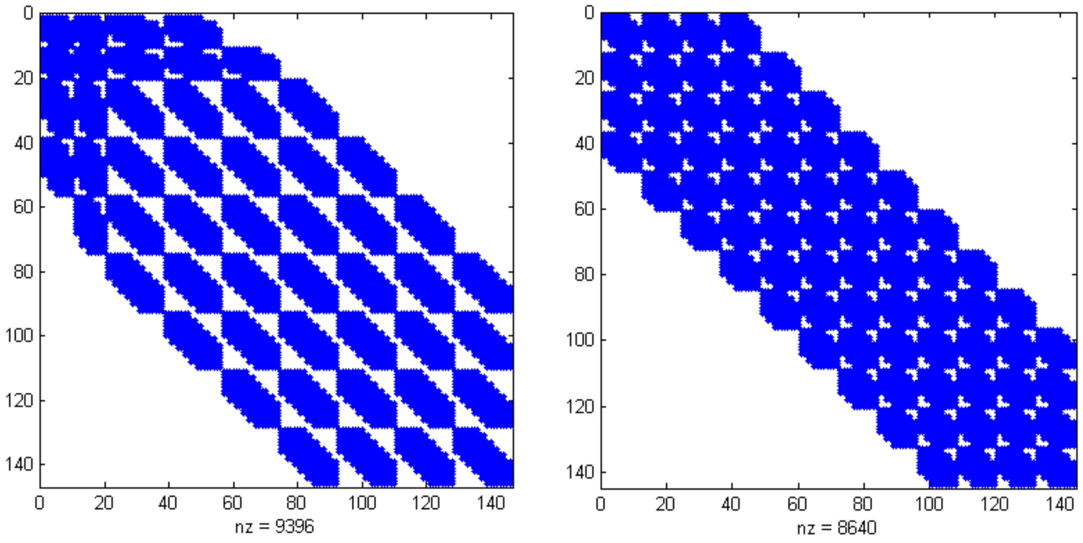
Figure 4.31. Fist refinement level.
Stiffness matrix comparison between **(a)** T-SPLines and **(b)** NURBS.
NURBS' full tensor product nature provides a steady overlapping and bandwidth.
On the contrary, T-SPLines have a varying bandwidth and increased accuracy.
(GeomIso Results)

The second stage of refinement creates similar results, as shown by the figures below. In both T-SPLines and NURBS, the control point number was chosen to be the same.



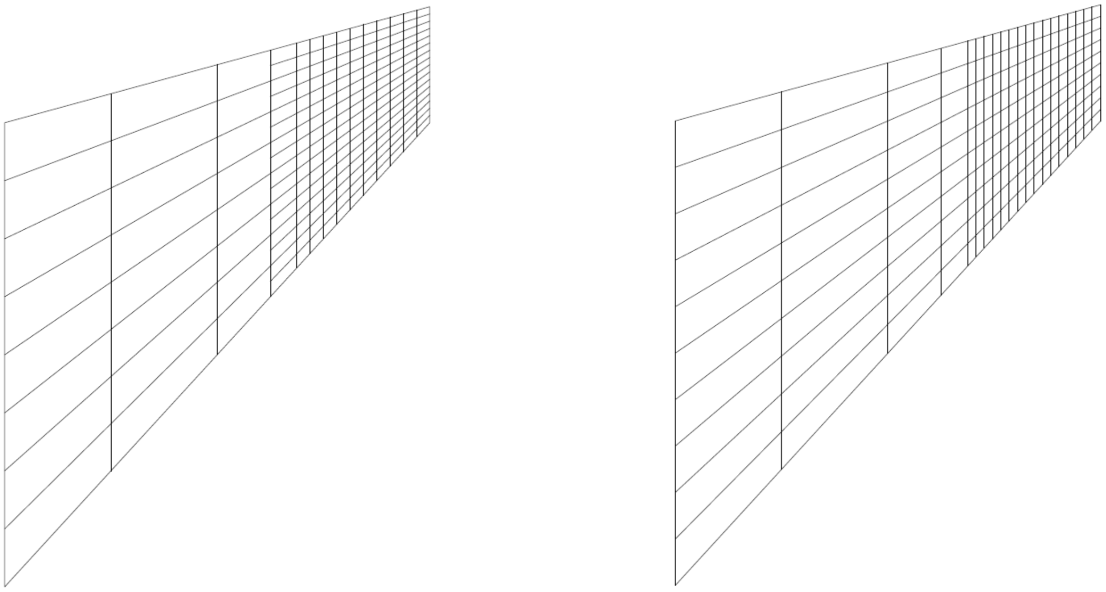
(a) **(b)**

Figure 4.32. Second refinement level.
Comparison between **(a)** T-SPLines and **(b)** NURBS.
Equivalent meshes in terms of refinement strategy
with the same degree freedom (144) and control point (72) number.
(Created with GeomIso)



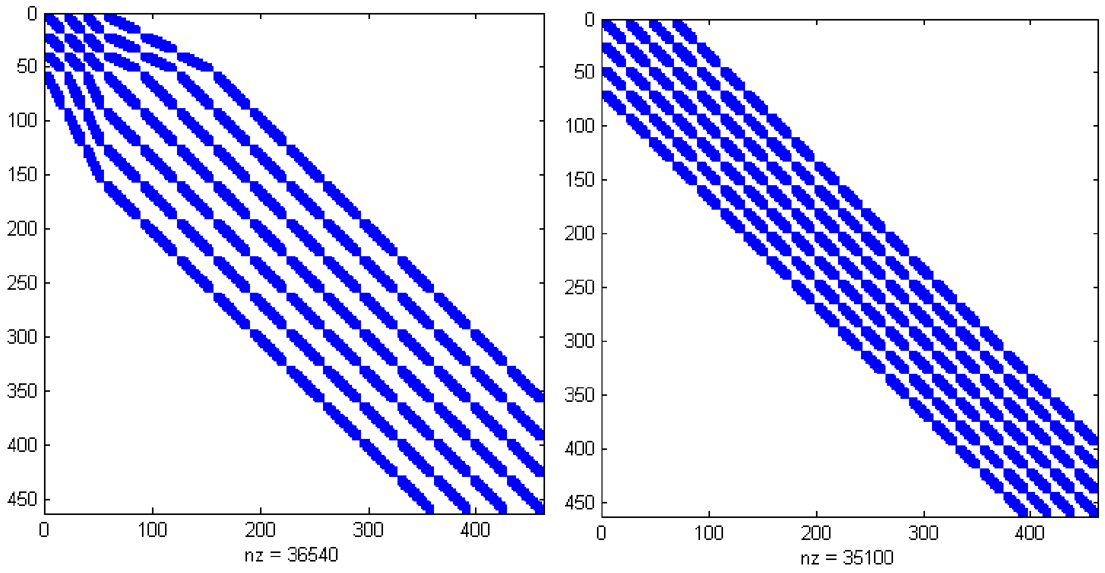
(a) (b)

Figure 4.33. Second refinement level.
Stiffness matrix comparison between **(a)** T-SPLines and **(b)** NURBS.
NURBS' full tensor product nature provides compact overlapping and steady bandwidth.
On the contrary, T-SPLines have a varying bandwidth and improved accuracy.
(GeomIso Results)



(a) **(b)**

Figure 4.34. Third refinement level.
Comparison between **(a)** T-SPLines and **(b)** NURBS.
Equivalent meshes in terms of refinement strategy
with the same degree freedom (462) and control point (231) number.
(Created with GeomIso)



(a) **(b)**

Figure 4.35. Third refinement level.
Stiffness matrix comparison between **(a)** T-SPLines and **(b)** NURBS.
NURBS' full tensor product nature provides compact overlapping and steady bandwidth.
On the contrary, T-SPLines have a varying bandwidth and improved accuracy.
(GeomIso Results)

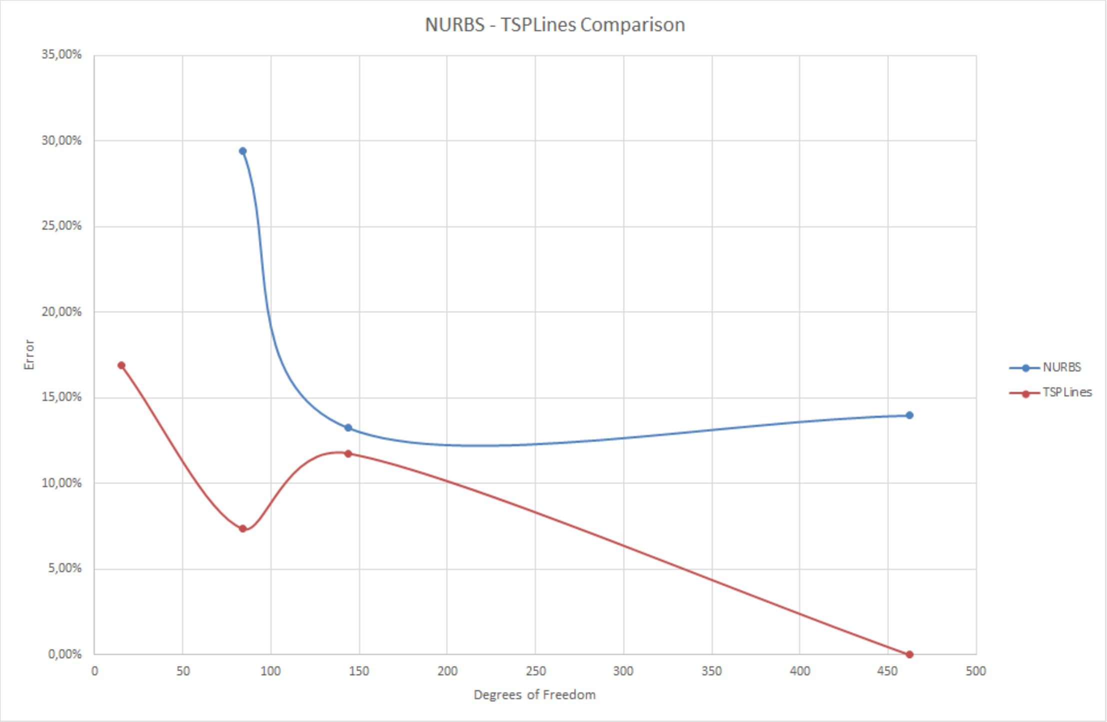


Figure 4.36.
Comparison between T-SPLines and NURBS. Error percentage.
T-SPLines provide increased accuracy and reduced error per degree of freedom.
(GeomIso Results)

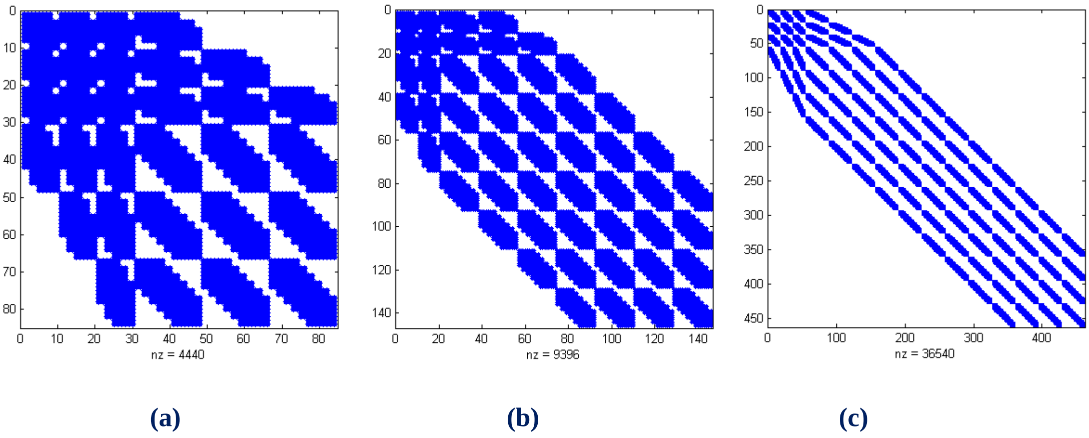


Figure 4.37.
Comparison between refinement levels of T-Mesh. Stiffness matrix.
NURBS exhibit smaller and compact bandwidth and steady overlapping.
T-SPLines do not own these properties, as they do not have full tensor product nature.

(a) First refinement level.

(b) Second refinement level.

(c) Third refinement level.

(GeomIso Results)

4.3. L-Shaped Domain

The L-shaped domain is a quite common geometry for researchers of computational mechanics worldwide. The applied external forces on the two (horizontal and vertical) sides tend to “open” the corner. This is similar to infinite stress concentration near a hole. A dense mesh was initially chosen near the edges of the opening corner, as the area, which was affected mostly, was known a priori.

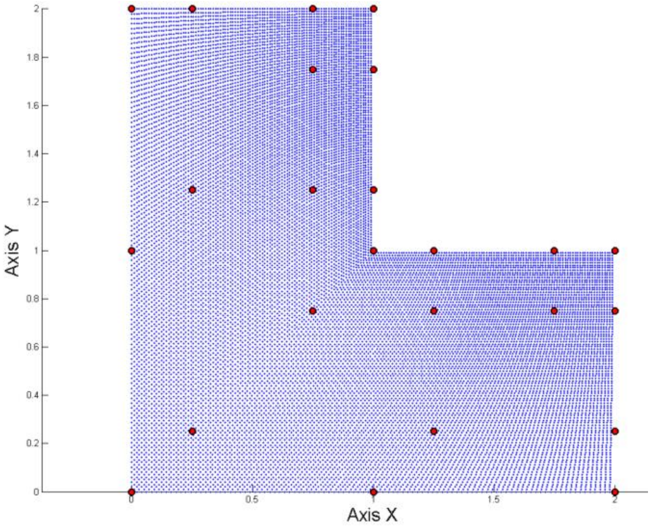


Figure 4.38. Physical space.
L-shaped domain with control points as red circles
and material points as blue dots.
(Created with GeomIso)

Figure 4.38 shows the physical space of an L-Shaped domain with 24 control points. It is obvious that only control points on the boundary and the diagonal of the corner are interpolatory to the surface.

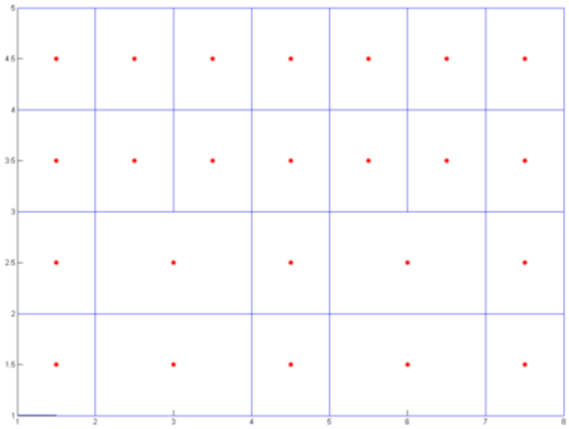


Figure 4.39. Index space of L-shaped domain with anchors as red circles.
(Created with GeomIso)

The initial index space was chosen in order to design the L-shaped domain with the minimum possible control points. The global knot value vectors are Ξ={0,0,1,1,2,2} and H={0,0,1,2,2}. We can see that, even for this coarse mesh, a T-SPLine representation requires much less control points than the equivalent NURBS model. This will become more obvious, when refined meshes will be shown as part of the application later on.

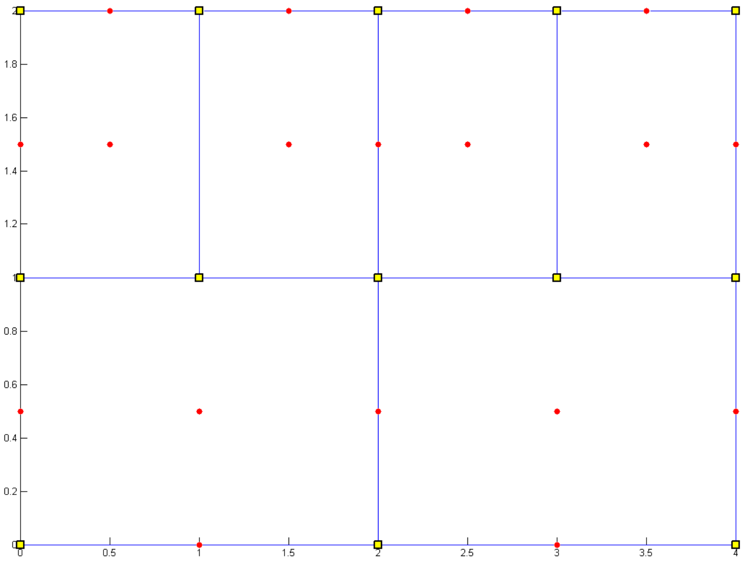


Figure 4.40. L-shaped domain. Parameter space.
Anchors as red circles and knots as yellow rectangles.
(Created with GeomIso)

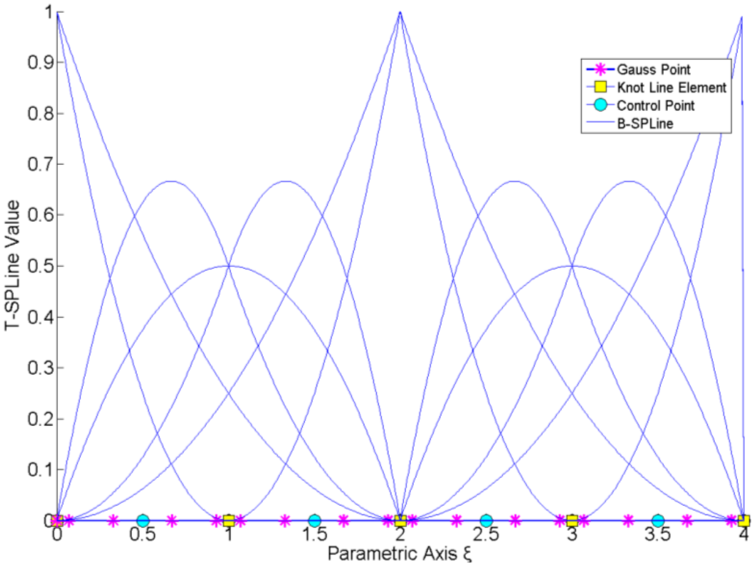


Figure 4.41. L-shaped domain. Parameter space.
Gauss points as magenta stars, knots as yellow squares and control points as cyan circles.
T-SPLine blending functions as blue lines (ξ).
(Created with GeomIso)

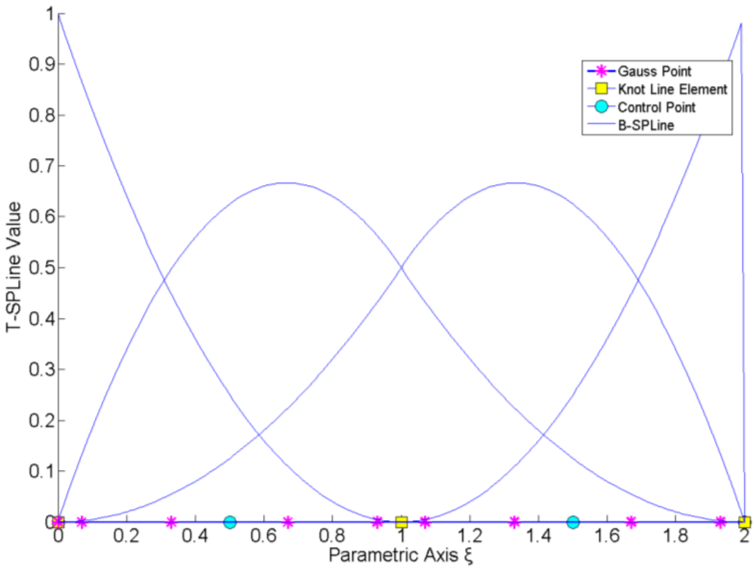
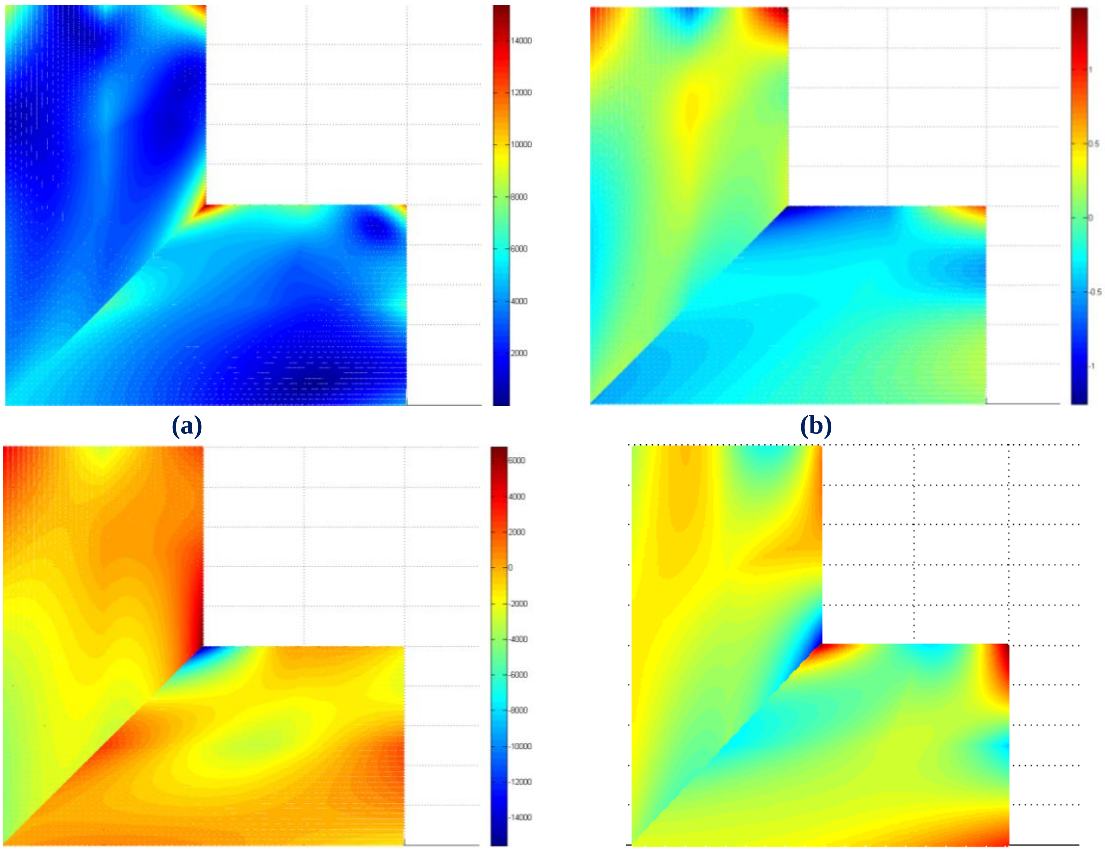


Figure 4.42. L-shaped domain. Parameter space. (Created with GeomIso)
Gauss points as magenta stars, knots as yellow squares and control points as cyan circles.
T-SPLine blending functions as blue lines (η).

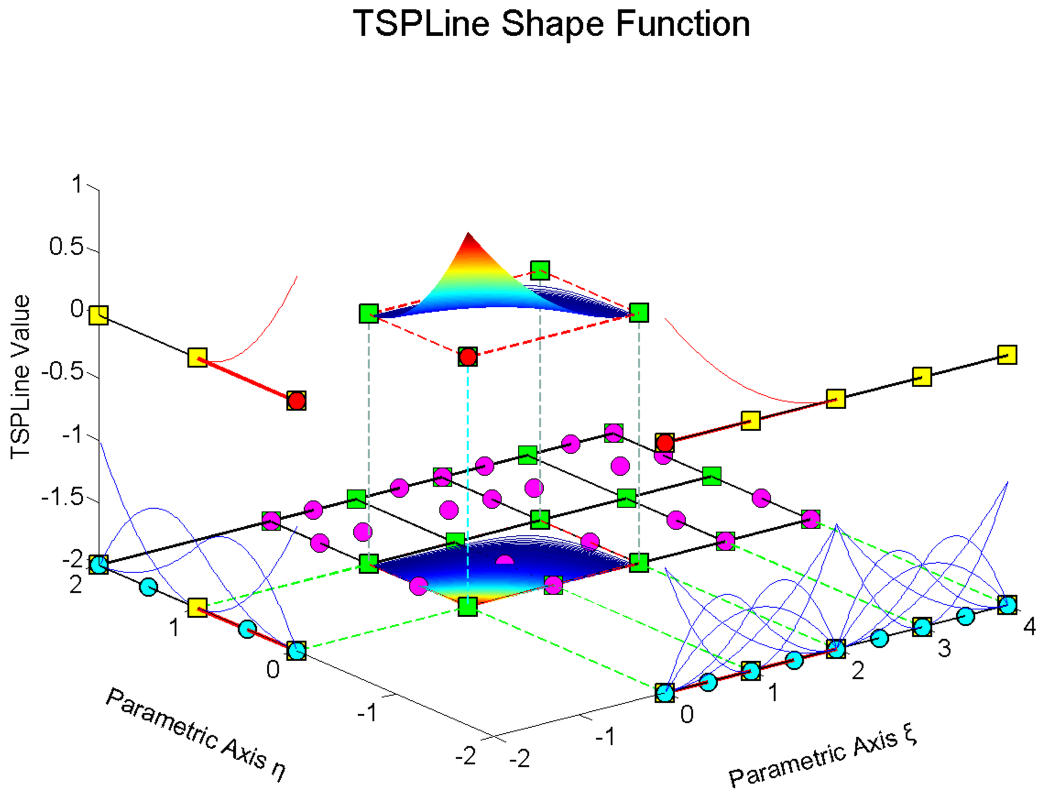
Figure 4.43 represents stress contour.



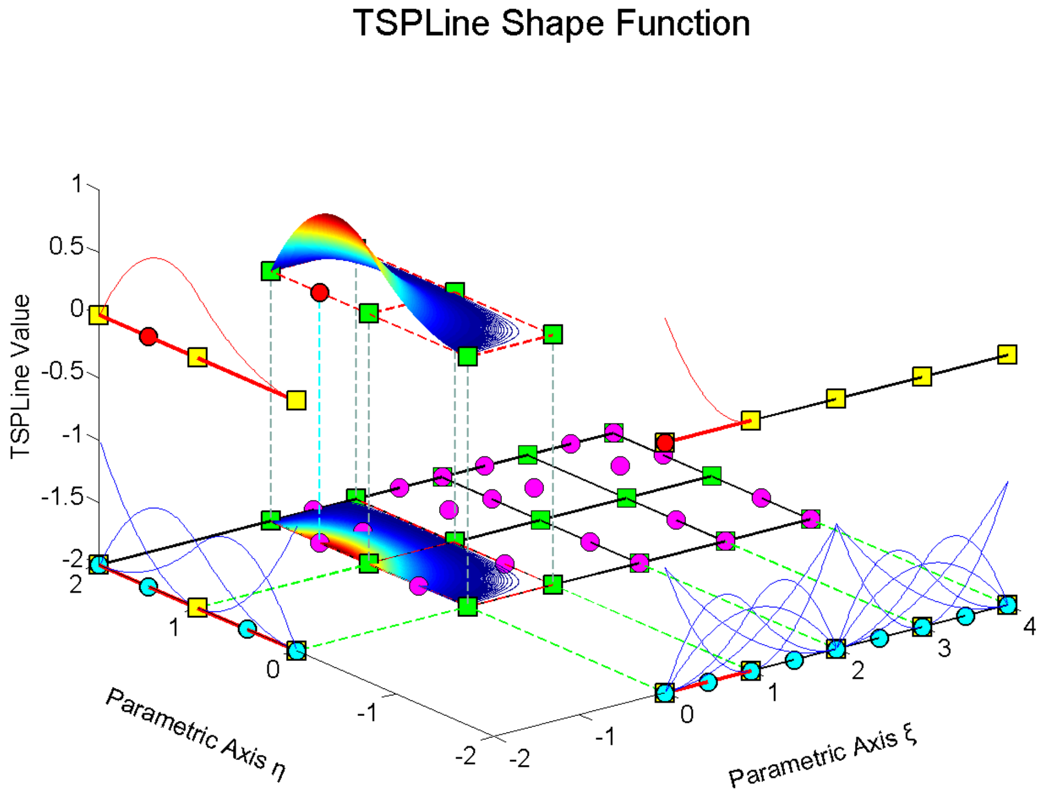
(a) **(b)**

(c) **(d)**

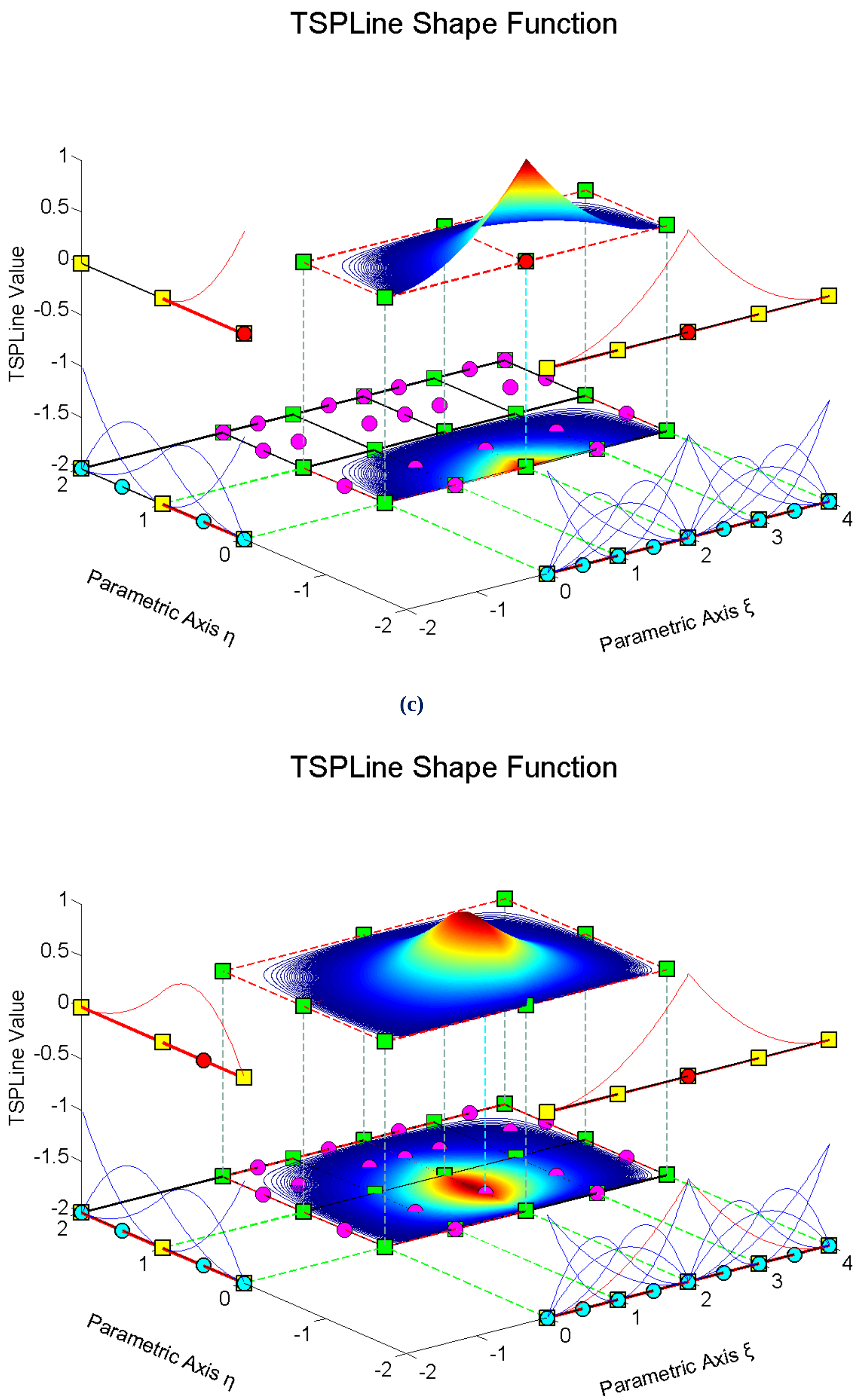
Figure 4.43. Physical space. Stress contour. (Created with GeomIso)
(a) von Mises **(b)** $\sigma_{\chi\chi}$ **(c)** σ_{yy} **(d)** τ_{xy}
Shape function with reduced continuity separates domain into two ones with continuous stress field.


TSPLine Shape Function
TSPLine Value
1
0.5
0
-0.5
-1
-1.5
-2
2
1
0
-1
-2
Parametric Axis η
-2
-1
0
1
2
3
4
Parametric Axis ξ

(a)


TSPLine Shape Function
TSPLine Value
1
0.5
0
-0.5
-1
-1.5
-2
2
1
0
-1
-2
Parametric Axis η
-2
-1
0
1
2
3
4
Parametric Axis ξ

(b)



(c)

(d)

Figure 4.44. Physical space. L-shaped domain. (Created with GeomIso)
Shape functions with reduced continuity..

5. Conclusions

5.1. Exact Geometry

Isogeometric analysis has a main advantage compared to finite element analysis, which is the 100 % utilization of the exact geometry mesh. Shape functions used for geometry design are identical to those used for mesh generation and solution approximation, thus there is no need for geometry approximation. Automatically, representation errors are eliminated. As far as T-SPLines are concerned, the full tensor product nature is no longer obligatory, fact that minimizes the required control point number for design and analysis as well. Conclusively, isogeometric analysis minimizes time needed for design and analysis, as the mesh generating step of finite elements is by passed, while, at the same time, accuracy is boosted, as no more geometrical approximation is need.

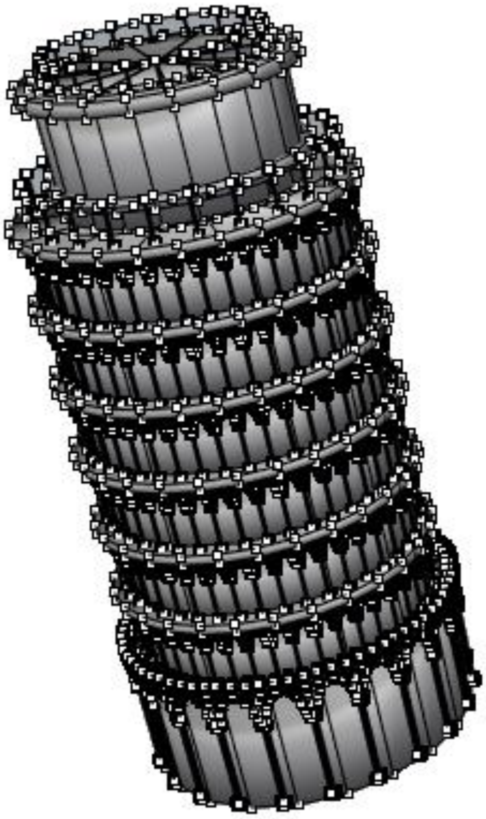
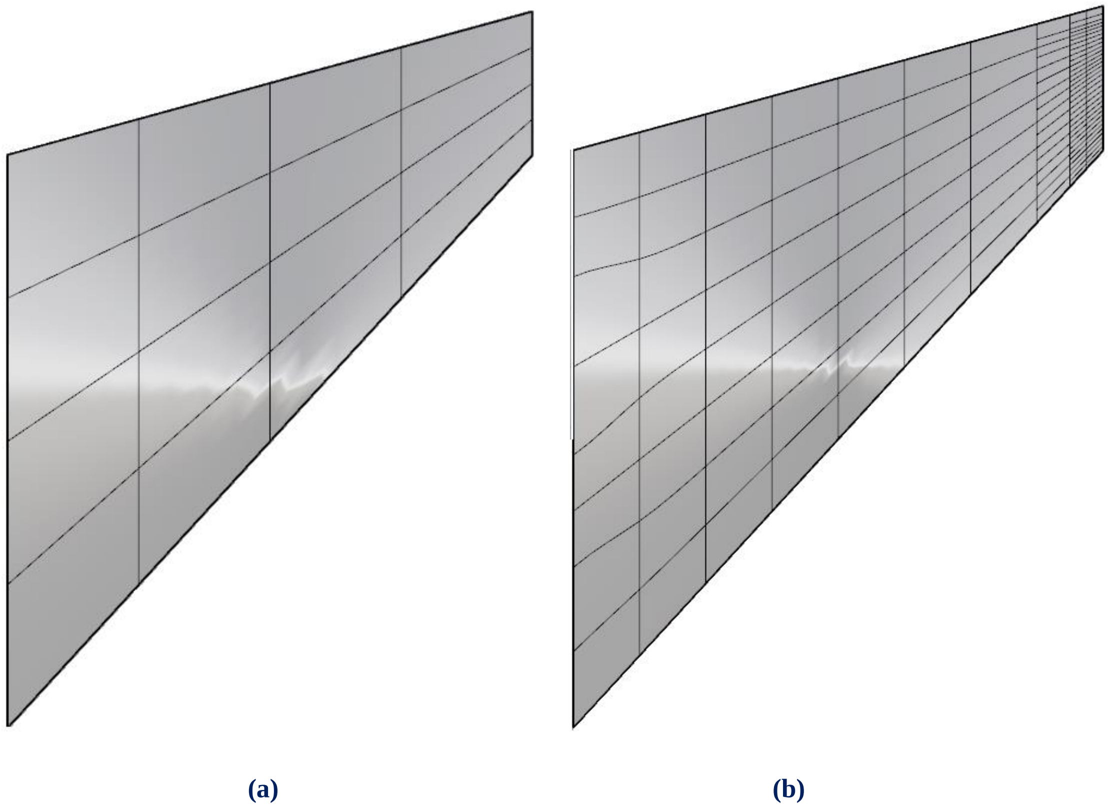


Figure 5.1.
Pisa leaning tower.
T-SPLine representation.

5.2. Local Refinement

Until now, refinement was used in finite element analysis in order to approximate better the tested geometry and then to improve the results in problematic areas, such as ones with stress concentration. On the other hand, isogeometric analysis uses the exact inserted geometry mesh in the analysis from the begging and makes FEA refinement procedure no longer needed, conserving precious CPU time. Finite method's refinement techniques demand always the initial geometry, on which a new better and finer mesh is created from scratch. This is no longer valid in IGA, as the geometry is exact and refinement schemes create finer meshes by passing the steps of FEA's re-meshing. In IGA, refinement techniques are more efficient and effective. T-SPLines are the first type of SPLines that enable local refinement (Figure 5.2). The lack of full tensor product structure allows true refinement where the structure demands it.



(a) **(b)**

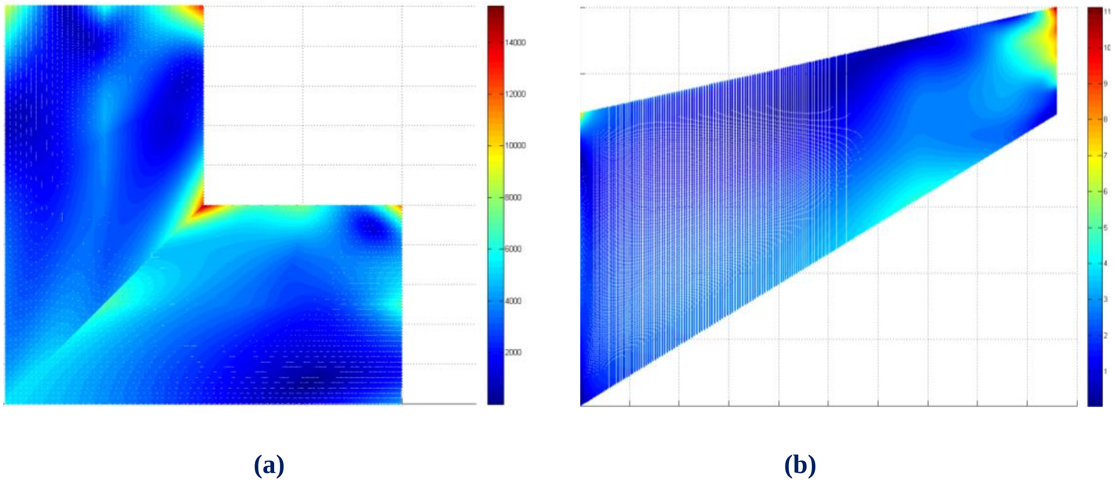
Figure 5.2.
Airplane wing.
Local refinement has been applied on the top right edge.
As a result, the mesh there is denser, without affecting the other model's area of the wing.

(a) Coarse T-SPLine mesh.

(b) Fine T-SPLine mesh,
which consists of more and smaller elements.

5.3. Element Connectivity

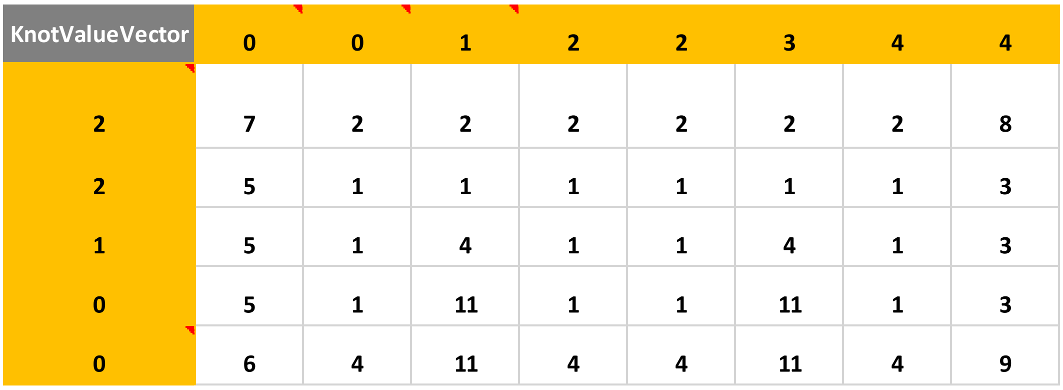
Similar to NURBS, T-SPLines have the overlapping property of their shape functions. Support can be considered constant with magnitude of p+2 spans for both axes. Since a T-Mesh has intricate interconnectivity, one span may equal to two or more knot value Spans. This way, one shape function may affect more elements. Higher order T-SPLine blending functions mean smoother derivatives, increased continuity in stress field and more accurate results.



(a)

(b)

Figure 5.3.
Stress contour σ_{xy}.
(a) L-shaped plate. C^0 continuity.
(b) Cook's cantilever. C^1 continuity.

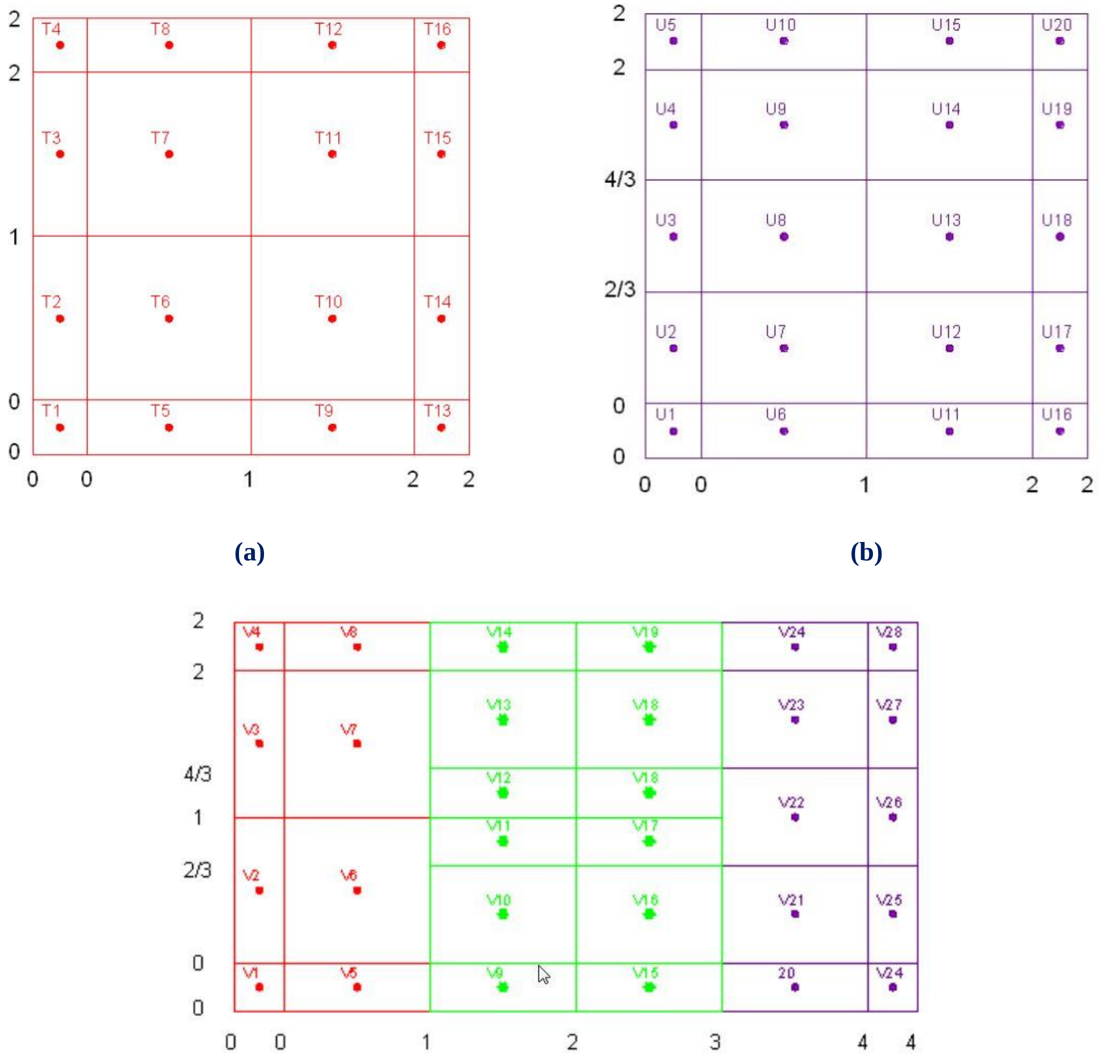


KnotValueVector	0	0	1	2	2	3	4	4
2	7	2	2	2	2	2	2	8
2	5	1	1	1	1	1	1	3
1	5	1	4	1	1	4	1	3
0	5	1	11	1	1	11	1	3
0	6	4	11	4	4	11	4	9

Figure 5.4. T-Mesh.
Junction connectivity and global knot value vectors, as given in input file.
Junction numbering is identical to the numbering, described in Chapter 2.
An equivalent NURBS mesh would contain only cross junctions,
so T-SPLine code can be used for NURBS analysis as well.

5.4. Patch Merging with T-SPLines

Patches are used in isogeometric analysis in order to simulate different materials and geometry types. This pattern inherited by finite element analysis, where each element can be considered as a single patch on its own. Patches don't allow shape functions to overlap, as they have C^{-1} continuity on their boundaries. In addition, patches are used to separate an enormous problem into more easily handled ones. The latter methodology is followed as far T-SPLines are concerned. T-SPLines are capable of representing any arbitrary topology, thus the only reason for patches division is faster analysis. Also, patch merging is a major advantage of T-SPLines against NURBS, since T-Mesh provides water tightness and smoother interconnections between patches, a problem that NURBS cannot always overcome and require much more effort and computational time.



(a)

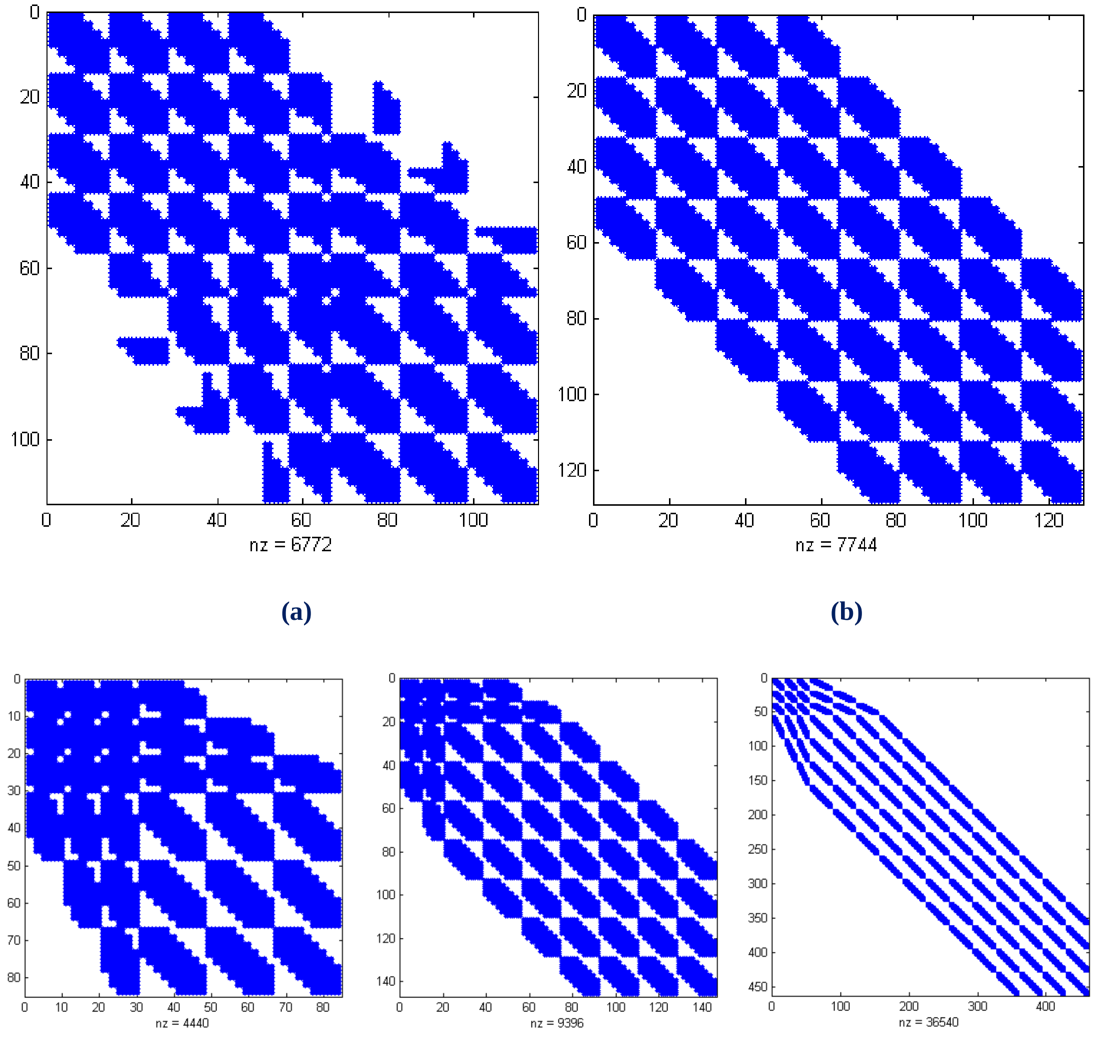
(b)

(c)

Figure 5.5.
T-SPLines enable watertight NURBS merging.

5.5. Stiffness Matrix Formulation

Stiffness matrix calculation in isogeometric analysis follows a similar fashion to finite element method. As known, in finite element analysis, elements are connected with C^0 continuity. On the other hand, IGA enables much greater continuity between elements, which leads to greater overlapping. In the case of NURBS, this overlapping is constant, while, in T-SPLines, T-junctions allow overlapping to vary. Finally, continuity of NURBS objects is constant throughout the domain, while this is not the case for T-SPLine geometries, as T-junctions enable continuity to change abruptly and give T-SPLine methodology tremendous design flexibility. As a result, stiffness matrix formulation can be a more laborious task compared to FEA.



(a)

(b)

(c)

(d)

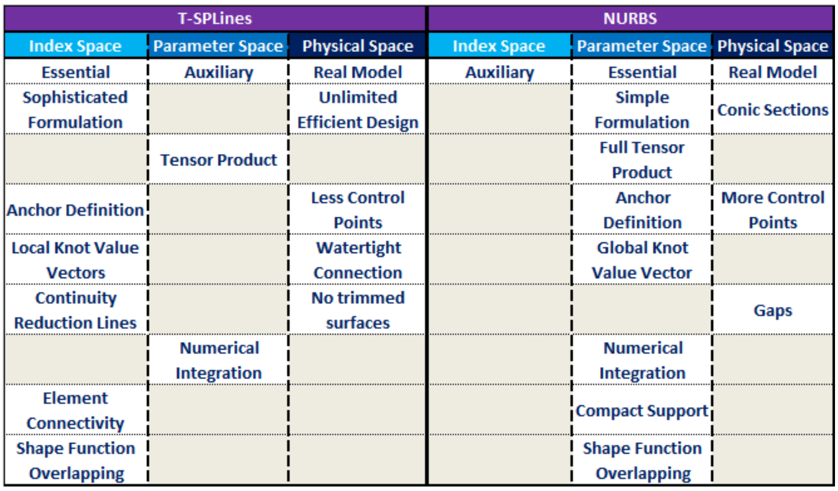
(e)

Figure 5.6.
Stiffness matrices with various interconnectivities.
Subfigures **(a)**, **(b)** show equivalent NURBS and T-SPLine stiffness matrices accordingly.
Subfigures **(c)** to **(e)** show consecutive refinement steps.

5.6. Comparison with NURBS

In general, T-SPLines, as a generalization of NURBS, exhibit more advantages compared to NURBS and overcome basic NURBS drawbacks, such as:

- patch watertightness
- local refinement
- accurate design of complex geometries. NURBS are able to design accurately only conic sections.



T-SPLines			NURBS		
Index Space	Parameter Space	Physical Space	Index Space	Parameter Space	Physical Space
Essential	Auxiliary	Real Model	Auxiliary	Essential	Real Model
Sophisticated Formulation		Unlimited Efficient Design		Simple Formulation	Conic Sections
	Tensor Product			Full Tensor Product	
Anchor Definition		Less Control Points		Anchor Definition	More Control Points
Local Knot Value Vectors		Watertight Connection		Global Knot Value Vector	
Continuity Reduction Lines		No trimmed surfaces			Gaps
	Numerical Integration			Numerical Integration	
Element Connectivity				Compact Support	
Shape Function Overlapping				Shape Function Overlapping	

Table 5.1. T-SPLines compared with NURBS.
The essential space for T-SPLines is index space, while for NURBS is parameter space.
The appropriate data of T-Mesh (junctions, anchors, local knot value vectors
and blending functions) are defined in index space.
On the other hand, NURBS' less sophisticated formulation requires only the number of control points, the BSPLine basis functions' polynomial degree and the global knot value vector (per axis).

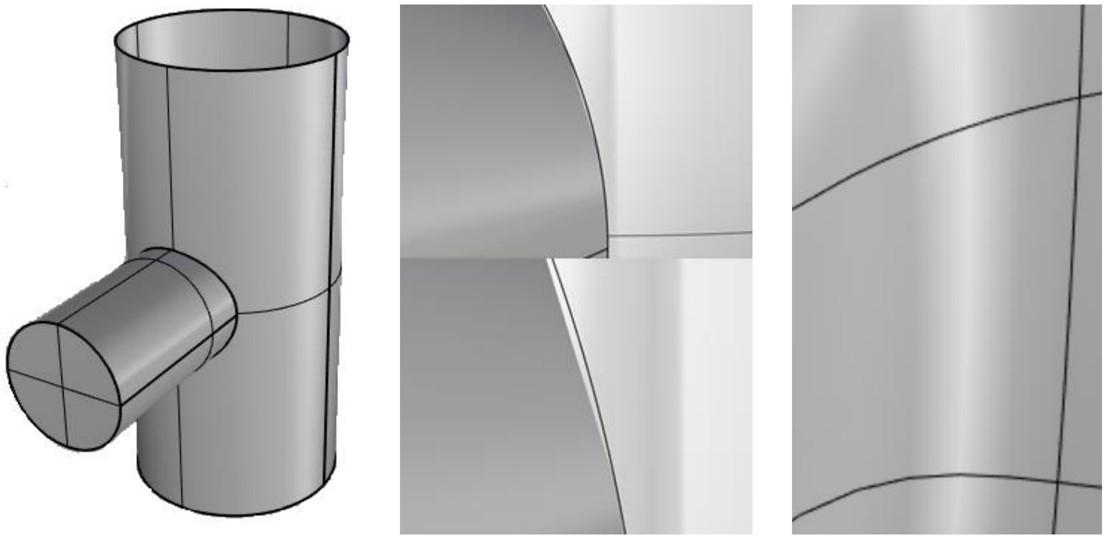


Figure 5.7. T-SPLine model.
Watertight patch interconnection.

Linear independence is a major drawback of T-SPLines, which can be easily overcome. Their intricate formulation poses programming difficulties. However, the results show that this laborious task proves itself worthwhile, as it allows water tightness, no trimmed surfaces and NURBS patch merging. At the same time, a significantly reduced required control point number is a major advantage both for geometry design and for model analysis. All the above make T-SPLines ideal for use and a candidate to conquer both computer aided design and computational analysis community in the future.

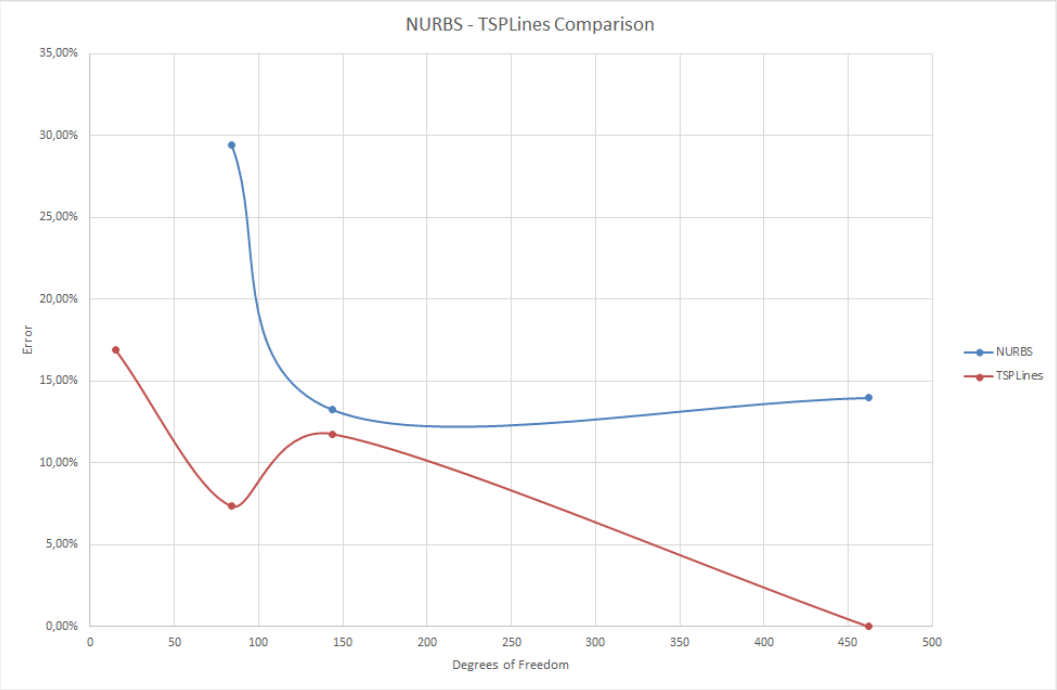


Figure 5.8. Accuracy comparison of NURBS and T-SPLines.

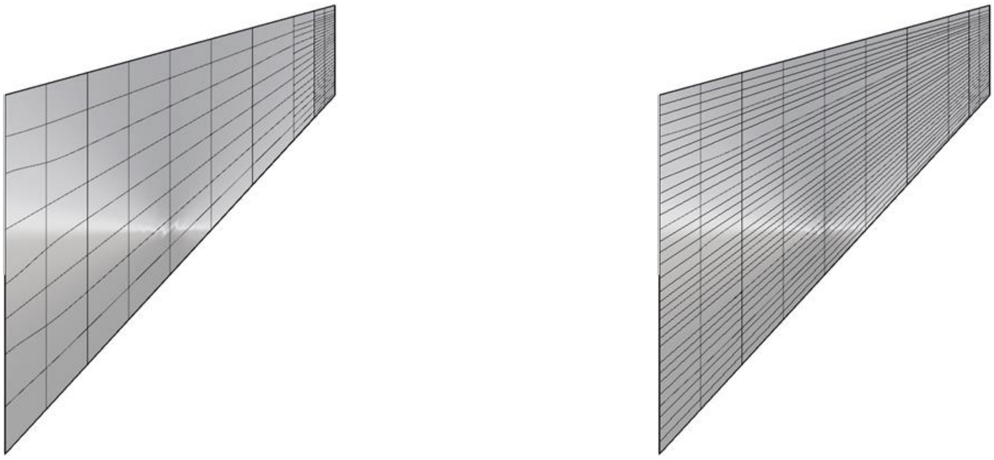
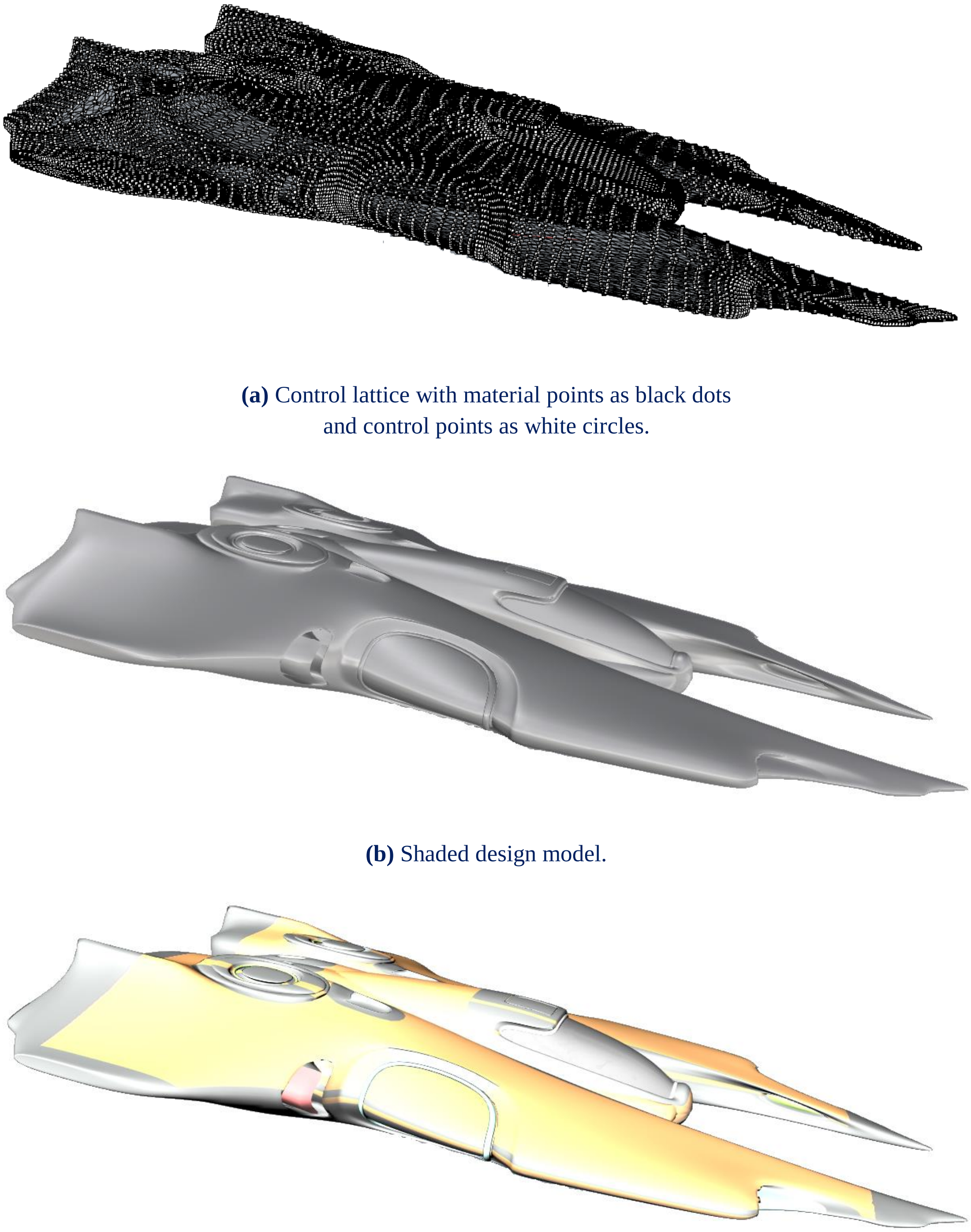


Figure 5.9.
(a) T-SPLine refinement.
(b) NURBS refinement.

6. Appendix (T-SPLine Drafts)

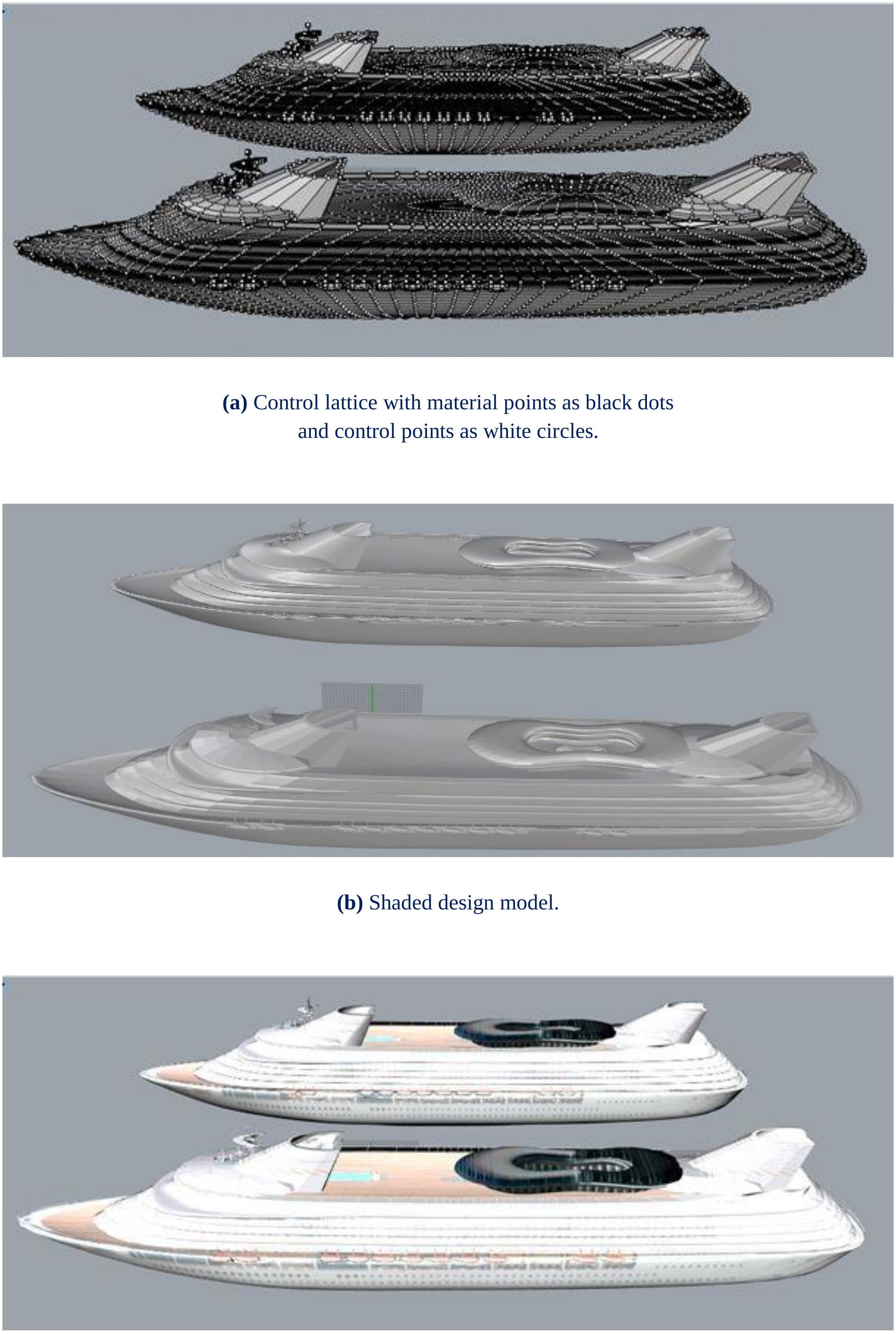


(a) Control lattice with material points as black dots and control points as white circles.

(b) Shaded design model.

(c) Rendered design model.

Figure 6.1.
Physical space.
Futuristic aircraft designed with T-SPLines.

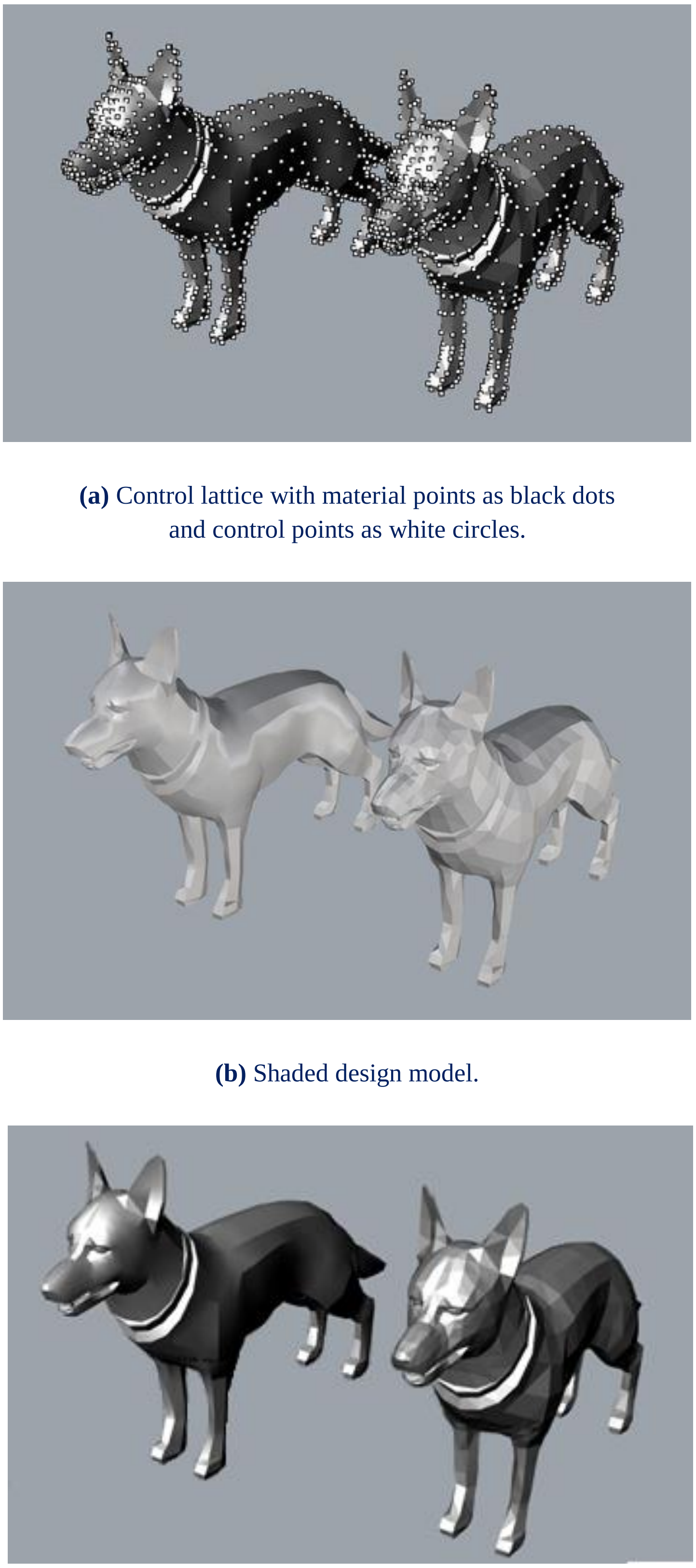


(a) Control lattice with material points as black dots
and control points as white circles.

(b) Shaded design model.

(c) Rendered design model.

Figure 6.2.
Physical space.
Cruise ship designed with T-SPLines.

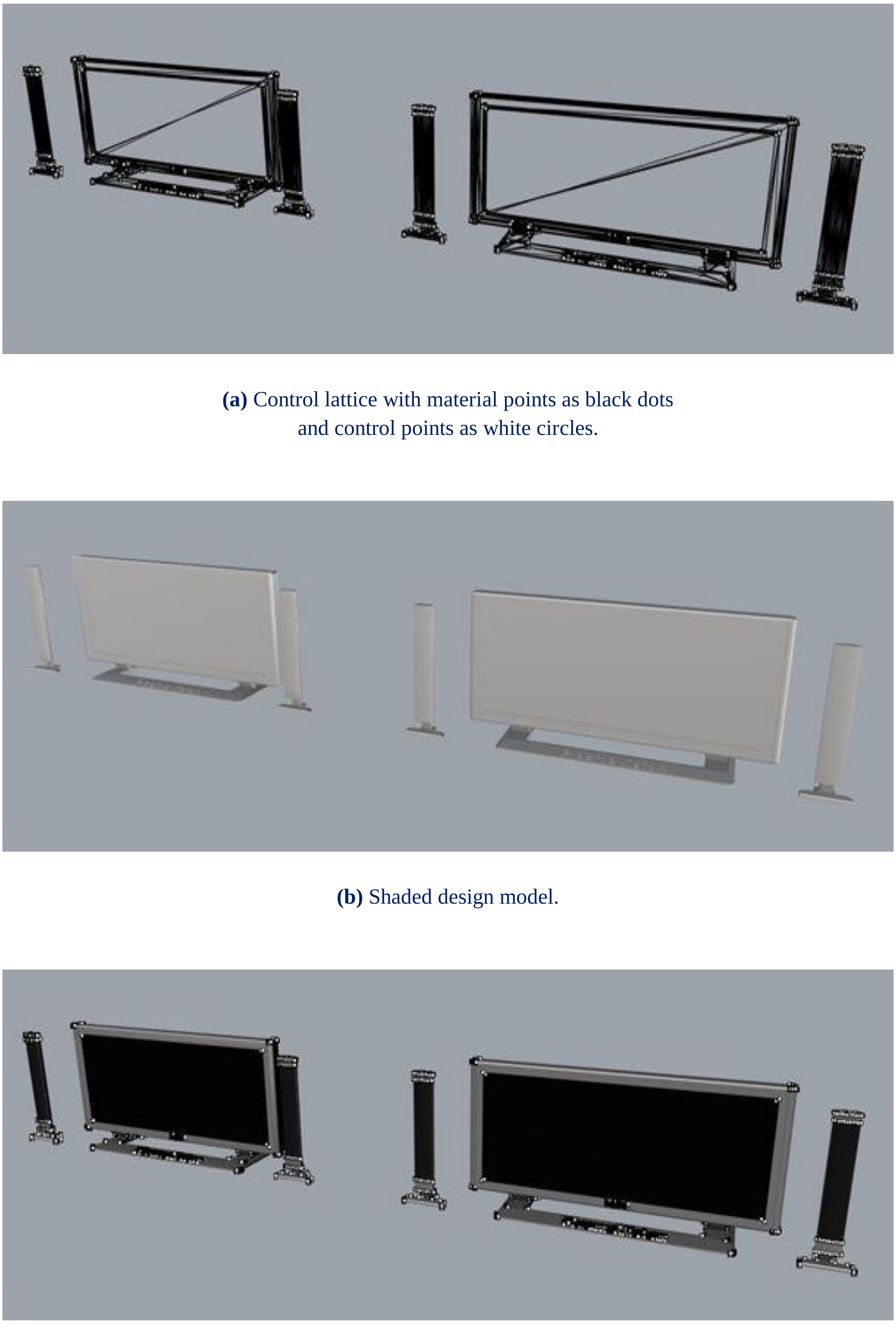


(a) Control lattice with material points as black dots and control points as white circles.

(b) Shaded design model.

(c) Rendered design model.

Figure 6.3.
Physical space.
Pets designed with T-SPLines.

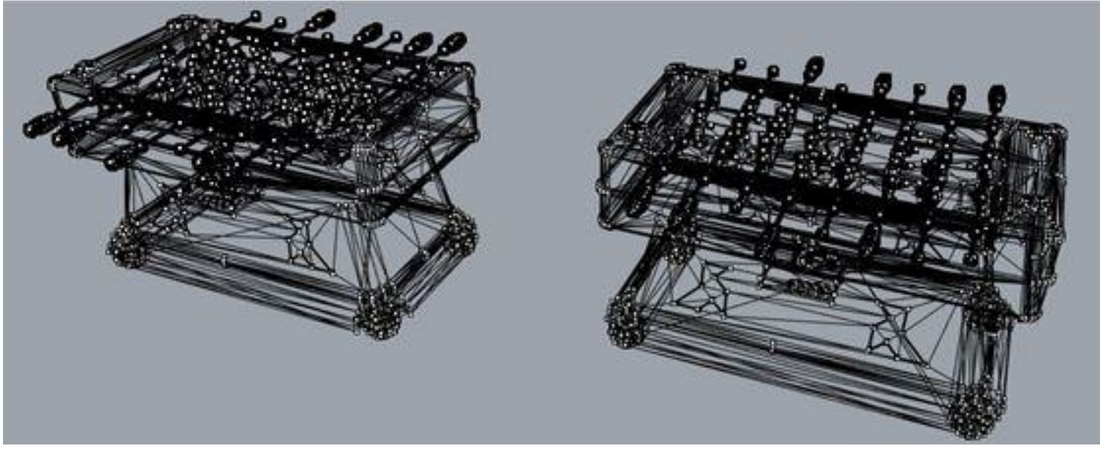


(a) Control lattice with material points as black dots
and control points as white circles.

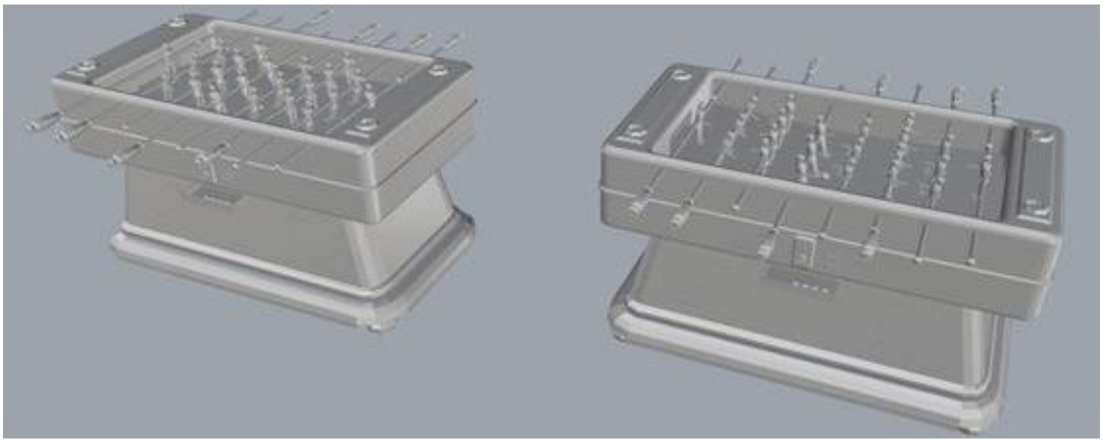
(b) Shaded design model.

(c) Rendered design model.

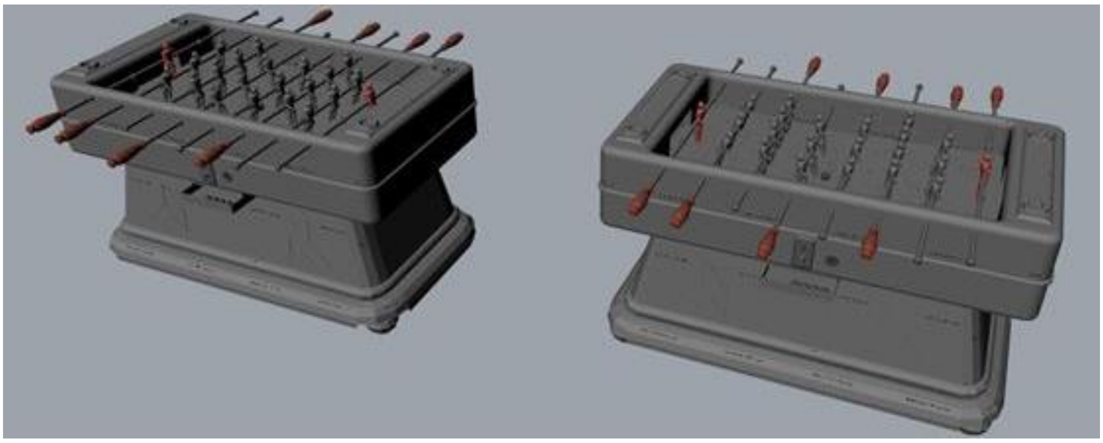
Figure 6.4.
Physical space.
Television set designed with T-SPLines.



(a) Control lattice with material points as black dots and control points as white circles.

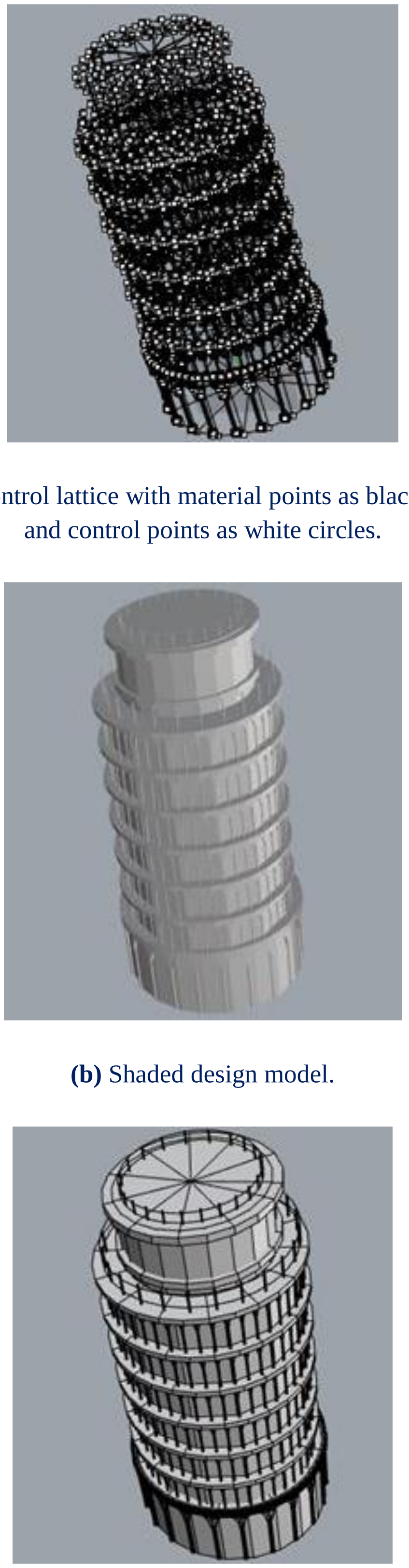


(b) Shaded design model.



(c) Rendered design model.

Figure 6.5.
Physical space.
Table soccer designed with T-SPLines.

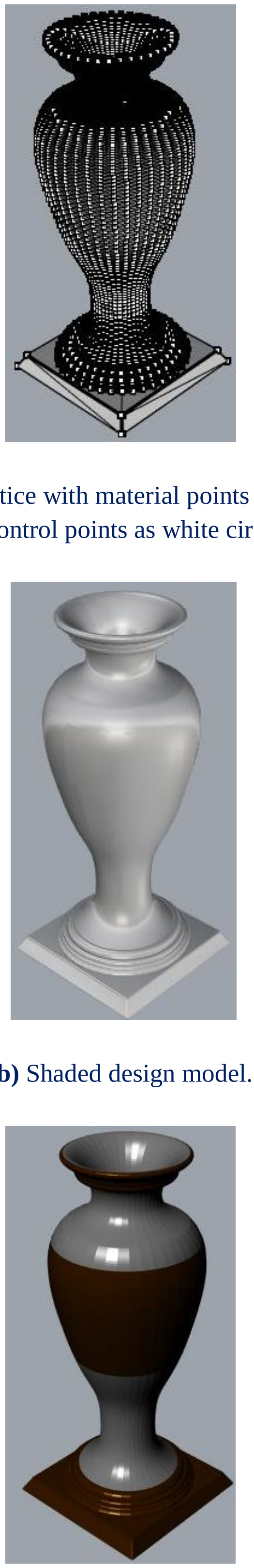


(a) Control lattice with material points as black dots and control points as white circles.

(b) Shaded design model.

(c) Rendered design model.

Figure 6.6.
Physical space.
Pisa leaning tower designed with T-SPLines.

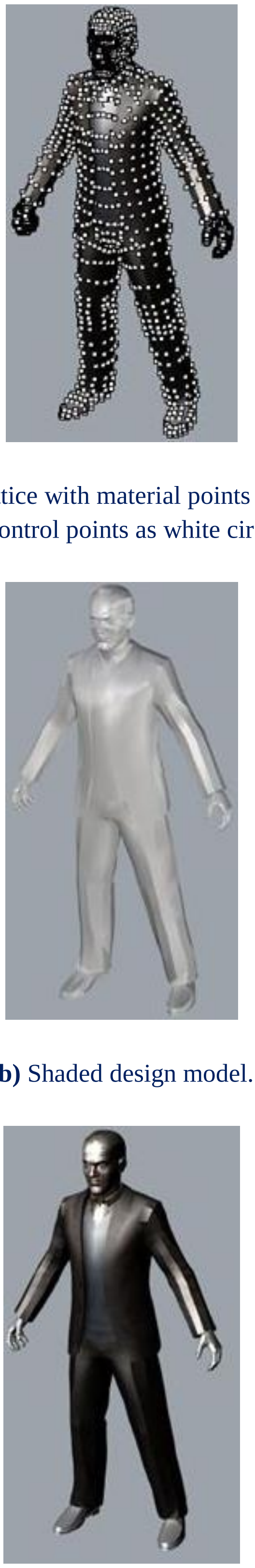


(a) Control lattice with material points as black dots and control points as white circles.

(b) Shaded design model.

(c) Rendered design model.

Figure 6.7.
Physical space.
Amphora designed with T-SPLines.



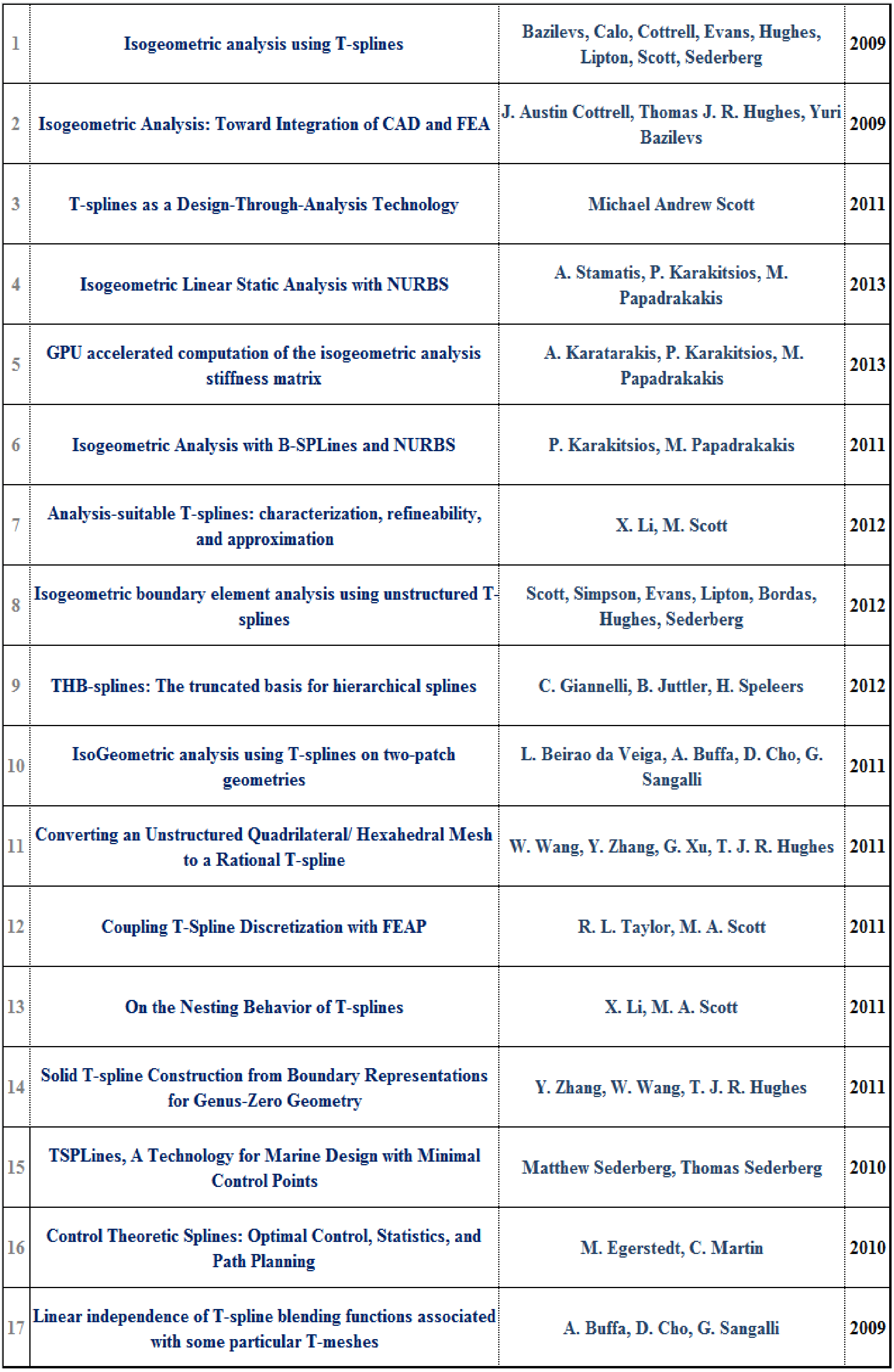
(a) Control lattice with material points as black dots and control points as white circles.

(b) Shaded design model.

(c) Rendered design model.

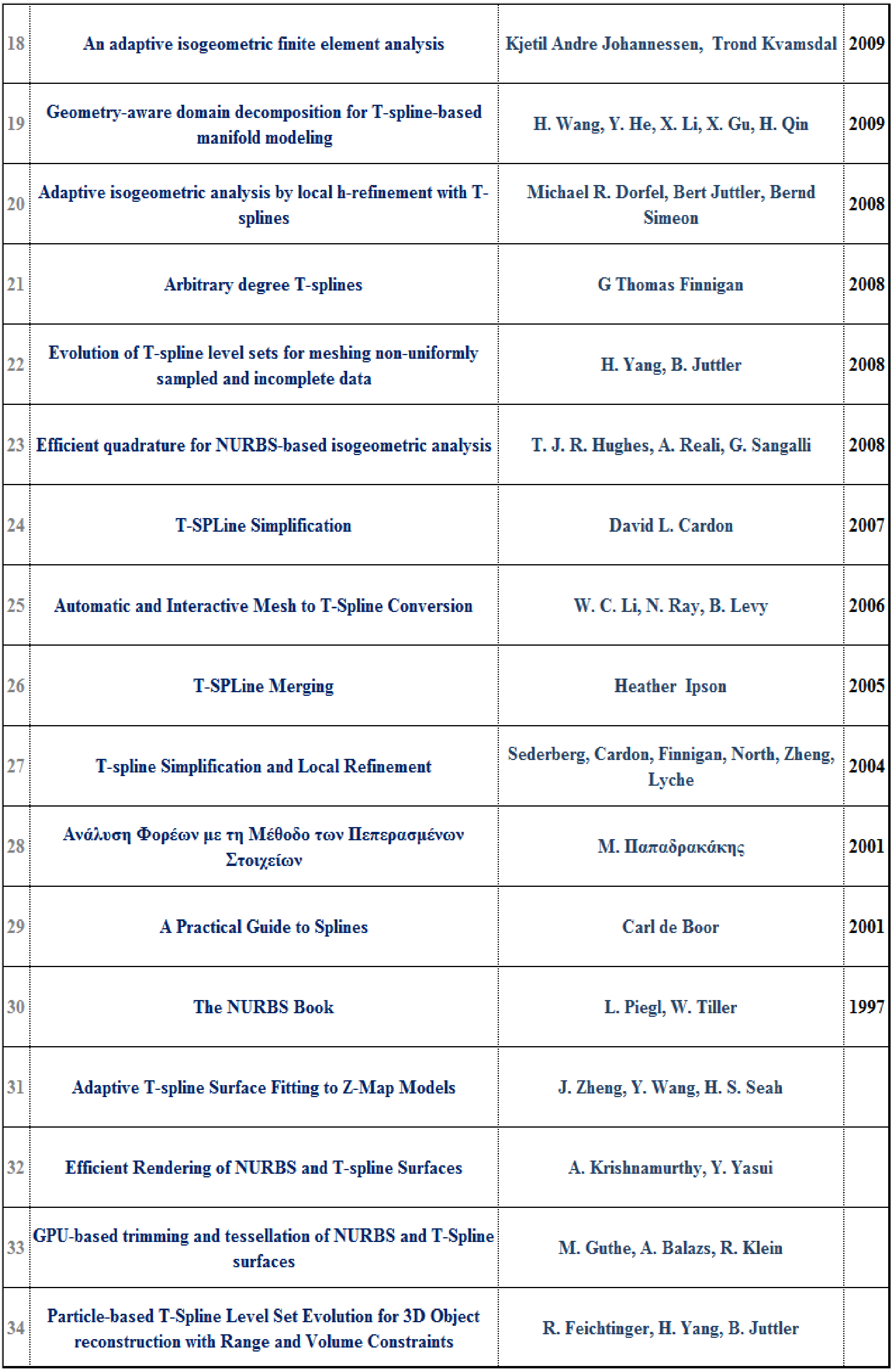
Figure 6.8.
Physical space.
Human designed with T-SPLines

References



1	Isogeometric analysis using T-splines	Bazilevs, Calo, Cottrell, Evans, Hughes, Lipton, Scott, Sederberg	2009
2	Isogeometric Analysis: Toward Integration of CAD and FEA	J. Austin Cottrell, Thomas J. R. Hughes, Yuri Bazilevs	2009
3	T-splines as a Design-Through-Analysis Technology	Michael Andrew Scott	2011
4	Isogeometric Linear Static Analysis with NURBS	A. Stamatis, P. Karakitsios, M. Papadrakakis	2013
5	GPU accelerated computation of the isogeometric analysis stiffness matrix	A. Karatarakis, P. Karakitsios, M. Papadrakakis	2013
6	Isogeometric Analysis with B-SPLines and NURBS	P. Karakitsios, M. Papadrakakis	2011
7	Analysis-suitable T-splines: characterization, refineability, and approximation	X. Li, M. Scott	2012
8	Isogeometric boundary element analysis using unstructured T-splines	Scott, Simpson, Evans, Lipton, Bordas, Hughes, Sederberg	2012
9	THB-splines: The truncated basis for hierarchical splines	C. Giannelli, B. Juttler, H. Speleers	2012
10	IsoGeometric analysis using T-splines on two-patch geometries	L. Beirao da Veiga, A. Buffa, D. Cho, G. Sangalli	2011
11	Converting an Unstructured Quadrilateral/ Hexahedral Mesh to a Rational T-spline	W. Wang, Y. Zhang, G. Xu, T. J. R. Hughes	2011
12	Coupling T-Spline Discretization with FEAP	R. L. Taylor, M. A. Scott	2011
13	On the Nesting Behavior of T-splines	X. Li, M. A. Scott	2011
14	Solid T-spline Construction from Boundary Representations for Genus-Zero Geometry	Y. Zhang, W. Wang, T. J. R. Hughes	2011
15	TSPLines, A Technology for Marine Design with Minimal Control Points	Matthew Sederberg, Thomas Sederberg	2010
16	Control Theoretic Splines: Optimal Control, Statistics, and Path Planning	M. Egerstedt, C. Martin	2010
17	Linear independence of T-spline blending functions associated with some particular T-meshes	A. Buffa, D. Cho, G. Sangalli	2009

References



18	An adaptive isogeometric finite element analysis	Kjetil Andre Johannessen, Trond Kvamsdal	2009
19	Geometry-aware domain decomposition for T-spline-based manifold modeling	H. Wang, Y. He, X. Li, X. Gu, H. Qin	2009
20	Adaptive isogeometric analysis by local h-refinement with T-splines	Michael R. Dorfel, Bert Juttler, Bernd Simeon	2008
21	Arbitrary degree T-splines	G Thomas Finnigan	2008
22	Evolution of T-spline level sets for meshing non-uniformly sampled and incomplete data	H. Yang, B. Juttler	2008
23	Efficient quadrature for NURBS-based isogeometric analysis	T. J. R. Hughes, A. Reali, G. Sangalli	2008
24	T-SPLine Simplification	David L. Cardon	2007
25	Automatic and Interactive Mesh to T-Spline Conversion	W. C. Li, N. Ray, B. Levy	2006
26	T-SPLine Merging	Heather Ipson	2005
27	T-spline Simplification and Local Refinement	Sederberg, Cardon, Finnigan, North, Zheng, Lyche	2004
28	Ανάλυση Φορέων με τη Μέθοδο των Πεπερασμένων Στοιχείων	Μ. Παπαδρακάκης	2001
29	A Practical Guide to Splines	Carl de Boor	2001
30	The NURBS Book	L. Piegl, W. Tiller	1997
31	Adaptive T-spline Surface Fitting to Z-Map Models	J. Zheng, Y. Wang, H. S. Seah	
32	Efficient Rendering of NURBS and T-spline Surfaces	A. Krishnamurthy, Y. Yasui	
33	GPU-based trimming and tessellation of NURBS and T-Spline surfaces	M. Guthe, A. Balazs, R. Klein	
34	Particle-based T-Spline Level Set Evolution for 3D Object reconstruction with Range and Volume Constraints	R. Feichtinger, H. Yang, B. Juttler	