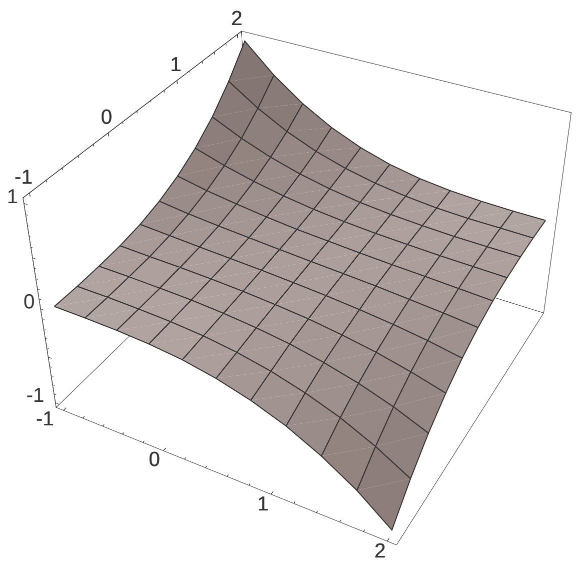
Geometric Modelling



Lecture in Winter Term
by G. Greiner

Transscript by J. Kaminski
Revision by S. Bergler

Contents

1 General Remarks on Curves and Surfaces 1

2 Polynomial Curves 3
2.1 Basics . . . 3
2.2 Different Basis of $\mathbb{P}_n$ . . . 4
2.2.1 Lagrange Polynomials . . . 4
2.2.2 Bernstein Polynomials . . . 5
2.2.3 Comparison of Lagrange and Bernstein Representation . . . 7
2.2.4 Change of Basis . . . 7
2.3 Polar Forms . . . 8

3 Bézier Curves 11
3.1 Basics . . . 11
3.1.1 Shape Properties for Bézier Curves . . . 13
3.2 Algorithms for Bézier Curves . . . 15
3.2.1 The de Casteljau Algorithm . . . 15
3.2.2 The de Casteljau Subdivision Algorithm . . . 17
3.2.3 How to draw a Bézier Curve . . . 18
3.2.4 Degree Elevation . . . 19
3.2.5 Degree Reduction . . . 21
3.3 Bézier Splines . . . 23
3.3.1 Construction of “complex” Curves . . . 23
3.3.2 C^k-Continuity of two Bézier Curves . . . 24
3.3.3 Geometrical Construction of continuous Joins . . . 26
3.3.4 Interpolation with Bézier Splines . . . 28

4 B-splines 31
4.1 Basics . . . 31
4.2 B-spline Curves . . . 34
4.2.1 Shape Properties of B-spline Curves . . . 35
4.2.2 Comparison of B-Spline and Bézier Curves . . . 36
4.3 Control Points and Polar Forms . . . 37
4.4 Multiple Knots . . . 38
4.5 Algorithms for B-spline Curves . . . 41
4.5.1 The de Boor Algorithm . . . 41
4.5.2 Conversion of B-spline Curves into Bézier Splines . . . 45
4.5.3 The Boehm Algorithm . . . 45
4.5.4 Knot Deletion . . . 47
4.5.5 Subdivision for uniform B-spline curves . . . 48

4.6 Interpolation and Approximation . . . 51
4.6.1 B-spline Interpolation . . . 51
4.6.2 Special case: Cubic spline interpolation at the knots . . . 52
4.6.3 B-spline approximation . . . 54
4.6.4 Parametrization . . . 55

5 Rational Curves 57
5.1 Rational Bézier Curves . . . 57
5.1.1 Properties of rational Bézier curves . . . 58
5.1.2 Evaluation of rational Béziercurves . . . 59
5.1.3 Influence and properties of the weights . . . 59
5.1.4 Quadratic rational Bézier Curves as Conic Sections . . . 61
5.1.5 Segments of Conic Sections as quadratic rational Bézier Curves . . . 62
5.1.6 Rational Bézier-Spline Curves . . . 64
5.2 Rational B-Spline Curves (NURBS) . . . 65

6 Elementary Differential Geometry for Curves 67
6.1 Reparametrization . . . 67
6.2 Planar Curves . . . 68
6.3 Space Curves . . . 70
6.4 Geometric and Parametric Continuity . . . 72
6.4.1 Geometric Continuity of Bézier Curves . . . 74

7 Tensor Product Surfaces 77
7.1 General Approach . . . 77
7.2 Tensor Product Bézier Surfaces . . . 78
7.3 Algorithms for TP Bézier Surfaces . . . 79
7.3.1 Evaluation with the TP Approach . . . 79
7.3.2 The “direct” de Casteljau Algorithm . . . 80
7.4 Tensor Product B-Spline Surfaces . . . 82
7.5 Algorithms for TP B-Spline Surfaces . . . 83
7.5.1 Evaluation of TP B-Spline Surfaces . . . 83
7.5.2 Knot insertion . . . 83
7.6 Interpolation and Approximation . . . 84
7.6.1 General interpolation . . . 84
7.6.2 Special case: Interpolation at the knots . . . 84
7.7 Trimming of parametric Patches . . . 85

8 Ray Tracing of TP-Bézier Surfaces (Bézier Clipping) 87

9 Triangular Bézier Patches 91
9.1 Barycentric Coordinates in $\mathbb{R}^2$ . . . 91
9.2 Bernstein Polynomials over Triangles . . . 92
9.3 Triangular Bézier Patches . . . 95
9.4 Algorithms for Triangular Bézier Patches . . . 96
9.4.1 The de Casteljau Algorithm . . . 96

9.4.2 Degree Elevation 96
9.4.3 Subdivision 97
9.5 C^k-Continuity of Triangular Bézier Patches 100
9.6 Interpolation with piecewise Bézier Triangles (Split Surfaces) 102
9.6.1 Clough-Tocher 102
9.6.2 Powell-Sabin 103

10 Triangle meshes **105**
10.1 Basics 105
10.2 Computer representation of triangle meshes 109
10.3 Parametrization of triangle meshes 111
10.3.1 The spring modell 111
10.3.2 Parametrization of the border vertices 112
10.3.3 Parametrization of the inner vertices 112

11 Subdivision Surfaces **115**
11.1 Approximative schemes: Loop-Subdivision 117
11.2 Interpolatory schemes: Modified Butterfly-Subdivision 120
11.3 Limit points for approximative subdivision 121
11.3.1 Limit points for the subdivision of cubic splines 121
11.3.2 Limit points for the subdivision of regular triangle meshes 124

List of Figures

1.1 Curves in the plane and in space 1
1.2 A torus and a free form surface 1

2.1 Quadratic Lagrange polynomials with respect to paramter values 0, $\frac{1}{2}$ and 1 . . 5
2.2 Bernstein polynomials for $n = 1$, $n = 2$ and $n = 3$ 6
2.3 Polar Form of a Curve of degree $n = 2$ 10

3.1 A Bézier curve 12
3.2 Convex hull 13
3.3 Variation diminishing property 14
3.4 de Casteljau schemas for $n = 2$ and $n = 3$ 15
3.5 Graphical representation of the de Casteljau algorithm 15
3.6 Evaluation of $F(0)$ with de Casteljau 16
3.7 Left partial curve after de Casteljau subdivision 18
3.8 Multiple subdivision 19
3.9 Recursive subdivision 19
3.10 Degree elevation $n = 2 \to n = 3$ 19
3.11 Degree elevation $n = 3 \to n = 4$ 20
3.12 Degree reduction $n = 3 \to n = 2$ 22
3.13 Joins of Bézier curves 24
3.14 Join conditions in the de Casteljau algorithm 25
3.15 de Casteljau conditions for C^1- and C^2-continuity 26
3.16 C^2-continuation of the standard parabola 27
3.17 C^2-join of several cubic Bézier segments 27
3.18 Interpolation with Catmull-Rom Splines 29

4.1 A knot vector 31
4.2 B-splines for $k = 0$ 32
4.3 B-splines for $k = 1$ 32
4.4 The B-spline $N_3^2(u)$ 33
4.5 Local convex hull 35
4.6 $n = 3$ control points coincide 35
4.7 $n = 3$ control points on a line 35
4.8 $n + 1 = 4$ control points on a line 36
4.9 B-splines over a knot vector with multiplicities (emphasized: N_1^1) 39
4.10 Bézier representation of a B-spline curve with endpoint interpolation 42
4.11 Evaluation schema of the de Boor algorithm for $n = 3$ 42
4.12 Visualization of the coefficients for the de Boor algorithm ($n = 3$) 43
4.13 Example for the de Boor algorithm (1): coefficients 43

4.14 Example for the de Boor algorithm (2): evaluation 44
4.15 Example for the de Boor algorithm (3): graphical construction 44
4.16 Schema for the Boehm algorithm 46
4.17 Example for the Boehm algorithm 47
4.18 Knot insertion at $u = \frac{5}{2}, \frac{7}{2}, \frac{9}{2}$ 48
4.19 Geometrical construction: cutting the corners 49
4.20 Subdivision with open knot vector and $n = 2$ 50
4.21 Illustration of parameter values 51
4.22 Chordal parametrisation 56

5.1 Homogeneous coordinates 57
5.2 Interpretation of rat. curves with homogeneous coordinates (proj. $\mathbb{R}^3 \to \mathbb{R}^2$) . . 58
5.3 de Casteljau scheme for rational Bézier curves 59
5.4 Effect of the weights ($n = 2$) 60
5.5 Conic sections: ellipsis, parabola und hyperbola 61
5.6 Intersection of a cone with a pyramid and projection of the intersection line . . 64

6.1 Arc length 68
6.2 The 2D Frenet frame 69
6.3 The 3D Frenet frame 71
6.4 Spiral 72
6.5 Continuity of Bézier splines 73
6.6 Generalized A-frame property 75

7.1 A tensor product Bézier surface 77
7.2 Parameter grid, control net and resulting surface 78
7.3 de Casteljau scheme for tensor product Bézier surfaces 80
7.4 Bilinear interpolation 80
7.5 Direct de Casteljau scheme 81
7.6 Knot insertion with the Boehm algorithm 83
7.7 Building a hierarchical knot grid for B-spline patches 83
7.8 Trimming a patch 85
7.9 Specifying inner and outer contours in the parameter domain 85
7.10 Determining inside and outside of a trimmed patch 86

8.1 Bézier clipping, step 1 87
8.2 Bézier clipping, step 2 88
8.3 Bézier clipping, step 3 89
8.4 Bézier clipping, steps 5 and 6 90
8.5 Bézier clipping, step 7 90

9.1 Barycentric coordinates 92
9.2 Divide ratios and barycentric coordinates 93
9.3 Arrangement of indices ijk for $n = 3$ Bernstein polynomials 93
9.4 Triangular control nets for $n = 2$ and $n = 3$ 95
9.5 de Casteljau scheme for triangular Bézier patches 96
9.6 Degree elevation $n = 2 \to n = 3$ 97

9.7 Splitting triangles: 1-to-2 and 1-to-4 split 98
9.8 A control net and the first 1-to-4 subdivision step 98
9.9 Farin-Prautzsch 1-to-4 subdivision scheme 99
9.10 Attaching two Bézier triangles continuously 100
9.11 “Diamond condition” for $n = 3$ 101
9.12 Clough-Tocher split surface scheme 102
9.13 Powell-Sabin split surface scheme 103

10.1 1-neighbourhood and 2-neighbourhood of a vertice v 105
10.2 not possible in manifold triangulations 106
10.3 open and closed fan 107
10.4 Moebius Strip and Klein Bottle 107
10.5 Tetrahedron 108
10.6 Two-torus: surface of genus 2 108
10.7 A local part of a triangular mesh (a) is mapped into the plane by expontial mapping (b) and the local weights are determined (c). 114

11.1 1-to-4 split 115
11.2 mask for the calculation of odd and even vertices 117
11.3 masks for the calculation of irregular even vertices 117
11.4 Loop-Subdivision: top: coarse triangle mesh, bottom: after 1 subdivision step . 118
11.5 Loop-Subdivision: top: after 2 steps, bottom: after 3 steps 119
11.6 Butterfly: Mask for the calculation of the odd vertices 120
11.7 one irregular neighbour vertice with valence 3, 4 and 5 120
11.8 4-point-scheme for boarder vertices 121
11.9 Butterfly-Subdivision: top: coarse triangle mesh, bottom: after 1 subdivision step 122
11.10 Butterfly-Subdivision: top: after 2 steps, bottom: after 3 steps 123

Bibliography

[1] C. de Boor. A Practical Guide to Splines (Applied Mathematical Sciences Vol. 27). *Springer*, Heidelberg 1987

[2] M. do Carmo. Differential Geometry of Curves and Surfaces. *Prentice Hall*, Englewood Cliffs, 1976

[3] J. Hoschek, D. Lasser. Grundlagen der geometrischen Datenverarbeitung. *Teubner*, Stuttgart, 1992

[4] G. Farin. Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide. *Academic Press*, San Diego, 4^{th} edition, 1997

[5] L. Piegel. The NURBS Book. *Springer*, 1995

1 General Remarks on Curves and Surfaces

- A *curve* is a one-dimensional connected point set in a two-dimensional plane or in three-dimensional space. It is at least piecewise smooth.
- A *surface* is a two-dimensional point set in three-dimensional space with the same properties.

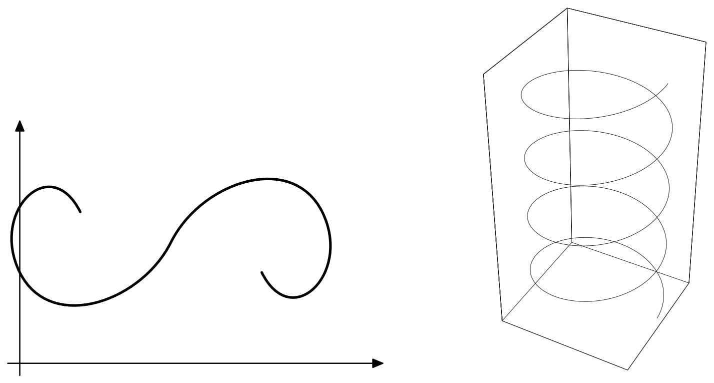


Figure 1.1: Curves in the plane and in space

- Distinguish between *regular* and *free form* (= non-regular) curves and surfaces. Regular curves are e. g. lines and line segments, circles and arcs; regular surfaces are e. g. triangles, rectangles, polygons in the plane and spheres, cylinders, cones and parts of them in 3D space.
  Non-regular curves and surfaces is everything else (e. g. fender or hood of a car).

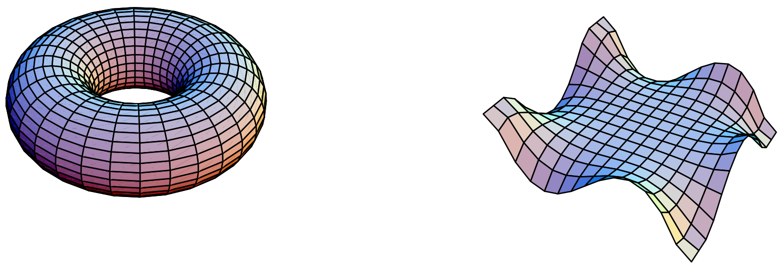


Figure 1.2: A torus and a free form surface

There are three ways to describe curves and surfaces mathematically.

example circle:

implicit	explicit	parameterized
$h(x,y) = x^2 + y^2 - r^2 = 0$	Graph of $f(x) = \pm\sqrt{r^2 - x^2}$	$F(t) = \begin{pmatrix} r \cdot \cos t \\ r \cdot \sin t \end{pmatrix}$
$h : \mathbb{R}^2 \to \mathbb{R}$	$f : \mathbb{R} \to \mathbb{R}$	$F : \mathbb{R} \to \mathbb{R}^2$

example sphere:

implicit	explicit	parameterized
$h(x,y,z) =$ $= x^2 + y^2 + z^2 - r^2 = 0$	Graph von $f(x,y) = \pm\sqrt{r^2 - x^2 - y^2}$	$F(t,s) = \begin{pmatrix} r \cdot \cos s \cdot \cos t \\ r \cdot \cos s \cdot \sin t \\ r \cdot \sin s \end{pmatrix}$
$h : \mathbb{R}^3 \to \mathbb{R}$	$f : \mathbb{R}^2 \to \mathbb{R}$	$F : \mathbb{R}^2 \to \mathbb{R}^3$

Proposition: "Reasonable" curves and surfaces can be described (at least piecewise) in each of the tree ways. Note that h and F are not unique!
Change of representation:

- $f \to F$: $F(t) = \begin{pmatrix} t \\ f(t) \end{pmatrix}$
- $F \to f$: $F(t) = \begin{pmatrix} F_1(t) \\ F_2(t) \end{pmatrix}$

  Invert $t \mapsto F_1(t)$ and define $f(x) := F_2(F_1^{-1}(x))$.

  Remark: "reasonable" means here: differentiable and $F'(t) \neq 0$, e. g. $F_1'(t) \neq 0$.
- $f \to h$: $h(x,y) = y - f(x)$
- $h \to f$: implicit function theorem. If the following statements are true, $h(x,y) = 0$ can be solved to $y = f(x)$:
  - $h(x,y)$ continuous
  - $h(x,y)$ has continous partial derivations $h_x(x,y)$ and $h_y(x,y)$
  - $h(x_0,y_0) = 0$ and $h_y(x_0,y_0) \neq 0$

  and it holds

$$f'(x) = -\frac{\frac{\partial}{\partial x} h(x,y)}{\frac{\partial}{\partial y} h(x,y)}, \quad y := f(x)$$

All statements are also true for surfaces.
Remark:

- regular forms (circles, straight lines, spheres, cones usw.) are mostly described explicitly
- free form geometries mostly parametric

2 Polynomial Curves

2.1 Basics

- Polynomial of nth degree: $p(u) = a_n\, u^n + a_{n-1}\, u^{n-1} + \cdots + a_1\, u + a_0$
- $p \in \mathbb{P} =$ set of all polynomials
- $a_n, \ldots, a_0 \in \mathbb{R}$ are coefficients of the polynomial
- If $a_n \neq 0$, then n is the degree of the polynomial
- Set of all polynomials of degree $\leq n$: $\mathbb{P}_n := \{p \in \mathbb{P} : \text{degree}(p) \leq n\}$

Definition 2.1 (Polynomial Curve)
A two-dimensional (three-dimensional, d-dimensional) *polynomial curve* is a parameterized curve $u \mapsto P(u)$:

$$P(u) = \sum_{i=0}^{n} \mathbf{a}_i \cdot u^i = \mathbf{a}_n\, u^n + \cdots + \mathbf{a}_1\, u + \mathbf{a}_0 \quad \text{with} \quad \mathbf{a}_n, \mathbf{a}_{n-1}, \ldots, \mathbf{a}_1, \mathbf{a}_0 \in \mathbb{R}^2\ (\mathbb{R}^3, \mathbb{R}^d) \tag{2.1}$$

The set of all d-dimensional polynomial curves of degree n is denoted by $\mathbb{P}_n^d$.

Example:

$$\begin{pmatrix}0\\2\\1\end{pmatrix} + \begin{pmatrix}-1\\0\\3\end{pmatrix} u + \begin{pmatrix}1\\-2\\1\end{pmatrix} u^2 + \begin{pmatrix}7\\1\\5\end{pmatrix} u^3 = \begin{pmatrix}-u + u^2 + 7u^3\\ 2 - 2u^2 + u^3\\ 1 + 3u + u^2 + 5u^3\end{pmatrix}$$

Evaluation with equation 2.1 requires $n + (n-1) + \cdots + 1$ multiplications and n additions, thus it is of type $\mathcal{O}(n^2)$. A smarter way to evaluate the equation is to save and reuse the intermediary results of the multiplication of the parameter u. In this case we have $2n$ multiplications and n additions (complexity $\mathcal{O}(n)$). An even faster evaluation is possible using *Horner's Rule*:

Algorithm 2.1 (Horner's Rule)

$$P(u) = (\cdots((a_n \cdot u + a_{n-1}) \cdot u + a_{n-2}) \cdot u + \cdots) \cdot u + a_0$$

Complexity: $\mathcal{O}(n)$ with n multiplications and n additions. Pseudo code:

```
$p := a_n$
for $i := n - 1$ to $0$ do
    $p := p \cdot u + a_i$
end for
```

2.2 Different Basis of $\mathbb{P}_n$

Note that $\mathbb{P}$ and $\mathbb{P}_n$ are linear spaces (vector spaces).

$\dim(\mathbb{P}_n) = n+1$ (polynomial functions)

$$\dim(\mathbb{P}_n) = \begin{cases} 2\,(n+1) & \text{2D curve} \\ 3\,(n+1) & \text{3D curve} \\ d\,(n+1) & d\text{D curve} \end{cases}$$

Remark: The dimension of a space corresponds to the number of degrees of freedom in this space. These are the coefficients $a_0, a_1, ..., a_n$ for polynomial functions.

The standard basis for $\mathbb{P}_n$ is the *monomial basis* $1, u, u^2, \ldots, u^n$. Other possible basis for $\mathbb{P}_n$ are presented in the next sections.

2.2.1 Lagrange Polynomials

Definition 2.2
Given $n+1$ parameter values $t_0, \ldots, t_n \in \mathbb{R}$ with $t_0 < t_1 < \cdots < t_n$. The polynomial

$$L_i^n(u) := \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{u-t_j}{t_i-t_j} = \frac{(u-t_0)(u-t_1)\cdots(u-t_n)}{(t_i-t_0)(t_i-t_1)\cdots(t_i-t_n)}\,, \quad i = 0,\ldots,n \tag{2.2}$$

is called the i th Lagrange[1] polynomial of degree n.

Theorem 2.1
The Lagrange polynomials $L_0^n, L_1^n, \ldots, L_n^n$ form a basis of $\mathbb{P}_n$:

$$P(u) = \sum_{i=0}^{n} \mathbf{a}_i\, u^i = \sum_{i=0}^{n} L_i^n(u)\cdot \mathbf{p}_i\,, \quad \mathbf{p}_i \in \mathbb{R}^d \tag{2.3}$$

Sketch of Proof:

$$L_i^n(t_j) = \begin{cases} 1 & i=j \\ 0 & i\neq j \end{cases} \;\Rightarrow\; F(t_i) = \mathbf{p}_i\,, \quad i=0,\ldots,n$$

Interpretation: The points $\mathbf{p}_i$ are *interpolated* at parameter values t_i. There is only one polynomial of nth degree, that interpolates these $n+1$ points (number of conditions = number of degrees of freedom $\rightarrow$ unique solvable system of linear equations), therefore the Lagrange representation is unique. □

Example: Given $t_0 = 0,\ t_1 = \frac{1}{2},\ t_2 = 1$.

$$L_0^2(u) = \frac{(u-\frac{1}{2})\,(u-1)}{(0-\frac{1}{2})\,(0-1)} = 2\,(u-\tfrac{1}{2})\,(u-1)$$

[1] Joseph Louis de Lagrange (1736–1813)

$$L_1^2(u) = \tfrac{(u-0)\,(u-1)}{(\frac{1}{2}-0)\,(\frac{1}{2}-1)} = -4\,u\,(u-1)$$

$$L_2^2(u) = \tfrac{(u-0)\,(u-\frac{1}{2})}{(1-0)\,(1-\frac{1}{2})} = 2\,u\,(u-\tfrac{1}{2})$$

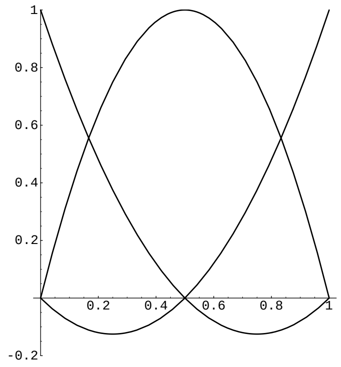


Figure 2.1: Quadratic Lagrange polynomials with respect to paramter values 0,$\frac{1}{2}$ and 1

2.2.2 Bernstein Polynomials

Definition 2.3
The polynomial

$$B_i^n(u) = \binom{n}{i}\,u^i\,(1-u)^{n-i}\;, \quad i = 0,\ldots,n \tag{2.4}$$

is called the i th Bernstein[2] polynomial of degree n.
Additional agreement: $B_i^n \equiv 0$, if $i < 0$ or $i > n$.

Remark: Properties of the binomial coefficients:

(1) definition: $\binom{n}{k} := \frac{n!}{k!(n-k)!}$

(2) recursive definition: $\binom{n}{0} := 1,\; \binom{n+1}{k+1} := \frac{n+1}{k+1}\binom{n}{k}$

(3) Pascal triangle: recursion: $\binom{n+1}{k+1} = \binom{n}{k+1} + \binom{n}{k}$ and symmetry: $\binom{n}{k} = \binom{n}{n-k}$

Theorem 2.2
Properties of Bernstein polynomials:

(1) $\sum_{i=0}^n B_i^n(u) = (u + (1-u))^n$ (Binomial Theorem: $(a+b)^n = \sum_{k=0}^n \binom{n}{k} a^k b^{n-k}$)

(2) $\sum_{i=0}^n B_i^n(u) = 1$ (partition of 1)

(3) $B_i^n(u) \geq 0\;,\; u \in [0,1]$ (positivity)

[2]Sergeĭ N. Bernstein, 1912

(4) $B_i^n(u) = u \cdot B_{i-1}^{n-1}(u) + (1-u) \cdot B_i^{n-1}(u)$ (recursion)

(5) $B_i^n(u) = B_{n-i}^n(1-u)$ (symmetry)

(6) $B_i^n(u)$ has a maximum in $[0,1]$ at $u = \frac{i}{n}$.

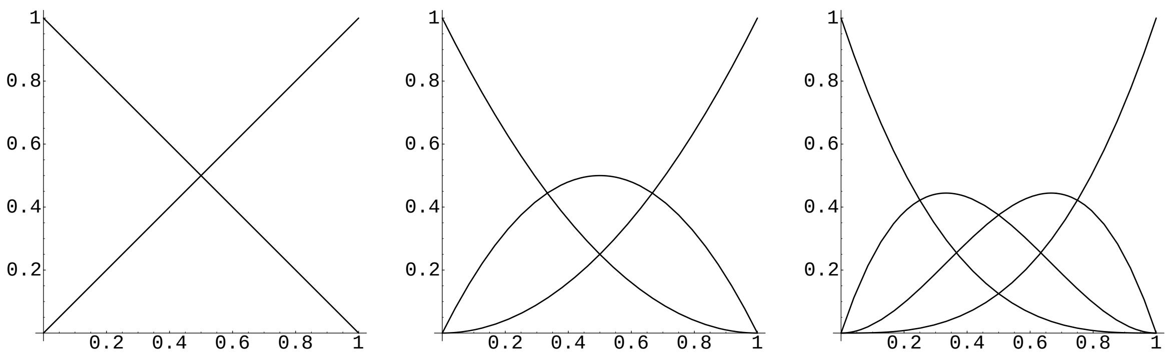


Figure 2.2: Bernstein polynomials for $n=1$, $n=2$ and $n=3$

Example:

$$B_0^2(u) = \binom{2}{0} u^0 (1-u)^{2-0} = (1-u)^2$$

$$B_1^2(u) = \binom{2}{1} u^1 (1-u)^{2-1} = 2\,u\,(1-u)$$

$$B_2^2(u) = \binom{2}{2} u^2 (1-u)^{2-2} = u^2$$

Definition 2.4 (General Bernstein Polynomial)
Given the parametric interval $\Delta := [s,t]$:

$$B_i^{\Delta,n}(u) = \frac{1}{(t-s)^n} \binom{n}{i} (u-s)^i \, (t-u)^{n-i} \tag{2.5}$$

is called the i th Bernstein polynomial for the interval $\Delta = [s,t]$.

Theorem 2.2 holds also true.

How we get the general Bernstein Polynomial:

$\varphi : [s,t] \to [0,1]$, $\varphi(u) = \frac{u-s}{t-s}$

Evaluating formula 2.3 at $\varphi(u)$ yields: $B_i^n(\varphi(u)) = B_i^{\Delta,n}(u)$

Remark: Note that $u = \alpha \cdot t + (1-\alpha) \cdot s$ with $\alpha = \frac{u-s}{t-s}$.
u is represented here in *barycentric coordinates* with respect to s and t.

Theorem 2.3
The Bernstein polynomials $B_0^n, \ldots, B_n^n$ form a basis of $\mathbb{P}_n$:

$$P(u) = \sum_{i=0}^{n} B_i^n(u) \cdot \mathbf{b}_i\,, \quad \mathbf{b}_i \in \mathbb{R}^d \tag{2.6}$$

2.2.3 Comparison of Lagrange and Bernstein Representation

Different representations: $F(u) = \sum_i \mathbf{a}_i\, u^i = \sum_i \mathbf{p}_i\, L_i^n(u) = \sum_i \mathbf{b}_i\, B_i^n(u)$

- $\mathbf{p}_i$ will be interpolated $(F(t_j) = \sum_i \mathbf{p}_i\, L_i^n(t_j) = \mathbf{p}_j)$.
- $\mathbf{b}_0,\ \mathbf{b}_n$ will be interpolated $(F(0) = \sum_i \mathbf{b}_i\, B_i^n(0) = \mathbf{b}_0)$.
- The location of $\mathbf{p}_0, \ldots, \mathbf{p}_n$ or $\mathbf{b}_0, \ldots, \mathbf{b}_n$ determine the shape of the curve “somehow intuitively”. This is not true for the monomial coefficients $\mathbf{a}_i$.

2.2.4 Change of Basis

How to switch from one representation to another? A general rule from Linear Algebra claims:

Theorem 2.4
Given two linear combinations representing one vector $\mathbf{v} = \sum_i \beta_i\, \mathbf{b}_i = \sum_i \delta_i\, \mathbf{d}_i$:

$$\begin{pmatrix} \vdots \\ \mathbf{b}_i \\ \vdots \end{pmatrix} = \mathbf{A} \begin{pmatrix} \vdots \\ \mathbf{d}_i \\ \vdots \end{pmatrix} \quad \Rightarrow \quad \begin{pmatrix} \vdots \\ \delta_i \\ \vdots \end{pmatrix} = \mathbf{A}^T \begin{pmatrix} \vdots \\ \beta_i \\ \vdots \end{pmatrix} \tag{2.7}$$

Proof:

$$\begin{aligned} \mathbf{b}_i &= \sum_j a_{i\,j}\, \mathbf{d}_j \\ \Rightarrow\ & \sum_i \beta_i \sum_j a_{i\,j}\, \mathbf{d}_j = \sum_j \left(\sum_i \beta_i\, a_{i\,j} \right) \mathbf{d}_j \overset{!}{=} \sum_j \delta_j\, \mathbf{d}_j \\ \Rightarrow\ & \delta_j = \sum_i a_{i\,j}\, \beta_i \end{aligned}$$

□

Example:

$$\begin{pmatrix} B_0^2 \\ B_1^2 \\ B_2^2 \end{pmatrix} = \begin{pmatrix} 1 & -2 & 1 \\ 0 & 2 & -2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ u \\ u^2 \end{pmatrix} \quad \Rightarrow \quad \begin{pmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \mathbf{a}_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 2 & 0 \\ 1 & -2 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{b}_0 \\ \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix}$$

Supplement: Affine Maps

Definition 2.5
A map $\varphi : \mathbb{R}^m \to \mathbb{R}^d$ is called *affine*, if

$$\varphi\left(\sum_i \alpha_i\, \mathbf{x}_i \right) = \sum_i \alpha_i\, \varphi(\mathbf{x}_i) \tag{2.8}$$

for all α_i with $\sum_i \alpha_i = 1.$ The linear combination $\sum_i \alpha_i \, \mathbf{x}_i$ is called *affine combination* in this case. If additionally $\alpha_i \geq 0$ holds true for all i, then it is called *convex combination*.

An affine map especially preserves affine combinations, i. e.

$$\varphi\left((1-\alpha)\,\mathbf{p}_1 + \alpha\,\mathbf{p}_2\right) = (1-\alpha)\,\varphi(\mathbf{p}_1) + \alpha\,\varphi(\mathbf{p}_2) \tag{2.9}$$

Definition 2.6
A map $\varphi : \mathbb{R}^m \to \mathbb{R}^d$ is called *multiaffine*, if for any fixed arguments $\bar{x}_j$ the map

$$x \mapsto \varphi_k(\bar{x}_1, \dots, \bar{x}_{k-1}, x, \bar{x}_{k+1}, \dots, \bar{x}_m), \quad j \neq k,\ 1 \leq j \leq m \tag{2.10}$$

is affine. (Diction: $m = 2 \to$ “bi-affine”, $m = 3 \to$ “tri-affine” etc.)

Example:

$\varphi : (x_1, x_2) \ \mapsto\ x_1 + x_2$ is multiaffine.

$\rho : (x_1, x_2) \ \mapsto\ x_1 \cdot x_2$ is multiaffine.

$\sigma : (x_1, x_2) \ \mapsto\ x_1^2 + x_2^2$ is not multiaffine.

Theorem 2.5 (Characterization of Affine Maps)
φ is affine $\Leftrightarrow$ a linear map $\mathbf{A}$ and a translation vector $\mathbf{b}$ exist such that $\varphi(\mathbf{x}) = \mathbf{A}(\mathbf{x}) + \mathbf{b}$.

Remark:

- affine maps map lines onto lines
- parallelity is preserved
- division ratios are preserved
- angles are *not* preserved
- examples for affine maps: rotation, scaling, shearing . . .

2.3 Polar Forms

Theorem 2.6 (Polynomials and Polar Forms)
For each polynomial $F : \mathbb{R} \to \mathbb{R}^d$ of degree n exists a unique map $f : \mathbb{R}^n \to \mathbb{R}^d$ with the following properties:

(1) f is *n-affine*:

$$f(x_1, \dots, x_{i-1}, \sum_j \alpha_j \, y_j, x_{i+1}, \dots, x_n) =$$
$$= \sum_j \alpha_j \, f(x_1, \dots, x_{i-1}, y_j, x_{i+1}, \dots, x_n) \quad \text{if } \sum_j \alpha_j = 1$$

(2) f is *symmetric* in each variable, i. e.

$$\begin{aligned} & f(x_1,\dots,x_{i-1},x_i,x_{i+1},\dots,x_{j-1},x_j,x_{j+1},\dots,x_n) = \\ = & f(x_1,\dots,x_{i-1},x_j,x_{i+1},\dots,x_{j-1},x_i,x_{j+1},\dots,x_n) \end{aligned}$$

(3) $F(u) = f(\underbrace{u,\dots,u}_{n})$ (diagonal property)

Diction: f is the *polar form* (“blossom”) of F.

Example:

	$F(u)$	$f(u_1,\dots,u_n)$
$n=1$	1	1
	u	u_1
$n=2$	1	1
	u	$\frac{1}{2}(u_1+u_2)$
	u^2	$u_1\,u_2$
	$\mathbf{a}_0+\mathbf{a}_1\,u+\mathbf{a}_2\,u^2$	$\mathbf{a}_0+\mathbf{a}_1\,\frac{1}{2}(u_1+u_2)+\mathbf{a}_2\,u_1\,u_2$
$n=3$	1	1
	u	$\frac{1}{3}(u_1+u_2+u_3)$
	u^2	$\frac{1}{3}(u_1\,u_2+u_1\,u_3+u_2\,u_3)$
	u^3	$u_1\,u_2\,u_3$
	$\dots$	$\dots$

General formula for polynomials of degree n in monomial representation: $F(u)=\sum_{i=0}^n \mathbf{a}_i\,u^i$:

$$f(u_1,\dots,u_n) = \sum_{i=0}^n \mathbf{a}_i\,\frac{1}{\binom{n}{i}} \sum_{\substack{\sigma\subseteq\{1,\dots,n\}\\ |\sigma|=i}} \prod_{j\in\sigma} u_j \tag{2.11}$$

Example: $F(u)=\begin{pmatrix} u \\ u^2 \end{pmatrix}$, $\quad f(u_1,u_2)=\begin{pmatrix} 1/2\,(u_1+u_2) \\ u_1\,u_2 \end{pmatrix}$

$f(0,0)=\begin{pmatrix} 0 \\ 0 \end{pmatrix}$, $\quad f(0,1)=f(1,0)=\begin{pmatrix} 1/2 \\ 0 \end{pmatrix}$, $\quad f(1,1)=\begin{pmatrix} 1 \\ 1 \end{pmatrix}$.

Consider a degree 2 polynomial curve: $F(u)=f(u,u)$. This can be decomposed in the following way:

$$\begin{aligned} f(u,u) &= f((1-u)\cdot 0+u\cdot 1,u) = (1-u)\cdot f(0,u)+u\cdot f(1,u) = \\ &= (1-u)[(1-u)\,f(0,0)+u\,f(0,1)]+u\,[(1-u)\,f(1,0)+u\,f(1,1)] = \\ &= (1-u)^2\,f(0,0)+2\,u\,(1-u)\,f(0,1)+u^2\,f(1,1) = \\ &= B_0^2(u)\,f(0,0)+B_1^2(u)\,f(0,1)+B_2^2(u)\,f(1,1) \end{aligned}$$

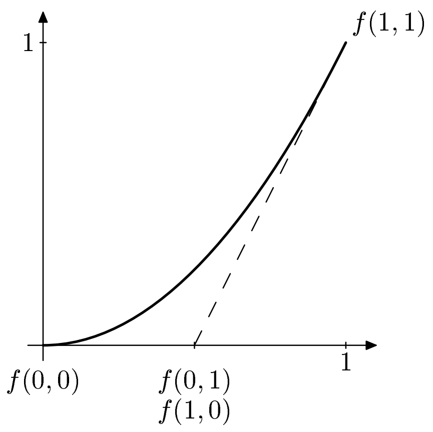
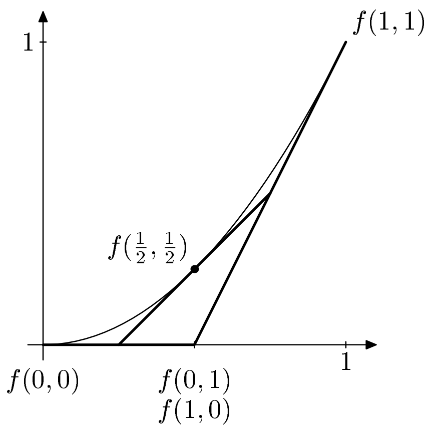


Figure 2.3: Polar Form of a Curve of degree $n = 2$

Corollary 2.1
$f(0,0),\ f(0,1),\ f(1,1)$ are coefficients with respect to the Bernstein basis. With degree n we have coefficients $f(\underbrace{0,\dots,0}_{n-k},\underbrace{1,\dots,1}_{k})$, $0 \le k \le n$.

Corollary 2.2
This gives reason to a geometric construction to get the point corresponding to the curve parameter $u = \frac{1}{2}$:



This geometric construction is called the *de Casteljau algorithm*. (see section 3.2.1).

3 Bézier Curves

3.1 Basics

Definition 3.1 (Bézier Curve)
A polynomial curve

$$F(u) = \sum_{i=0}^{n} B_i^{\Delta,n}(u) \cdot \mathbf{b}_i \,, \quad \mathbf{b}_i \in \mathbb{R}^d \tag{3.1}$$

is called *Bézier curve*[1] of degree n over $\Delta = [s,t]$ with the gereralized Bernstein polynomial $B_i^{\Delta,n}(u)$. The points $\mathbf{b}_i$ are called *control points* or *Bézier points*, the polygon formed by the control points is called *control polygon*.

Theorem 3.1 (Bézier Points and Polar Form)
Given a polynomial $F : \mathbb{R} \to \mathbb{R}^d$ of degree n, the Bézier points with respect to $\Delta = [s,t]$ are

$$\mathbf{b}_i = f(\underbrace{s,\dots,s}_{n-i},\underbrace{t,\dots,t}_{i}) \,,\ i = 0,\dots,n. \tag{3.2}$$

Proof: Write u as an affine combination of s and t:

$$u = \left(\frac{u-s}{t-s}\right) t + \left(\frac{t-u}{t-s}\right) s$$

Now we have:

$$\begin{aligned}
F(u) = f(\underbrace{u,\dots,u}_{n}) &= \left(\frac{u-s}{t-s}\right) f(t,\underbrace{u,\dots,u}_{n-1}) + \left(\frac{t-u}{t-s}\right) f(s,\underbrace{u,\dots,u}_{n-1}) = \\
&= \left(\frac{u-s}{t-s}\right)^2 f(t,t,\underbrace{u,\dots,u}_{n-2}) + \left(\frac{u-s}{t-s}\right)\left(\frac{t-u}{t-s}\right) f(t,s,\underbrace{u,\dots,u}_{n-2}) + \\
&+ \left(\frac{t-u}{t-s}\right)\left(\frac{u-s}{t-s}\right) f(s,t,\underbrace{u,\dots,u}_{n-2}) + \left(\frac{t-u}{t-s}\right)^2 f(s,s,\underbrace{u,\dots,u}_{n-2}) = \\
&= \dots \quad \text{decompose } u\text{'s} \\
&= \sum_{i=0}^{n} \binom{n}{i} \left(\frac{u-s}{t-s}\right)^i \left(\frac{t-u}{t-s}\right)^{n-i} f(\underbrace{s,\dots,s}_{n-i},\underbrace{t,\dots,t}_{i}) \\
&= \sum_{i=0}^{n} B_i^{\Delta,n}(u)\, f(\underbrace{s,\dots,s}_{n-i},\underbrace{t,\dots,t}_{i})
\end{aligned}$$

□

[1]Pierre Bézier, 1966

Example: Given: $F(u) = (u, u^2)$, $f(u_1, u_2) = (\frac{1}{2}(u_1 + u_2), u_1\, u_2)$, $\Delta = [-1, 1]$.
$\mathbf{b}_0 = f(-1, 1) = (-1, 1), \quad \mathbf{b}_1 = f(-1, 1) = (0, -1), \quad \mathbf{b}_2 = f(1, 1) = (1, 1)$

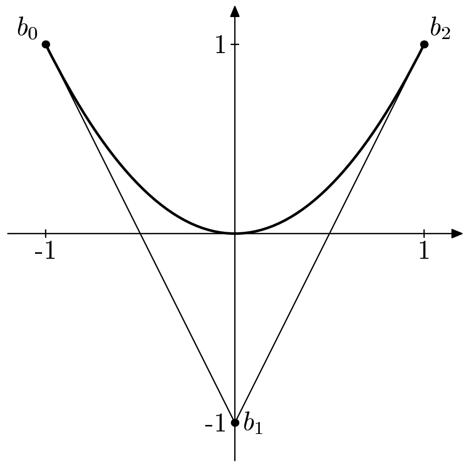


Figure 3.1: A Bézier curve

Theorem 3.2 (Derivative of a Bézier Curve)
Let $F(u) = \sum_{i=0}^{n} B_i^{\Delta,n}(u)\, \mathbf{b}_i$ a Bézier curve over $\Delta = [s, t]$. Then:

$$F'(u) = \frac{n}{t-s} \sum_{i=0}^{n-1} B_i^{\Delta,n-1}(u) \cdot (\mathbf{b}_{i+1} - \mathbf{b}_i) \tag{3.3}$$

Proof: Consider $B_i^{\Delta,n}(u) := \binom{n}{i} \left(\frac{u-s}{t-s}\right)^i \left(\frac{t-u}{t-s}\right)^{n-i}$

(i)
$$\begin{aligned}(B_i^{\Delta,n})'(u) &= \frac{n!}{i!(n-i)!}\, \frac{i}{t-s} \left(\frac{u-s}{t-s}\right)^{i-1} \left(\frac{t-u}{t-s}\right)^{n-i} - \\ &- \frac{n!}{i!(n-i)!} \left(\frac{u-s}{t-s}\right)^{i} \frac{n-i}{t-s} \left(\frac{t-u}{t-s}\right)^{n-i-1} = \\ &= \frac{n}{t-s} \left[\frac{(n-1)!}{(i-1)!(n-1-(i-1))!} \left(\frac{u-s}{t-s}\right)^{i-1} \left(\frac{t-u}{t-s}\right)^{(n-1)-(i-1)} - \right. \\ &\left. - \frac{(n-1)!}{i!(n-1-i)!} \left(\frac{u-s}{t-s}\right)^{i} \left(\frac{t-u}{t-s}\right)^{n-1-i} \right] = \\ &= \frac{n}{t-s} \left(B_{i-1}^{\Delta,n-1}(u) - B_i^{\Delta,n-1}(u) \right)\end{aligned}$$

(ii) $$F'(u) = \sum_{i=0}^{n} (B_i^{\Delta,n})'(u)\,\mathbf{b}_i \overset{(1)}{=} \frac{n}{t-s}\left(\sum_{j=1}^{n} B_{j-1}^{\Delta,n-1}(u)\,\mathbf{b}_j - \sum_{i=0}^{n-1} B_i^{\Delta,n-1}(u)\,\mathbf{b}_i\right) =$$
$$= \frac{n}{t-s}\left(\sum_{i=0}^{n-1} B_i^{\Delta,n-1}(u)\,\mathbf{b}_{i+1} - \sum_{i=0}^{n-1} B_i^{\Delta,n-1}(u)\,\mathbf{b}_i\right) =$$
$$= \frac{n}{t-s}\sum_{i=0}^{n-1} B_i^{\Delta,n-1}(u)\,(\mathbf{b}_{i+1} - \mathbf{b}_i)$$

□

Corollary 3.1

$$F''(u) = \frac{n\,(n-1)}{(t-s)^2}\sum_{i=0}^{n-2} B_i^{\Delta,n-2}(u)\,(\mathbf{b}_{i+2} - 2\mathbf{b}_{i+1} + \mathbf{b}_i) \tag{3.4}$$

etc. for higher derivatives.

3.1.1 Shape Properties for Bézier Curves

Theorem 3.3 (Lemma of Bézier)
For $F(u) = \sum_{i=0}^{n} B_i^{\Delta,n}(u)\,\mathbf{b}_i$ Bézier curve over $\Delta = [s,t]$ we have:

(1) For $s \le u \le t$, $F(u)$ is contained in the closed convex hull of the control polygon. (The curve point $F(u)$ is a convex combination of the $\{\mathbf{b}_i\}$).

Remark: A set of points M is called *convex*, if for any two points $\mathbf{p}_1,\, \mathbf{p}_2 \in M$ the line $\overline{\mathbf{p}_1\mathbf{p}_2} \in M$.

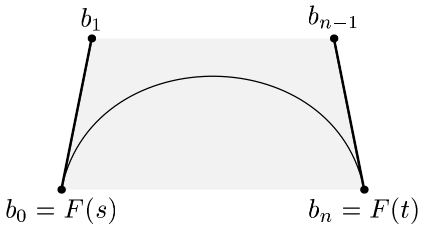


Figure 3.2: Convex hull

(2) End-point interpolation: $F(s) = \mathbf{b}_0$ and $F(t) = \mathbf{b}_n$, i. e. the curve runs through the first and the last control point.

(3) Tangency condition: In the end-points the curve is tangent to the control polygon. $F'(s) = \frac{n}{t-s}(\mathbf{b}_1 - \mathbf{b}_0)$ und $F'(t) = \frac{n}{t-s}(\mathbf{b}_n - \mathbf{b}_{n-1})$

(4) Affine invariance: Let $\varphi(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b}$ an affine map, then

$$\varphi(F(u)) = \sum_{i=0}^{n} B_i^{\Delta,n}(u)\,\varphi(\mathbf{b}_i).$$

I. e. to transform the curve with the affine transformation φ, it is sufficient to transform the control points.

(5) Variation diminishing property:
"A Bézier curve does not vary more than its control polygon".
More precisely: For any hyperplane H in $\mathbb{R}^d$ (line in $\mathbb{R}^2$, plane in $\mathbb{R}^3$ etc.), then

$$\#(H \cap F) \leq \#(H \cap \text{control polygon}).$$

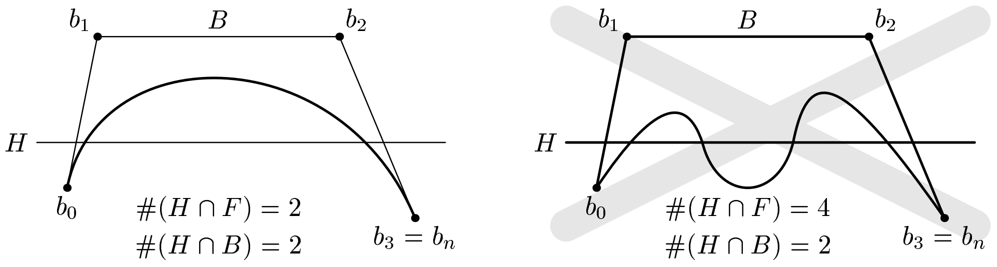


Figure 3.3: Variation diminishing property

Proof:

(1) convex hull: $[\mathbf{b}_0, \ldots, \mathbf{b}_n] := \{\sum_{i=0}^{n} \alpha_i \mathbf{b}_i \mid \sum_i \alpha_i = 1,\ \alpha_i \geq 0\}$

For $s \leq u \leq t$ set $\alpha_i = B_i^{\Delta,n}(u)$

$\Rightarrow\ \sum_i \alpha_i = 1,\ \alpha_i \geq 0$

$\Rightarrow\ F(u) = \sum_i \alpha_i\, \mathbf{b}_i\ \in\ [\mathbf{b}_0, \ldots, \mathbf{b}_n]$

(2) $F(s) = f(\underbrace{s, \ldots, s}_{n}) = \mathbf{b}_0$ und $F(t) = f(\underbrace{t, \ldots, t}_{n}) = \mathbf{b}_n$

(Theorem 3.1, diagonal property of the polar form)

(3) $F'(u) = \frac{n}{t-s} \sum_i B_i^{\Delta,n-1}(u)(\mathbf{b}_{i+1} - \mathbf{b}_i)$

$\Rightarrow\ F'(s) = \frac{n}{t-s}(\mathbf{b}_1 - \mathbf{b}_0)$ because $B_i^{\Delta,n-1}(s) = \begin{cases} 1 & , i = 0 \\ 0 & , \text{sonst} \end{cases}$

The same for $F'(t)$.

(4) Let $\varphi(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{y}$ an affine map $\mathbb{R}^d \to \mathbb{R}^d$.

$$\varphi(F(u)) = \varphi\left(\sum_{i=0}^{n} B_i^{\Delta,n}(u)\,\mathbf{b}_i\right) = \mathbf{A}\left(\sum_{i=0}^{n} B_i^{\Delta,n}(u)\,\mathbf{b}_i\right) + \mathbf{y}\sum_{i=0}^{n} B_i^{\Delta,n}(u) =$$

$$= \sum_{i=0}^{n} B_i^{\Delta,n}(u)\,(\mathbf{A}(\mathbf{b}_i) + \mathbf{y}) = \sum_{i=0}^{n} B_i^{\Delta,n}(u)\,\varphi(\mathbf{b}_i)$$

(5) see [4]. □

3.2 Algorithms for Bézier Curves

3.2.1 The de Casteljau Algorithm

The de Casteljau algorithm is a method for evaluating Bézier curves.
Given: control points $\mathbf{b}_0, \dots, \mathbf{b}_n$ of a Bézier curve $\sum_{i=0}^{n} B_i^{\Delta,n}(u)\,\mathbf{b}_i$.

Algorithm 3.1 (de Casteljau)

$$\begin{aligned} \mathbf{b}_i^0 &:= \mathbf{b}_i \\ \mathbf{b}_i^j &:= \frac{t-u}{t-s}\,\mathbf{b}_i^{j-1} + \frac{u-s}{t-s}\,\mathbf{b}_{i+1}^{j-1} \qquad i = 0, \dots, n-l \end{aligned} \tag{3.5}$$

Then: $\mathbf{b}_0^n = F(u)$ is a point on the Bézier curve.
Remark: The points $\mathbf{b}_i^j$ depend on parameter u for $1 \le i \le n-l$.

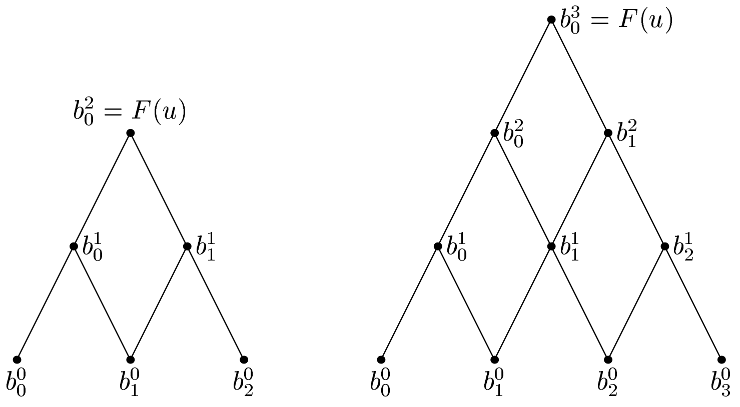


Figure 3.4: de Casteljau schemas for $n = 2$ and $n = 3$

Sketch of Proof: The proof of theorem 3.1 on page 11 is just the application of de Casteljau's algorithm. Only the intermediate results are renamed:

$$\mathbf{b}_i^j = f(\underbrace{u, \dots, u}_{j}, \underbrace{s, \dots, s}_{n-j-i}, \underbrace{t, \dots, t}_{i}).$$

□

Graphical representation of the de Casteljau algorithm for $n = 2$, $n = 3$ and $\frac{u-s}{t-s} = \frac{1}{3}$:

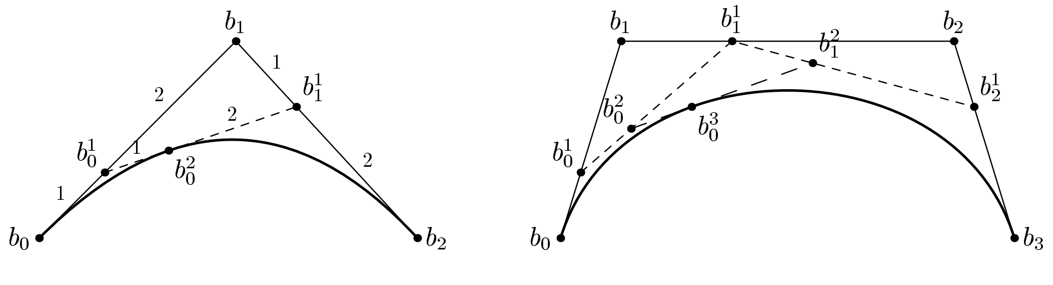


Figure 3.5: Graphical representation of the de Casteljau algorithm

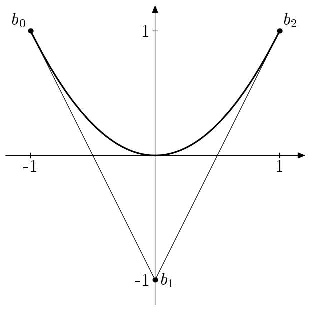
Example: $F(u) = \begin{pmatrix} u \\ u^2 \end{pmatrix}$, $\quad f(u_1, u_2) = \begin{pmatrix} 1/2\,(u_1 + u_2) \\ u_1\, u_2 \end{pmatrix}$

Bézier representation with respect to $\Delta = [-1, 1]$:

$$\mathbf{b}_0 = f(-1,-1) = \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \quad \mathbf{b}_1 = f(-1,1) = \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \quad \mathbf{b}_2 = f(1,1) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Evaluation of $F(u)$ with de Casteljau at $u = 0$:

$$\mathbf{b}_0^1 = \tfrac{1}{2}\mathbf{b}_0 + \tfrac{1}{2}\mathbf{b}_1 = \begin{pmatrix} -1/2 \\ 0 \end{pmatrix}, \quad \mathbf{b}_1^1 = \tfrac{1}{2}\mathbf{b}_1 + \tfrac{1}{2}\mathbf{b}_2 = \begin{pmatrix} 1/2 \\ 0 \end{pmatrix}, \quad \mathbf{b}_0^2 = \tfrac{1}{2}\mathbf{b}_0^1 + \tfrac{1}{2}\mathbf{b}_1^1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$



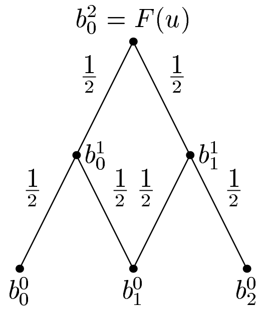


Figure 3.6: Evaluation of $F(0)$ with de Casteljau

We can compute the polar form of any value using the de Casteljau algorithm. Note that due to the symmetry of the polar form, there are two possibilities.

Find: $f(0, \frac{1}{2})$

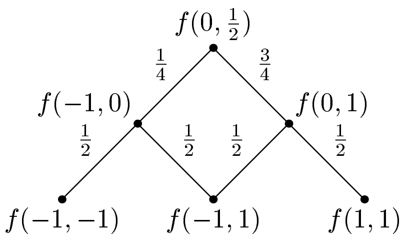
(1) $0 = \frac{1}{2}(-1) + \frac{1}{2}(1) \quad \frac{1}{2} = \frac{1}{4}(-1) + \frac{3}{4}(1)$

de Casteljau: evaluate first 0, then $\frac{1}{2}$:

$$f(-1,0) = \tfrac{1}{2}\, f(-1,-1) + \tfrac{1}{2}\, f(-1,1) = \begin{pmatrix} -1/2 \\ 0 \end{pmatrix}$$

$$f(0,1) = \tfrac{1}{2}\, f(-1,1) + \tfrac{1}{2}\, f(1,1) = \begin{pmatrix} 1/2 \\ 0 \end{pmatrix}$$

$$f(0,\tfrac{1}{2}) = \tfrac{1}{4}\, f(-1,0) + \tfrac{3}{4}\, f(0,1) = \begin{pmatrix} 1/4 \\ 0 \end{pmatrix}$$

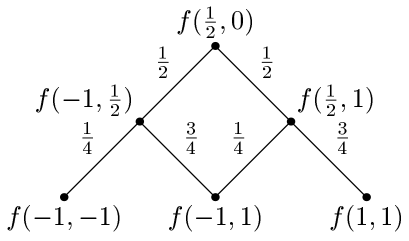


(2) symmetric variation: first $\frac{1}{2}$, then 0

$$f(-1,\tfrac{1}{2}) = \tfrac{1}{4}\, f(-1,-1) + \tfrac{3}{4}\, f(-1,1) = \begin{pmatrix} -1/4 \\ -1/2 \end{pmatrix}$$

$$f(\tfrac{1}{2},1) = \tfrac{1}{4}\, f(-1,1) + \tfrac{3}{4}\, f(1,1) = \begin{pmatrix} 3/4 \\ 1/2 \end{pmatrix}$$

$$f(\tfrac{1}{2},0) = \tfrac{1}{2}\, f(-1,\tfrac{1}{2}) + \tfrac{1}{2}\, f(\tfrac{1}{2},1) = \begin{pmatrix} 1/4 \\ 0 \end{pmatrix}$$



Remark:

- The de Casteljau algorithm is stable (only evaluation of convex combinations).
- Complexity is $\mathcal{O}(n^2)$ (in contrast to $\mathcal{O}(n)$ with Horner's rule).
- Improvement is possible by recursive subdivision.

3.2.2 The de Casteljau Subdivision Algorithm

Theorem 3.4
Let $F(u) = \sum_{i=0}^{n} B_i^{\Delta,n}(u)\, \mathbf{b}_i$ a Bézier curve over $\Delta = [s,t]$ and $q \in [s,t]$.
Define:

$$F_l\,(u) = \sum_{i=0}^{n} B_i^{\Delta_l,n}(u)\, \mathbf{b}_0^i(q) \quad \text{Bézier curve over } \Delta_l = [s,q]$$

$$F_r\,(u) = \sum_{i=0}^{n} B_i^{\Delta_r,n}(u)\, \mathbf{b}_i^{n-i}(q) \quad \text{Bézier curve over } \Delta_r = [q,t]$$

where $\mathbf{b}_i^l(q)$ are points of the de Casteljau algorithm evaluated at q. Then:

$$F(u) = \begin{cases} F_l(u), & u \in [s,q] \\ F_r(u), & u \in [q,t] \end{cases} \tag{3.6}$$

Proof: According to the proof of theorem 3.1 the left part of F is:

$$F_l(u) = \sum_{i=0}^{n} B_i^{\Delta_l,n}(u)\, f(\underbrace{s,\ldots,s}_{n-i},\underbrace{q,\ldots,q}_{i}) = \sum_{i=0}^{n} B_i^{\Delta_l,n}(u)\, b_0^i(q)$$

($b_0^i(q)$ are Bézier points with respect to $[s,q]$). Similar for the right part. $\square$

Example: According to figure 3.5 set $n = 3$ and $\frac{u-s}{t-s} = \frac{1}{3}$. See figure 3.7 for evaluation of the de Casteljau algorithm. The control points of the Bézier curve over $[s,q]$ are $\mathbf{b}_0^0, \dots, \mathbf{b}_0^3$.

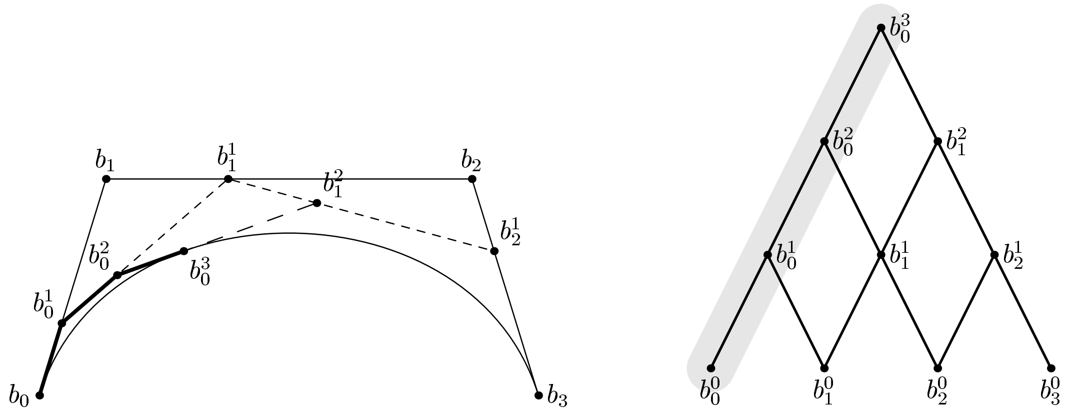


Figure 3.7: Left partial curve after de Casteljau subdivision

By repeated subdivision (e. g. uniform subdivision at the midpoints) we get a sequence of polygons that converges *fast* to the Bézier curve.

Lemma 3.1
Let $F : [s,t] \to \mathbb{R}^d$ a C^2-continuous curve (i. e. two times differentiable).
Let $L : [s,t] \to \mathbb{R}^d$ a line defined by $L(u) := \frac{u-s}{t-s} F(t) + \frac{t-u}{t-s} F(s)$. Then:

$$\|F(u) - L(u)\| \leq \frac{1}{8}(t-s)^2 \max_{\xi \in [s,t]} \|F''(\xi)\| \tag{3.7}$$

(Proof follows from Taylor expansion.)
Consequence: The uniform subdivision for Bézier curves converges (at least) quadratically.
Remark: The evaluation of Bézier curves at equidistant parameter values converges quadratically, too.

3.2.3 How to draw a Bézier Curve

Given: degree n, $\Delta = [s,t]$, $\mathbf{b}_0, \dots, \mathbf{b}_n$
Find: polygon for piecewise linear approximation of $F(u)$.

Algorithm 3.2
(1) Evaluate $F(u)$ at $m+1$ parameter values q_j with $q_j < q_{j+1}$ (e. g. with de Casteljau or by evaluation of Berstein polynomials).

(2) Draw the polygon $F(q_0), \dots, F(q_m)$.

Example: $n = 3, \quad \Delta = [0,1], \quad q_0 = 0,\ q_1 = \frac{1}{4},\ q_2 = \frac{1}{2},\ q_3 = \frac{3}{4},\ q_4 = 1$

Algorithm 3.3
(1) Subdivide $F(u)$ at the parametrical midpoint ($q = \frac{s+t}{2}$) in $F_l(u)$ and $F_r(u)$ using de Casteljau.

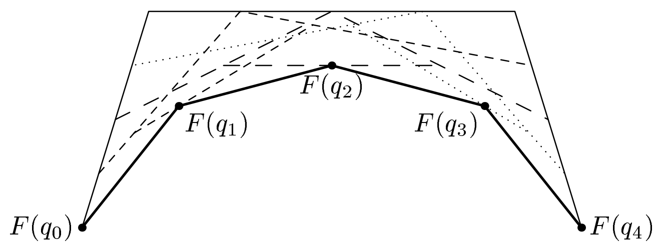


Figure 3.8: Multiple subdivision

(2) Continue recursively until a certain recursion depth r.

(3) Draw the control polygon of the last recursion step of (2).

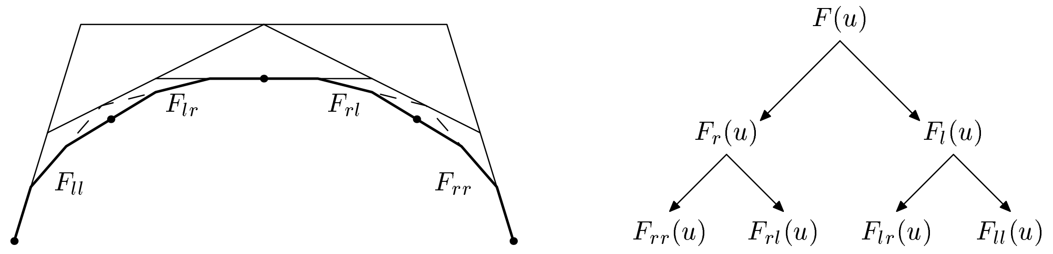


Figure 3.9: Recursive subdivision

3.2.4 Degree Elevation

Theorem 3.5
Let $F(u) = \sum_{i=0}^{n} B_i^{\Delta,n}(u)\,\mathbf{b}_i$ a Bézier curve of degree n over $\Delta = [s,t]$.

Assume $F(u) = \sum_{i=0}^{n+1} B_i^{\Delta,n+1}(u)\,\mathbf{b}_i^*$ as Bézier curve of degree $n+1$ over $\Delta = [s,t]$. Then:

$$\mathbf{b}_i^* = \frac{i}{n+1}\,\mathbf{b}_{i-1} + (1 - \frac{i}{n+1})\,\mathbf{b}_i\,, \quad i = 0, \dots, n+1 \tag{3.8}$$

(where $\mathbf{b}_{-1} = \mathbf{b}_{n+1} = \mathbf{0}$)

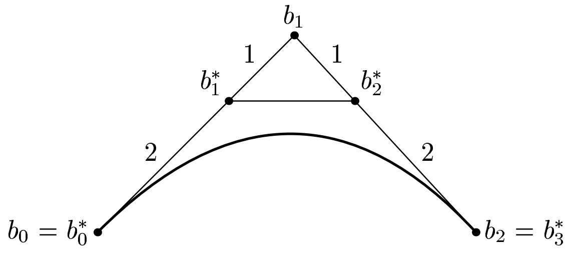


Figure 3.10: Degree elevation $n = 2 \rightarrow n = 3$

Proof:

(i) Let $f_n(u_1, \dots, u_n)$ the n-affine polar form of F (as polynomial of degree n).

Definition 3.2

$$f^*(u_1, \dots, u_{n+1}) := \frac{1}{n+1} \sum_{j=1}^{n+1} f_n(u_1, \dots, u_{j-1}, u_{j+1}, \dots, u_{n+1}) \tag{3.9}$$

(a) affine in every u
(b) symmetric
(c) diagonal

$\Rightarrow$ f^* is the polar form of $F(u)$ (as polynomial of degree $n+1$).

(ii)

$$\begin{aligned} \mathbf{b}_i^* &= f^*(\underbrace{s, \dots, s}_{n+1-i}, \underbrace{t, \dots, t}_{i}) = f^*(\underbrace{t, \dots, t}_{i}, \underbrace{s, \dots, s}_{n+1-i}) = \\ &= \frac{1}{n+1}\left(\sum_{j=1}^{i} f_n(\underbrace{t, \dots, t}_{i-1}, \underbrace{s, \dots, s}_{n-(i-1)}) + \sum_{j=i+1}^{n+1} f_n(\underbrace{t, \dots, t}_{i}, \underbrace{s, \dots, s}_{n-i}) \right) = \\ &= \frac{1}{n+1}\,(i\,\mathbf{b}_{i-1} + (n+1-i)\,\mathbf{b}_i) = \frac{i}{n+1}\mathbf{b}_{i-1} + (1 - \frac{i}{n+1})\,\mathbf{b}_i \qquad \square \end{aligned}$$

Example: $n = 3 \to n = 4$

$$\mathbf{b}_0^* = \frac{1}{4}(0 \cdot \mathbf{b}_{-1} + 4 \cdot \mathbf{b}_0) \qquad \mathbf{b}_1^* = \frac{1}{4}(1 \cdot \mathbf{b}_0 + 3 \cdot \mathbf{b}_1)$$
$$\mathbf{b}_2^* = \frac{1}{4}(2 \cdot \mathbf{b}_1 + 2 \cdot \mathbf{b}_2) \qquad \mathbf{b}_3^* = \frac{1}{4}(3 \cdot \mathbf{b}_2 + 1 \cdot \mathbf{b}_3)$$
$$\mathbf{b}_4^* = \frac{1}{4}(4 \cdot \mathbf{b}_3 + 0 \cdot \mathbf{b}_4)$$

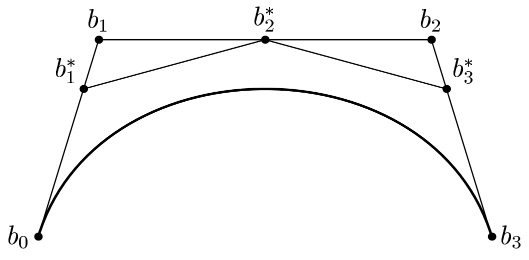


Figure 3.11: Degree elevation $n = 3 \to n = 4$

Remark:

- The repeated degree elevation yields a sequence of polygons then converge *slowly* to the Bézier curve.
- One can use this fact to prove the variation diminishing property of Bézier curves (see [4]).

3.2.5 Degree Reduction

Given: a Bézier curve $F(u) = \sum_{i=0}^{n} B_i^n(u)\,\mathbf{b}_i$.

Find: a representation with reduced degree: $F(u) = \sum_{i=0}^{n-1} B_i^{n-1}\,\check{\mathbf{b}}_i$.

This is not possible in general; therefore find the best approximation $\check{F}$. The degree elevation of $\check{F}$ yields $\check{F}(u) = \sum_{i=0}^{n} B_i^n(u)\,\check{\mathbf{b}}_i^*$, where in general $\mathbf{b}_i \neq \check{\mathbf{b}}_i^*$. Find control points $\{\check{\mathbf{b}}_i\}$ such that

$$\sum_{i=0}^{n} \|\check{\mathbf{b}}_i^* - \mathbf{b}_i\|^2 = \min .$$

1st Method of Solution:
Evaluation of equation 3.8 yields:

$$\check{\mathbf{b}}_i^* = \frac{i}{n}\check{\mathbf{b}}_{i-1} + \frac{n-i}{n}\check{\mathbf{b}}_i$$
$$\Rightarrow \sum_{i=0}^{n} \left\| \frac{i}{n}\check{\mathbf{b}}_{i-1} + \frac{n-i}{n}\check{\mathbf{b}}_i - \mathbf{b}_i \right\|^2 = \min .$$

This leaves a linear system to solve.
Example: $n = 3 \to n = 2$

Given: control points $\mathbf{b}_0 = \begin{pmatrix}0\\0\end{pmatrix}, \quad \mathbf{b}_1 = \begin{pmatrix}0\\20\end{pmatrix}, \quad \mathbf{b}_2 = \begin{pmatrix}40\\20\end{pmatrix}, \quad \mathbf{b}_3 = \begin{pmatrix}40\\0\end{pmatrix}$

The new control points $\check{\mathbf{b}}_0, \dots, \check{\mathbf{b}}_2$ are obtained as follows:

$$f(\check{\mathbf{b}}_0, \dots, \check{\mathbf{b}}_2) = \sum_{i=0}^{3} \left\| \frac{i}{3}\check{\mathbf{b}}_{i-1} + \frac{3-i}{3}\check{\mathbf{b}}_i - \mathbf{b}_i \right\|^2 = \min$$

To receive the minimum we have to set the gradient of f to zero. The partial derivatives of f can be written as

$$\frac{\partial}{\partial \check{\mathbf{b}}_j} f(\check{\mathbf{b}}_0, \dots, \check{\mathbf{b}}_2) = \frac{\partial}{\partial \check{\mathbf{b}}_j} \left(\left\| \frac{j}{3}\check{\mathbf{b}}_{j-1} + \frac{3-j}{3}\check{\mathbf{b}}_j - \mathbf{b}_j \right\|^2 + \left\| \frac{j+1}{3}\check{\mathbf{b}}_j + \frac{3-(j+1)}{3}\check{\mathbf{b}}_{j+1} - \mathbf{b}_{j+1} \right\|^2 \right)$$

with $j = 0, \dots, 2$
Then, the gradient of f is:

$$\textbf{grad}\ f = \begin{pmatrix} \frac{\partial f}{\partial \check{\mathbf{b}}_0} \\ \frac{\partial f}{\partial \check{\mathbf{b}}_1} \\ \frac{\partial f}{\partial \check{\mathbf{b}}_2} \end{pmatrix} = \begin{pmatrix} \frac{20}{9}\check{\mathbf{b}}_0 + \frac{4}{9}\check{\mathbf{b}}_1 - 2\mathbf{b}_0 - \frac{2}{3}\mathbf{b}_1 \\ \frac{4}{9}\check{\mathbf{b}}_0 + \frac{16}{9}\check{\mathbf{b}}_1 + \frac{4}{9}\check{\mathbf{b}}_2 - \frac{4}{3}\mathbf{b}_1 - \frac{4}{3}\mathbf{b}_2 \\ \frac{4}{9}\check{\mathbf{b}}_1 + \frac{20}{9}\check{\mathbf{b}}_2 - \frac{2}{3}\mathbf{b}_2 - 2\mathbf{b}_3 \end{pmatrix} = 0$$

We receive two sets of linear equations: one for the x-coordinate and one for the y-coordinate:

$$\begin{pmatrix} 20/9 & 4/9 & 0 \\ 4/9 & 16/9 & 4/9 \\ 0 & 4/9 & 20/9 \end{pmatrix} \begin{pmatrix} \check{b}_0^x \\ \check{b}_1^x \\ \check{b}_2^x \end{pmatrix} = \begin{pmatrix} 0 \\ 160/3 \\ 320/3 \end{pmatrix}, \qquad \begin{pmatrix} 20/9 & 4/9 & 0 \\ 4/9 & 16/9 & 4/9 \\ 0 & 4/9 & 20/9 \end{pmatrix} \begin{pmatrix} \check{b}_0^y \\ \check{b}_1^y \\ \check{b}_2^y \end{pmatrix} = \begin{pmatrix} 40/3 \\ 160/3 \\ 40/3 \end{pmatrix}$$

with the solution:

$$\check{\mathbf{b}}_0 = \begin{pmatrix} -4 \\ 0 \end{pmatrix}, \quad \check{\mathbf{b}}_1 = \begin{pmatrix} 20 \\ 30 \end{pmatrix}, \quad \check{\mathbf{b}}_2 = \begin{pmatrix} 44 \\ 0 \end{pmatrix}$$

The Bézier curve with reduced degree is depicted in figure 3.12.

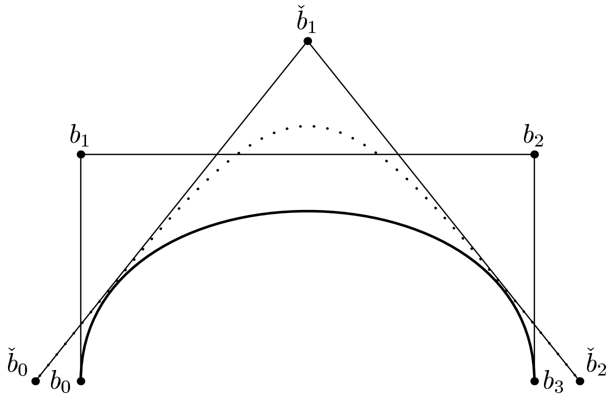


Figure 3.12: Degree reduction $n = 3 \to n = 2$

2nd Method of Solution:
Another method is based on the fact that degree elevation is a linear porcess, i. e. the control points of the degree-elevated curve can be obtained by an appropriate matrix multiplication on the initial control points. E. g., the degree elevation $n = 3 \to n = 4$ with the initial points $\mathbf{b}_0, \dots, \mathbf{b}_3$ and the control points $\mathbf{d}_0, \dots, \mathbf{d}_4$ of the degree elevated curve can be described by the following matrix:

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1/4 & 3/4 & 0 & 0 \\ 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & 3/4 & 1/4 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \Rightarrow \qquad \begin{pmatrix} \mathbf{d}_0 \\ \mathbf{d}_1 \\ \mathbf{d}_2 \\ \mathbf{d}_3 \\ \mathbf{d}_4 \end{pmatrix} = \mathbf{A} \cdot \begin{pmatrix} \mathbf{b}_0 \\ \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{pmatrix}$$

To find the *degree reduction* of a curve, one has to solve the linear system $\mathbf{A}\,\mathbf{d} = \mathbf{b}$. In general, this is not possible (over-determined linear system). Therefore find the best possible solution satisfying the condition $|\mathbf{A}\,\mathbf{d} - \mathbf{b}| = \min$.

Theorem 3.6 (Gauss Normal Equation)
The best approximation for the unsolvable linear system $\mathbf{A}\,\mathbf{d} = \mathbf{b}$ can be obtained by

$$\mathbf{A}^T\,\mathbf{A}\,\mathbf{d} = \mathbf{A}^T\,\mathbf{b} \tag{3.10}$$

Proof:

$$|\mathbf{A}\,\mathbf{d} - \mathbf{b}| = \min \;\Leftrightarrow\; |\mathbf{A}\,\mathbf{d} - \mathbf{b}|^2 = \min$$

$$|\mathbf{A}\,\mathbf{d} - \mathbf{b}|^2 = (\mathbf{A}\,\mathbf{d} - \mathbf{b})^T(\mathbf{A}\,\mathbf{d} - \mathbf{b}) = \mathbf{d}^T\mathbf{A}^T\mathbf{A}\mathbf{d} - 2\mathbf{d}^T\mathbf{A}^T\mathbf{b} + \mathbf{b}^T\mathbf{b}$$

To receive the minimum set the derivation to zero:

$$\frac{\partial}{\partial \mathbf{d}} \left(\mathbf{d}^T \mathbf{A}^T \mathbf{A} \mathbf{d} - 2\mathbf{d}^T \mathbf{A}^T \mathbf{b} + \mathbf{b}^T \mathbf{b} \right) = 2\mathbf{A}^T \mathbf{A} \mathbf{d} - 2\mathbf{A}^T \mathbf{b} = 0 \Leftrightarrow \mathbf{A}^T \mathbf{A} \mathbf{d} = \mathbf{A}^T \mathbf{b}$$

$\square$

Remark: Due to the structure of $\mathbf{A}$, the square matrix $\mathbf{A}^T\mathbf{A}$ is tri-diagonal. Therefore, the linear system can be solved very efficiently.

Example: $n = 3 \rightarrow n = 2$ (see figure 3.12)

Given: control points $\mathbf{b}_0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, $\mathbf{b}_1 = \begin{pmatrix} 0 \\ 20 \end{pmatrix}$, $\mathbf{b}_2 = \begin{pmatrix} 40 \\ 20 \end{pmatrix}$, $\mathbf{b}_3 = \begin{pmatrix} 40 \\ 0 \end{pmatrix}$

The following matrix $\mathbf{A}$ describes the degree elevation $n = 2 \rightarrow n = 3$:

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 \\ 1/3 & 2/3 & 0 \\ 0 & 2/3 & 1/3 \\ 0 & 0 & 1 \end{pmatrix}$$

To calculate the new control points use *Gauss Normal Equation*

$$\begin{aligned} \mathbf{d} &= \left(\mathbf{A}^T\mathbf{A}\right)^{-1} \mathbf{A}^T \mathbf{b} \\ \mathbf{d} &= \frac{1}{20} \begin{pmatrix} 19 & 3 & -3 & 1 \\ -5 & 15 & 15 & -5 \\ 1 & -3 & 3 & 19 \end{pmatrix} \cdot \mathbf{b} \end{aligned}$$

with the result

$$\mathbf{d}_0 = \begin{pmatrix} -4 \\ 0 \end{pmatrix}, \quad \mathbf{d}_1 = \begin{pmatrix} 20 \\ 30 \end{pmatrix}, \quad \mathbf{d}_2 = \begin{pmatrix} 44 \\ 0 \end{pmatrix}$$

3.3 Bézier Splines

3.3.1 Construction of “complex” Curves

To model more complex shapes with Bézier curves, there are two possibilities:

- Method 1: use a Bézier curve of *high degree*. This method may cause numerical problems.
- Method 2: split curve into *several simple parts* and represent every segment as a low degree Bézier curve.

Definition 3.3
A *Bézier spline* is a piecewise polynomial curve. The segments are represented by Bézier curves.

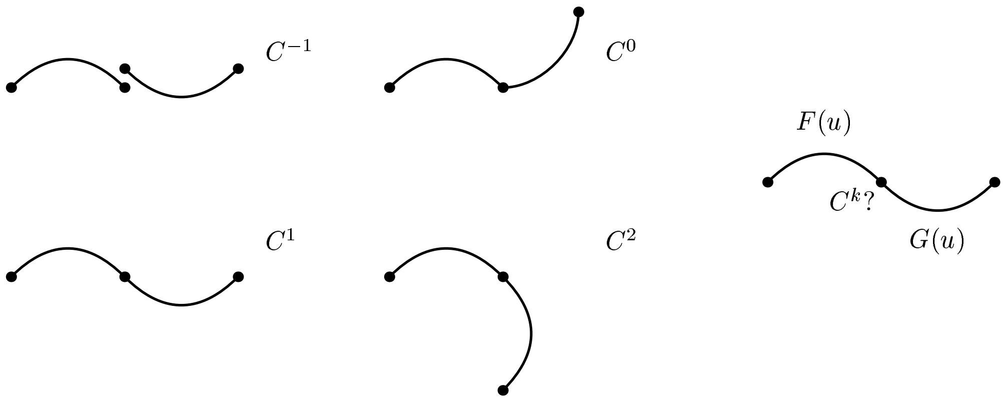


Figure 3.13: Joins of Bézier curves

Problem: continuity of the joins? Discontinuities or non-differentiable joins cause "non-smoothness" of the curve.

Definition 3.4
Given two polynomial curves $F : [r, s] \to \mathbb{R}^d$ and $G : [s, t] \to \mathbb{R}^d$. Then the curves are C^k-*continuous* at s if $F(s) = G(s),\ F'(s) = G'(s),\ F''(s) = G''(s), \dots, F^{(k)}(s) = G^{(k)}(s)$.

Interpretation:

- C^0-continuous = continuous, no jumps
- C^1-continuous = C^0-continuous + same tangent vector
- C^2-continuous = C^1-continuous + same osculating circle.

3.3.2 C^k-Continuity of two Bézier Curves

Theorem 3.7
Let $F(u) = \sum_{i=0}^{n} B_i^{\Delta_l,n}\, \mathbf{b}_i$ and $G(u) = \sum_{i=0}^{n} B_i^{\Delta_r,n}\, \mathbf{c}_i$ Bézier curves over $\Delta_l = [r, s]$ and $\Delta_r = [s, t]$, $r < s < t$. Then:

$$F, G \text{ are } C^k\text{-continuous at } s \;\Leftrightarrow\; \mathbf{c}_i = \mathbf{b}^i_{n-i}(t) \quad \forall\, 0 \le i \le k \tag{3.11}$$

Proof: Any Bézier curve has a unique representation over every interval. (In this case we only consider $F(u)$ over the interval Δ_l, but in fact the curve has the domain$(-\infty, +\infty)$.) Therefore we can define curve $F(u)$ over the interval Δ_r without changing it:

$$\begin{aligned}
F(u) &= \sum_{i=0}^{n} B_i^{\Delta_l,n}(u) f(\underbrace{r,\dots,r}_{n-i},\underbrace{s,\dots,s}_{i}) = \sum_{i=0}^{n} B_i^{\Delta_r,n}(u) f(\underbrace{s,\dots,s}_{n-i},\underbrace{t,\dots,t}_{i}) \\
F'(u) &= \frac{n}{t-s}\sum_{i=0}^{n-1} B_i^{\Delta_r,n-1}(u)(f(\underbrace{s,\dots,s}_{n-i-1},\underbrace{t,\dots,t}_{i+1}) - f(\underbrace{s,\dots,s}_{n-i},\underbrace{t,\dots,t}_{i})) \\
&\vdots \\
F^{(k)}(u) &= \frac{n!}{(t-s)^k (n-k)!}\sum_{i=0}^{n-k} B_i^{\Delta_r,n-k}(u) \sum_{j=0}^{k} \binom{k}{j} (-1)^{k-j} f(\underbrace{s,\dots,s}_{n-i-j},\underbrace{t,\dots,t}_{i+j})
\end{aligned}$$

$$F(s) = f(\underbrace{s,\ldots,s}_{n}) \qquad G(s) = f(\underbrace{s,\ldots,s}_{n})$$

$$F'(s) = \frac{n}{t-s}(f(\underbrace{s,\ldots,s}_{n-1},t) - f(\underbrace{s,\ldots,s}_{n})) \qquad G'(s) = \frac{n}{t-s}(g(\underbrace{s,\ldots,s}_{n-1},t) - g(\underbrace{s,\ldots,s}_{n}))$$

$$\vdots$$

$$F^{(k)}(s) = \frac{n!}{(t-s)^k(n-k)!}\sum_{j=0}^{k}\binom{k}{j}(-1)^{k-j}f(\underbrace{s,\ldots,s}_{n-j},\underbrace{t,\ldots,t}_{j})$$

$$G^{(k)}(s) = \frac{n!}{(t-s)^k(n-k)!}\sum_{j=0}^{k}\binom{k}{j}(-1)^{k-j}g(\underbrace{s,\ldots,s}_{n-j},\underbrace{t,\ldots,t}_{j})$$

$\Rightarrow F, G$ are C^k-continuous at s

$$\Leftrightarrow \quad f(\underbrace{s,\ldots,s}_{n-k},\underbrace{t,\ldots,t}_{k}) = g(\underbrace{s,\ldots,s}_{n-k},\underbrace{t,\ldots,t}_{k})$$

$$\Leftrightarrow \quad \mathbf{b}^i_{n-i}(t) = \mathbf{c}_i \qquad \text{for all} \quad 0 \le i \le k$$

$\square$

Remark: $t = \left(\frac{t-r}{s-r}\right)\cdot s + \left(1 - \frac{t-r}{s-r}\right)\cdot r$

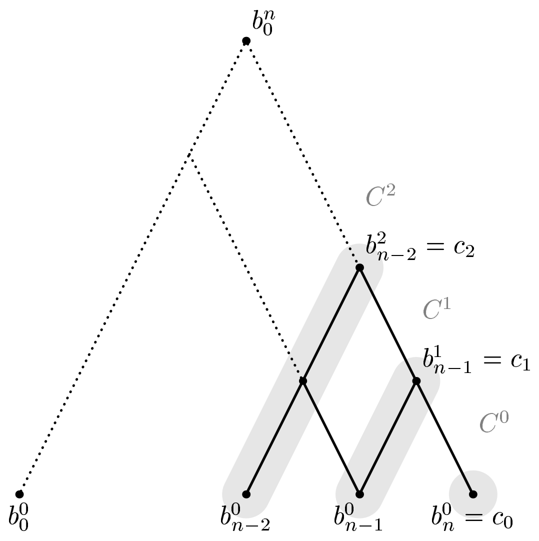


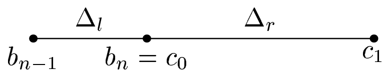
Figure 3.14: Join conditions in the de Casteljau algorithm

3.3.3 Geometrical Construction of continuous Joins

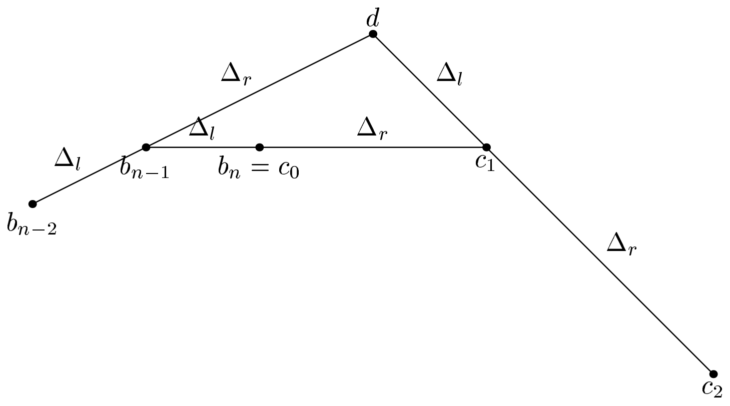
Let F, G and Δ_l, Δ_r like above and $\frac{\Delta_l}{\Delta_r} = \frac{s-r}{t-s}$. Then:

(1) F, G are C^0-continuous $\Leftrightarrow \mathbf{b}_n = \mathbf{c}_0$.

(2) F, G are C^1-continuous $\Leftrightarrow$



(3) F, G are C^2-continuous $\Leftrightarrow$ “A-frame” property:



Example: Given: standard parabola $F(u) = \begin{pmatrix} u \\ u^2 \end{pmatrix}$ over $[r, s] = [-1, 1]$

Find: continuation $G(u) =?$ over $[s, t] = [1, 3]$

C^0-continuity: $\mathbf{c}_0 = \mathbf{b}_2$

C^1-continuity: $t = \frac{t-r}{s-r} \cdot s + \left(1 - \frac{t-r}{s-r}\right) \cdot r = 2 \cdot s - 1 \cdot r$.

Result: see figure 3.16

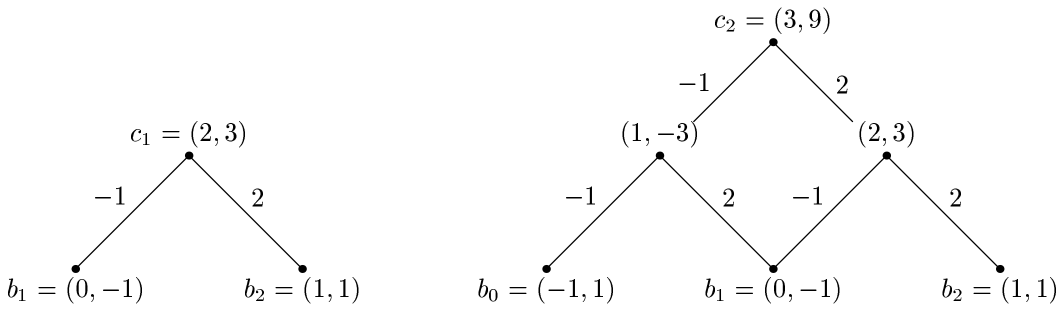


Figure 3.15: de Casteljau conditions for C^1- and C^2-continuity

Remark: The figure 3.17 is fully described by the location of $\mathbf{d}_0, \mathbf{d}_1$, the endpoints and the intervals Δ_i.
Note: The points $\mathbf{d}_j$ are B-Spline control points (see chapter 4).

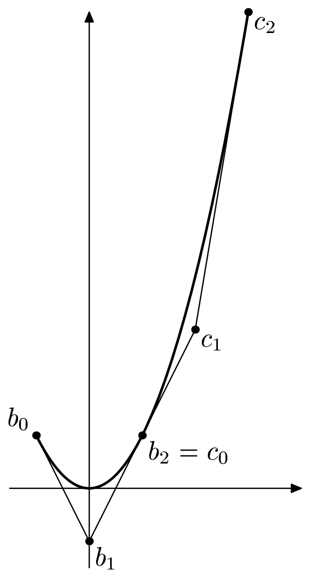


Figure 3.16: C^2-continuation of the standard parabola

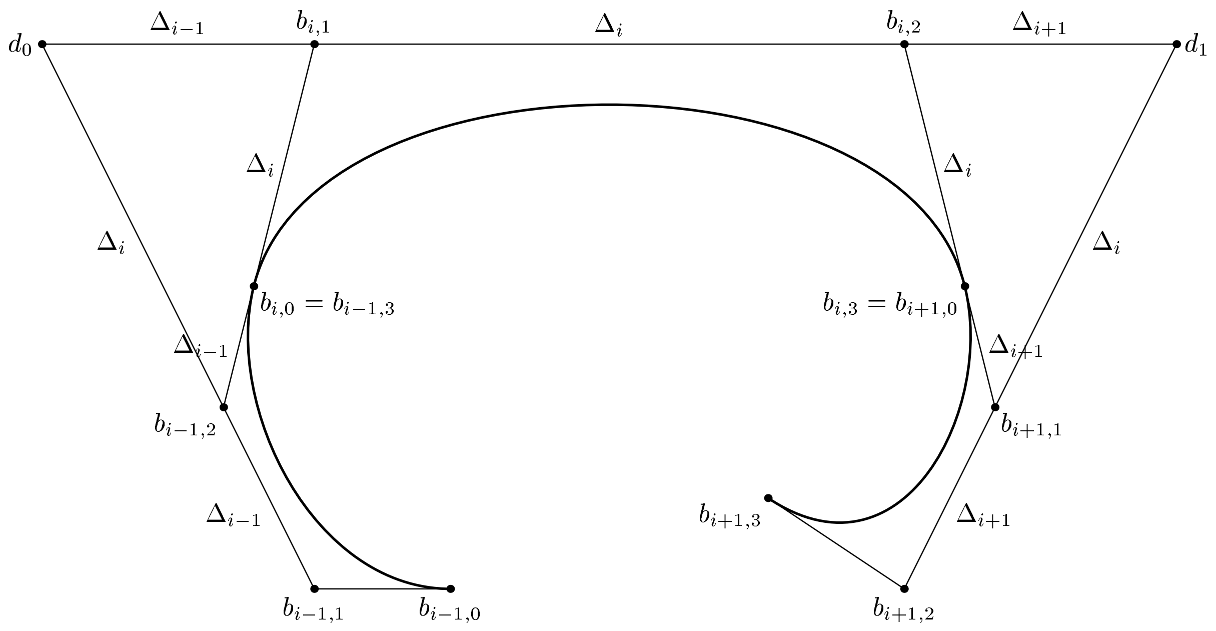


Figure 3.17: C^2-join of several cubic Bézier segments

3.3.4 Interpolation with Bézier Splines

Given: Points $\mathbf{p}_0, \dots, \mathbf{p}_k$ and parameter values $t_0, \dots, t_k$.

Find: A curve that *interpolates* the points $\mathbf{p}_i$ at the corresponding parameter values, i. e. all $\mathbf{p}_i$ must be on this curve.

A simple idea are *Catmull-Rom Splines* [2]:

- find a (e. g. cubic) Bézier curve between two neighbour points $\mathbf{p}_i$, $\mathbf{p}_{i+1}$
- specify the tangents (= derivatives) at the end points as follows:

$$\begin{aligned} F'(t_i) &= \tfrac{1}{2}\,(\mathbf{p}_{i+1} - \mathbf{p}_{i-1}) \\ F'(t_{i+1}) &= \tfrac{1}{2}\,(\mathbf{p}_{i+2} - \mathbf{p}_i) \qquad (3.12) \\ F(t_i) &= \mathbf{p}_i \\ F(t_{i+1}) &= \mathbf{p}_{i+1} \end{aligned}$$

(3.13)

Theorem 3.8
If $u \in \Delta = [t_i, t_{i+1}]$, you get the control points of an interpolating Bézier curve as follows:

$$\begin{aligned} \mathbf{b}_0^i &= \mathbf{p}_i \\ \mathbf{b}_1^i &= \mathbf{p}_i + \frac{t_{i+1} - t_i}{6}\,(\mathbf{p}_{i+1} - \mathbf{p}_{i-1}) \qquad (3.14) \\ \mathbf{b}_2^i &= \mathbf{p}_{i+1} - \frac{t_{i+1} - t_i}{6}\,(\mathbf{p}_{i+2} - \mathbf{p}_i) \\ \mathbf{b}_3^i &= \mathbf{p}_{i+1} \end{aligned}$$

(3.15)

Sketch of Proof:

$$F'(t_i) = \frac{3}{t_{i+1} - t_i} \sum_{j=0}^{2} B_j^{\Delta_i,2}(t_i)\,(\mathbf{b}_{j+1}^i - \mathbf{b}_j^i) = \frac{3}{t_{i+1} - t_i}\,(\mathbf{b}_1^i - \mathbf{b}_0^i) \overset{!}{=} \frac{1}{2}\,(\mathbf{p}_{i+1} - \mathbf{p}_{i-1})$$

$$F'(t_{i+1}) = \frac{3}{t_{i+1} - t_i} \sum_{j=0}^{2} B_j^{\Delta_i,2}(t_{i+1})\,(\mathbf{b}_{j+1}^i - \mathbf{b}_j^i) = \frac{3}{t_{i+1} - t_i}\,(\mathbf{b}_3^i - \mathbf{b}_2^i) \overset{!}{=} \frac{1}{2}\,(\mathbf{p}_{i+2} - \mathbf{p}_i)$$

□

Remark:

- Problem: What happens at the end points?
  - 1st case: closed curve (i. e. $\mathbf{p}_0 = \mathbf{p}_k$).
  - 2nd case: open curve ($\mathbf{p}_0 \neq \mathbf{p}_k$). In this case the natural end conditions $F''(t_0) = 0$ and $F''(t_k) = 0$ hold true.

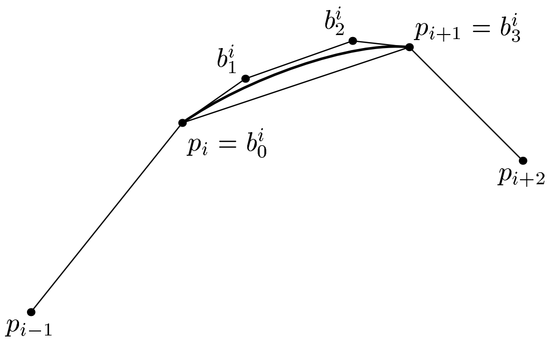
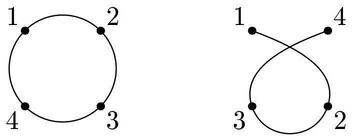


Figure 3.18: Interpolation with Catmull-Rom Splines



- When interpolating points, you have to consider the order (or *topology* = neighbourhood relation) of the points.
- Only if the points are "uniformly distributed", you can assume that all parameter intervals have the length 1 (e. g. $t_i = i$). In general, this is not true.

[2]E. Catmull, R. Rom. A Class of interpolating Splines, 1974

4 B-splines

4.1 Basics

B-spline curves are piecewise polynomial curves with built-in C^k-continuity at the joins. Mainly we deal with curves of degree n that are C^{n-1}-continuous, e. g. a B-spline curve of degree 1 is piecewise linear and C^0-continuous.

Given: degree n and a sequence of knots $t_0 < t_1 < t_2 < \cdots < t_{m+n}$ where m is the number of control points of the curve. Thus, the *knot vector* $T = (t_i)_{i \in \mathbb{Z}}$ defines parameter intervals $[t_i, t_{i+1})$.

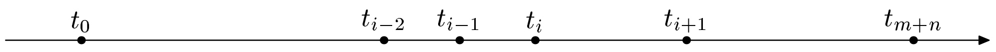


Figure 4.1: A knot vector

The B-spline basis functions $N_i^k(u)$ are defined recursively by degree k:

$$N_i^0(u) = \begin{cases} 1 & t_i \leq u < t_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$N_i^1(u) = \frac{u - t_i}{t_{i+1} - t_i} N_i^0(u) + \frac{t_{i+2} - u}{t_{i+2} - t_{i+1}} N_{i+1}^0(u)$$

General recursion formula:

Definition 4.1 (Normalized B-spline Basis Functions)

$$N_i^0(u) := \begin{cases} 1 & t_i \leq u < t_{i+1} \\ 0 & \text{otherwise} \end{cases}, \quad i = 0, \ldots, n+m-1 \tag{4.1}$$

$$N_i^k(u) := \frac{u - t_i}{t_{i+k} - t_i} N_i^{k-1}(u) + \frac{t_{i+k+1} - u}{t_{i+k+1} - t_{i+1}} N_{i+1}^{k-1}(u), \quad k = 1, \ldots, n \tag{4.2}$$

The N_i^k are computed according to the follwing scheme:

$$\begin{array}{ccccccccccccccc}
N_0^0 & & N_1^0 & & N_2^0 & & N_3^0 & & \cdots & & N_{n+m-3}^0 & & N_{n+m-2}^0 & & N_{n+m-1}^0 \\
\downarrow & \swarrow & \downarrow & \swarrow & \downarrow & \swarrow & \downarrow & \swarrow & & \swarrow & \downarrow & \swarrow & \downarrow & \swarrow & \\
N_0^1 & & N_1^1 & & N_2^1 & & N_3^1 & & \cdots & & N_{n+m-3}^1 & & N_{n+m-2}^1 & & \\
\downarrow & \swarrow & \downarrow & \swarrow & \downarrow & \swarrow & \downarrow & \swarrow & & \swarrow & \downarrow & \swarrow & & & \\
\vdots & & \vdots & & \vdots & & \vdots & & & & \vdots & & & & \\
\downarrow & \swarrow & \downarrow & \swarrow & \downarrow & \swarrow & \downarrow & \swarrow & & \swarrow & \downarrow & & & & \\
N_0^n & & N_1^n & & N_2^n & & N_3^n & & \cdots & & N_{m-1}^n & & & &
\end{array}$$

Example: We choose the knot vector $T = (\ldots, 0, 1, 2, 3, 4, 5, 6, \ldots)$, i. e. $t_i = i$.

$$k = 0: \qquad N_3^0(u) = \begin{cases} 0 & u < 3 \\ 1 & 3 \le u < 4 \\ 0 & 4 \le u \end{cases}$$

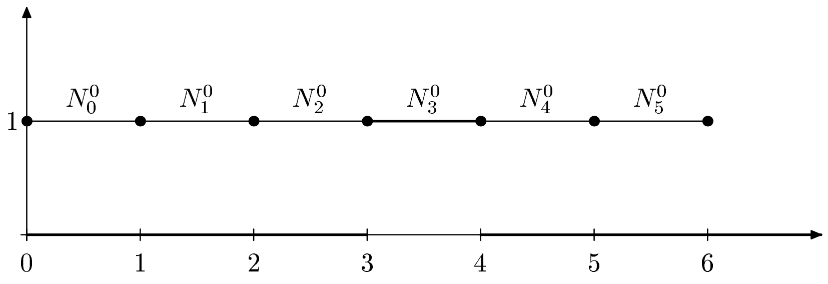


Figure 4.2: B-splines for $k = 0$

$$k = 1: \qquad N_3^1(u) = \begin{cases} 0 & u < 3 \\ u - 3 & 3 \le u < 4 \\ 5 - u & 4 \le u < 5 \\ 0 & 5 \le u \end{cases}$$

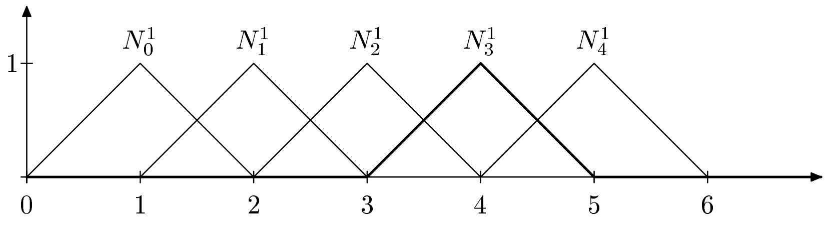


Figure 4.3: B-splines for $k = 1$

$$k = 2: \qquad N_3^2(u) = \frac{u-3}{5-3}\, N_3^1(u) + \frac{6-u}{6-4}\, N_4^1(u) =$$

$$= \begin{cases} 0 & u < 3 \\ \frac{1}{2}\,(u-3)^2 & 3 \le u < 4 \\ \frac{1}{2}\,[(u-3)\,(5-u) + \,(6-u)\,(u-4)] & 4 \le u < 5 \\ \frac{1}{2}\,(6-u)^2 & 5 \le u < 6 \\ 0 & 6 \le u \end{cases}$$

Remark: The B-spline $N_3^2(u)$ is tangent-continuous at the joins.

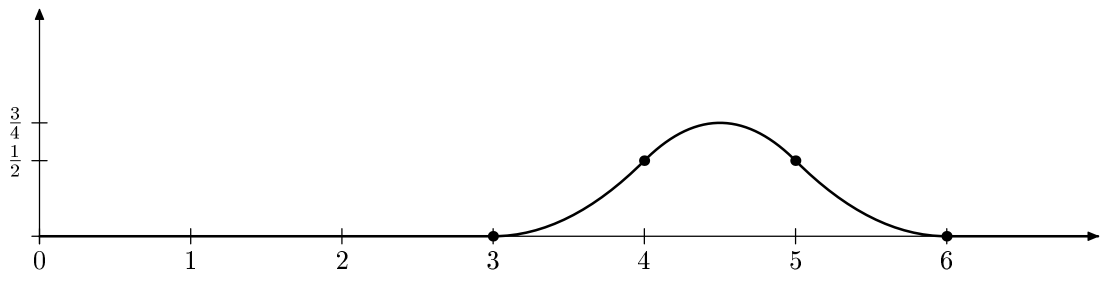


Figure 4.4: The B-spline $N_3^2(u)$

Properties of B-splines

(1) N_i^k is piecewise polynomial of degree k.

(2) The support of N_i^k: $\operatorname{supp}(N_i^k) := \overline{\{u \mid N_i^k(u) \neq 0\}} = [t_i, t_{i+k+1}]$, i. e. the support consists of $k+1$ intervals per section.

(3) Positivity: $N_i^k(u) > 0$ for $u \in (t_i, t_{i+k+1})$.

(4) Partition of 1: $\sum_{i=0}^{m-1} N_i^n(u) = 1$ for $u \in [t_n, t_m)$.

(5) The derivative of a basis function is given by

$$(N_i^k)'(u) = \frac{k}{t_{i+k} - t_i} N_i^{k-1}(u) - \frac{k}{t_{i+k+1} - t_{i+1}} N_{i+1}^{k-1}(u) \tag{4.3}$$

Proof by induction on k:
initial condition: $k = 1$
$N_i^{k-1}(u)$ and $N_{i+1}^{k-1}(u)$ are either 0 or 1 (see fig. 4.2), and thus $(N_i^1)'(u)$ is either

$$\frac{1}{t_{i+1} - t_i} \quad \text{or} \quad -\frac{1}{t_{i+2} - t_{i+1}}$$

(see fig. 4.3).
induction step: Inductional assumption: Equation 4.3 is true for $k > 1$.
Use the product rule to differentiate the basis function:

$$\begin{aligned}(N_i^k)'(u) = {} & \frac{1}{t_{i+k} - t_i} N_i^{k-1}(u) + \frac{u - t_i}{t_{i+k} - t_i} (N_i^{k-1})'(u) - \frac{1}{t_{i+k+1} - t_{i+1}} N_{i+1}^{k-1}(u) + \\ & + \frac{t_{i+1} - u}{t_{i+k+1} - t_{i+1}} (N_{i+1}^{k-1})'(u)\end{aligned}$$

Substitute equation 4.3:

$$\begin{aligned}(N_i^k)'(u) = {} & \frac{1}{t_{i+k} - t_i} N_i^{k-1}(u) - \frac{1}{t_{i+k+1} - t_{i+1}} N_{i+1}^{k-1}(u) + \frac{k-1}{t_{i+k} - t_i} \frac{u - t_i}{t_{i+k-1} - t_i} N_i^{k-2}(u) + \\ & + \frac{k-1}{t_{i+k} - t_{i+1}} \left(\frac{t_{i+k+1} - u}{t_{i+k+1} - t_{i+1}} - \frac{u - t_i}{t_{i+k} - t_i} \right) N_{i+1}^{k-2}(u) - \\ & - \frac{k-1}{t_{i+k+1} - t_{i+1}} \frac{t_{i+k+1} - u}{t_{i+k+1} - t_{i+2}} N_{i+2}^{k-2}(u)\end{aligned}$$

Because of $\frac{t_{i+k+1}-u}{t_{i+k+1}-t_{i+1}} - \frac{u-t_i}{t_{i+k}-t_i} = \frac{t_{i+k}-u}{t_{i+k}-t_i} - \frac{u-t_{i+1}}{t_{i+k+1}-t_{i+1}}$, we obtain:

$$\begin{aligned}
(N_i^k)'(u) &= \frac{1}{t_{i+k}-t_i} N_i^{k-1}(u) - \frac{1}{t_{i+k+1}-t_{i+1}} N_{i+1}^{k-1}(u) + \\
&\quad + \frac{k-1}{t_{i+k}-t_i} \underbrace{\left(\frac{u-t_i}{t_{i+k-1}-t_i} N_i^{k-2}(u) + \frac{t_{i+k}-u}{t_{i+k}-t_{i+1}} N_{i+1}^{k-2}(u) \right)}_{N_i^{k-1}(u)} - \\
&\quad - \frac{k-1}{t_{i+k+1}-t_{i+1}} \underbrace{\left(\frac{u-t_{i+1}}{t_{i+k}-t_{i+1}} N_{i+1}^{k-2}(u) + \frac{t_{i+k+1}-u}{t_{i+k+1}-t_{i+2}} N_{i+2}^{k-2}(u) \right)}_{N_{i+1}^{k-1}(u)} = \\
&= \frac{k}{t_{i+k}-t_i} N_i^{k-1}(u) - \frac{k}{t_{i+k+1}-t_{i+1}} N_{i+1}^{k-1}(u) \qquad \square
\end{aligned}$$

Example: Support for cubic B-splines:

$$\begin{tikzcd}
N_0^0[t_0,t_1] \arrow[d] & N_1^0[t_1,t_2] \arrow[d] \arrow[dl] & N_2^0[t_2,t_3] \arrow[d] \arrow[dl] & N_3^0[t_3,t_4] \arrow[d] \arrow[dl] & N_4^0[t_4,t_5] \ \cdots \arrow[dl] \\
N_0^1[t_0,t_2] \arrow[d] & N_1^1[t_1,t_3] \arrow[d] \arrow[dl] & N_2^1[t_2,t_4] \arrow[d] \arrow[dl] & N_3^1[t_3,t_5] \ \cdots \arrow[dl] & \vdots \\
N_0^2[t_0,t_3] \arrow[d] & N_1^2[t_1,t_4] \arrow[d] \arrow[dl] & N_2^2[t_2,t_5] \ \cdots \arrow[dl] & \vdots & \\
N_0^3[t_0,t_4] & N_1^3[t_1,t_5] \ \cdots & \vdots & &
\end{tikzcd}$$

4.2 B-spline Curves

Definition 4.2
Given:

– degree n

– knot vector $t_0 < t_1 < \cdots < t_{m+n}$

– control points $\mathbf{d}_0, \ldots, \mathbf{d}_{m-1} \in \mathbb{R}^2$ or $\mathbb{R}^3$.

$$F(u) := \sum_{i=0}^{m-1} N_i^n(u)\, \mathbf{d}_i \tag{4.4}$$

is called *B-spline curve*. The points $\mathbf{d}_i$ are called *control points* or *de Boor points*[1]. They form a control polygon.

[1]Carl de Boor, 1972

4.2.1 Shape Properties of B-spline Curves

Theorem 4.1 (Shape properties of B-spline curves)
Let $F(u) = \sum_{i=0}^{m-1} N_i^n(u)\,\mathbf{d}_i$ a B-spline curve of degree n over the knot vector T. Then the following properties hold true:

(1) In general there is *no* endpoint interpolation.

(2) For $t_i \le u < t_{i+1}$, $F(u)$ lies in the convex hull of the $n+1$ control points $\mathbf{d}_{i-n}, \dots, \mathbf{d}_i$ ("local convex hull").

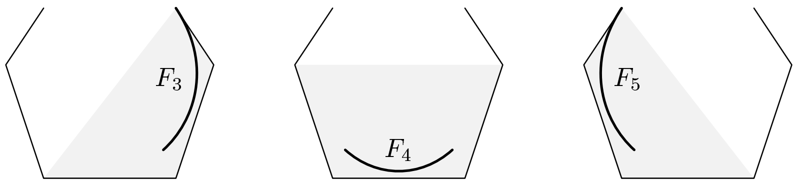


Figure 4.5: Local convex hull

(3) Local control: For $u \in [t_i, t_{i+1})$ the curve is independent of $\mathbf{d}_j$ for $j < i - n$ and $j > i$.

(4) If n control points coincide, then the curve goes through this point and is tangent to the control polygon (Fig. 4.6).

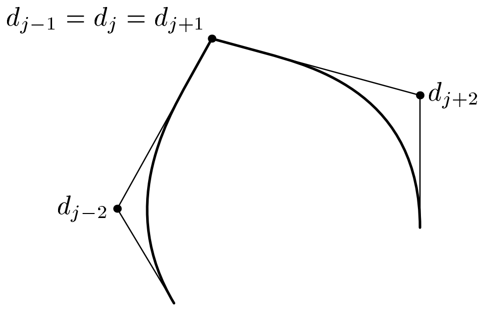


Figure 4.6: $n = 3$ control points coincide

(5) If n control points are on a line, then the curve touches this line (Fig. 4.7).

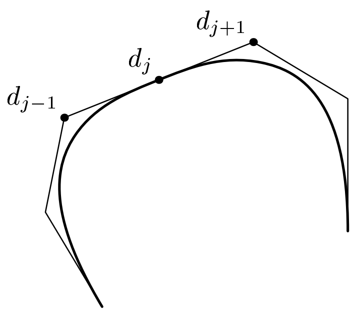


Figure 4.7: $n = 3$ control points on a line

(6) If $n+1$ control points are on a line L, then $F(u) \in L$ for $t_i \leq u < t_{i+1}$, i. e. a whole segment of the curve $F(u)$ coincides with L (Fig. 4.8).

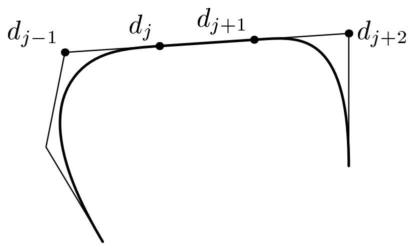


Figure 4.8: $n+1=4$ control points on a line

(7) Derivative:

$$F'(u) = \sum_{i=0}^{m-2} N_{i+1}^{n-1}(u)\,\mathbf{d}_i^* \quad \text{mit} \quad \mathbf{d}_i^* = \frac{n}{t_{i+n+1} - t_{i+1}}\,(\mathbf{d}_{i+1} - \mathbf{d}_i) \tag{4.5}$$

(8) If n knots $t_i = t_{i+1} = \cdots = t_{i+n-1}$ coincide, then $F(t_i) = \mathbf{d}_i$, i. e. the curve goes through a control point and is tangent to the control polygon.

(9) If φ is an affine map, then:

$$\varphi\left(\sum_{i=0}^{m-1} N_i^n(u)\,\mathbf{d}_i\right) = \sum_{i=0}^{m-1} N_i^n(u)\,\varphi(\mathbf{d}_i)$$

(affine invariance).

(10) Variation diminishing property: If H is a hyper plane in $\mathbb{R}^d$, then

$$\#(F \cap H) \leq \#(H \cap \text{control polygon})$$

4.2.2 Comparison of B-Spline and Bézier Curves

Bézier Curves

+ control polygon gives an impression of the possible run of the curve

+ the run of the Bézier curve can be changed by the translation of control points

but:

– control points have global influence on the run of the Bézier curve

– number of control points depends on the degree of the Bézier curve $\Rightarrow$ more control points, i. e. more possibilities to influence the run of the curve, only possible by degree elevation

B-Spline Curves

B-Spline curves do not have the disadvantages of Bézier curves.
Geometric properties of Bézier curves like "convex hull" or "variation diminishing" are also valid for B-Spline curves.
Additional advantages are:

+ B-Spline functions are defined locally
+ Translation of a control point has only local influence on the run of the curve
+ Insertion of additional control points without degree elevation
+ Smoothness of the junction of adjoining segments can be influenced "easily"

	Basis Functions	**Curves**
polynomial	Bernstein polynomials $B_i^{\Delta,n}$	Bézier curves $F(u) = \sum_{i=0}^{n} B_i^{\Delta,n}(u)\,\mathbf{b}_i$
piecewise polynomial	(normalized) B-splines N_i^n over knots T	B-spline curves $G(u) = \sum_{i=0}^{m-1} N_i^n(u)\,\mathbf{d}_i$

Table 4.1: Review: curve schemes mentioned so far

4.3 Control Points and Polar Forms

Theorem 4.2 (Control points and polar forms)
Given: a B-spline curve $F(u) = \sum_{i=0}^{m-1} N_i^n(u)\,\mathbf{d}_i$ of degree n over a knot vector $t_0 < t_1 < \cdots < t_{m+n}$. F restricted to the interval $[t_j, t_{j+1}]$ is a polynomial curve and has a unique polar form f_j. Then:

$$\mathbf{d}_l = f_j(t_{l+1}, \ldots, t_{l+n}) \quad \text{for } j-n \leq l \leq j \tag{4.6}$$

e. g. for $n = 3$ this means:

$$\mathbf{d}_{j-3} = f_j(t_{j-2}, t_{j-1}, t_j) \qquad \mathbf{d}_{j-1} = f_j(t_j, t_{j+1}, t_{j+2})$$
$$\mathbf{d}_{j-2} = f_j(t_{j-1}, t_j, t_{j+1}) \qquad \mathbf{d}_j = f_j(t_{j+1}, t_{j+2}, t_{j+3})$$

Special case: The Bézier representation of the curve segment $f_j(u)$ over the interval $[t_j, t_{j+1})$ can be obtained as usual by inserting the interval borders t_j und t_{j+1} into the polar form:

$$\mathbf{b}_{j,0} = f_j(t_j, t_j, t_j) \qquad \mathbf{b}_{j,2} = f_j(t_j, t_{j+1}, t_{j+1})$$
$$\mathbf{b}_{j,1} = f_j(t_j, t_j, t_{j+1}) \qquad \mathbf{b}_{j,3} = f_j(t_{j+1}, t_{j+1}, t_{j+1})$$

An application of this property is the conversion of a B-spline curve into a Bézier spline curve.

Example: $n = 3,\ T = (\dots, -6, \dots, -1, 0, 1, 2, 3, \dots)$. Consider the curve segment where $u \in [0,1]$. In this segment, f_0 is the polar form of the B-spline curve F. Control points:

$$\begin{aligned} \mathbf{d}_{-3} &= f_0(-2,-1,0) & \mathbf{d}_{-1} &= f_0(0,1,2) \\ \mathbf{d}_{-2} &= f_0(-1,0,1) & \mathbf{d}_{0} &= f_0(1,2,3) \end{aligned}$$

Bézier points of $F\,|_{[0,1]}$:

$$\begin{aligned} \mathbf{b}_{0} &= f_0(0,0,0) & \mathbf{b}_{2} &= f_0(0,1,1) \\ \mathbf{b}_{1} &= f_0(0,0,1) & \mathbf{b}_{3} &= f_0(1,1,1) \end{aligned}$$

Given: $\mathbf{d}_{-3}, \mathbf{d}_{-2}, \mathbf{d}_{-1}, \mathbf{d}_0$

Find: $\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$

The Bézier points are obtained by construction of suitable affine combinations, e. g.

$$\begin{aligned} f_0(0,0,0) &= \frac{1}{2} f_0(0,0,-1) + \frac{1}{2} f_0(0,0,1) = \\ &= \frac{1}{2}\left[\frac{2}{3} f_0(0,1,-1) + \frac{1}{3} f_0(0,-2,-1)\right] + \frac{1}{2}\left[\frac{2}{3} f_0(0,-1,1) + \frac{1}{3} f_0(0,2,1)\right] = \\ &= \frac{1}{2}\left[\frac{2}{3}\mathbf{d}_{-2} + \frac{1}{3}\mathbf{d}_{-3}\right] + \frac{1}{2}\left[\frac{2}{3}\mathbf{d}_{-2} + \frac{1}{3}\mathbf{d}_{-1}\right] = \\ &= \frac{1}{6}\mathbf{d}_{-3} + \frac{2}{3}\mathbf{d}_{-2} + \frac{1}{6}\mathbf{d}_{-1} = \mathbf{b}_0 \end{aligned}$$

Similar computation for the other control points.

4.4 Multiple Knots

Multiple Knots are allowed, if the multiplicity of the knots is less or equal to the degree. Generalized knot vector:

$$t_0 \le t_1 \le \dots \le t_j \le t_{j+1} \le \dots \le t_{m+n}\,, \qquad t_j \lneq t_{j+n} \tag{4.7}$$

Exception: The first and last knot may have the multiplicity $n+1$.

We have to extend the definition of the B-spline basis functions to the case of multiplicities:

$$N_i^0(u) = \begin{cases} 1 & u \in [t_i, t_{i+1}) \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

The recursion formula reduces B-splines of degree k to B-splines of degree $k-1$:

$$N_i^k(u) = \frac{u - t_i}{t_{i+k} - t_i} N_i^{k-1}(u) + \frac{t_{i+k+1} - u}{t_{i+k+1} - t_{i+1}} N_{i+1}^{k-1}(u)$$

(1) case $t_{i+k} - t_i = 0$, i. e. the multiplicity of t_i ist greater or equal to $k + 1$.

$$N_i^k(u) = \frac{t_{i+k+1} - u}{t_{i+k+1} - t_{i+1}} N_{i+1}^{k-1}(u) \tag{4.9}$$

(2) case $t_{i+k+1} - t_{i+1} = 0$, i. e. the multiplicity of t_{i+1} is greater or equal to $k + 1$.

$$N_i^k(u) = \frac{u - t_i}{t_{i+k} - t_i} N_i^{k-1}(u) \tag{4.10}$$

(3) case $t_{i+k+1} - t_i = 0$, i. e. the multiplicity of t_i is greater or equal to $k + 2$.

$$N_i^k(u) \equiv 0 \tag{4.11}$$

Example: $t_0 < t_1 < t_2 = t_3 < t_4 < t_5$

$$\begin{aligned} N_0^1(u) &= \frac{u - t_0}{t_1 - t_0} N_0^0(u) + \frac{t_2 - u}{t_2 - t_1} N_1^0(u) \\ N_1^1(u) &= \frac{u - t_1}{t_2 - t_1} N_1^0(u) \\ N_2^1(u) &= \frac{t_4 - u}{t_4 - t_3} N_3^0(u) \\ N_3^1(u) &= \frac{u - t_3}{t_4 - t_3} N_3^0(u) + \frac{t_5 - u}{t_5 - t_4} N_4^0(u) \end{aligned}$$

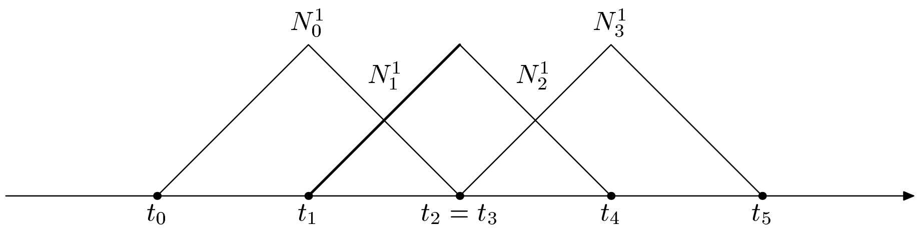


Figure 4.9: B-splines over a knot vector with multiplicities (emphasized: N_1^1)

Using a knot vector with multiple knots, some B-spline basis functions can vanish:
Example: Quadratic B-spline basis functions over $T = (\dots, t_0, t_1, t_2, t_3, t_4, t_5, \dots) = (\dots, -1, 0, 1, 1, 2, 3, \dots) \Rightarrow N_2^0$ does not exist!

Remark: The higher the multiplicity of the knots the smaller is the continuity (see Theorem 4.3).

Theorem 4.3 (Curry-Schoenberg)
Given: a sequence of knots $t_0 \leq t_1 \leq \cdots \leq t_{m+n}$. These knots have multiplicities $\mu_0, \dots, \mu_{m+n}$, where the inner knots have the multiplicity $1 \leq \mu_i \leq n$ and the border knots

have the multiplicity $\mu_0 = \mu_{m+n} = n+1$. Now consider this knot vector as a sequence of single knots $\tau_0 < \tau_1 < \cdots < \tau_l$ with the corresponding multiplicities $\mu_0, \ldots, \mu_l$.

The normalized B-splines of degree n with respect to this knot vector make a basis of the following space: $\{F : [t_n, t_m] \to \mathbb{R}^d \ : \ F\,|_{[t_j,t_{j+1})}$ is a polynomial of degree n and f is $C^{n-\mu_i}$-continuous at the value $\tau_i, \forall\, n \leq j \leq m-1\}$.

Remark:

- The most important special case are the *proper splines*, where $\mu_1 = \cdots = \mu_{l-1} = 1$.
- If we insert a new knot $\bar{t}$, the dimension of the space increases by 1, i. e. the space of all piecewise polynomial functions over the old knot vector is a linear subspace of that over the new knot vector.

Multiple Knots and Endpoint Interpolation

The curves are to be at least C^0-continious. So from Theorem 4.3 follows, that the maximal multiplicity of the knots may be n. At the end points we can equate $t_0 \ldots t_n$, because the curve does not depend on t_0 (see scheme on page 34: For $u > t_3$, $N_0^2[t_0, t_3]$ does not contribute to the computation of $N_0^3[t_0, t_4]$, but $N_1^2[t_1, t_4]$ does). Therefore the maximum multiplicity allowed at the endpoints is $n+1$.

Theorem 4.4 (Endpoint interpolation)
If an end knot has multiplicity $n+1$, then the B-spline curve interpolates the corresponding control point and is tangent to the control polygon.

Sketch of Proof: Consider a B-spline curve of nth degree over the knot vector

$$\underbrace{a = \cdots = a}_{n+1} < b < \underbrace{c = \cdots = c}_{n+1}$$

For the basis function $N_0^n(u)$ for $u \in [a, b]$ we obtain:

$$N_0^n(u) = \frac{b-u}{b-a} N_1^{n-1}(u) = \left(\frac{b-u}{b-a}\right)^2 N_2^{n-2}(u) = \cdots = \left(\frac{b-u}{b-a}\right)^n.$$

This is the first Bernstein polynomial of nth degree with respect to the interval $[a, b]$.
$\Rightarrow N_0^n(u) = B_0^{\Delta,n}(u)$ with $\Delta = [a, b]$.
Similar, it can be shown that $N_n^n(u) = B_n^{\Delta,n}(u)$ with $\Delta = [b, c]$.
Because of $B_0^{\Delta,n}(a) = 1$ and $B_n^{\Delta,n}(c) = 1$ the end points are interpolated.
Tangential property:

$$\begin{aligned} F'(a) &= \sum_{i=0}^{m-2} N_{i+1}^{n-1}(a) \cdot \frac{n}{t_{i+n+1} - t_{i+1}}(d_{i+1} - d_i) = \\ &= N_1^{n-1}(a) \cdot \frac{n}{b-a}(d_1 - d_0) = \frac{n}{b-a}(d_1 - d_0), \end{aligned}$$

since $N_0^{n-1}(u)$ does not exist and $N_1^{n-1}(a) = 1$. All the other basis functions must be zero at the parameter value a (affine combination). So the line $d_0 d_1$ is the tangent to the Bézier curve at control point d_0. Similar for $F'(c)$. $\square$

Example: $n = 3,\ m = 6,\ T = (0,0,0,0,1,2,3,3,3,3)$.

We construct a piecewise polynomial curve $F(u)$ over this knot vector that "lives" on three not vanishing parameter intervals:

$$F(u) = \begin{cases} F_3(u) & 0 \le u < 1 \quad \text{control points: } \mathbf{d}_0\ \mathbf{d}_1\ \mathbf{d}_2\ \mathbf{d}_3 \\ F_4(u) & 1 \le u < 2 \quad \text{control points: } \mathbf{d}_1\ \mathbf{d}_2\ \mathbf{d}_3\ \mathbf{d}_4 \\ F_5(u) & 2 \le u < 3 \quad \text{control points: } \mathbf{d}_2\ \mathbf{d}_3\ \mathbf{d}_4\ \mathbf{d}_5 \end{cases}$$

Consider the interval $[t_4, t_5) = [1, 2)$. B-spline control points that affect F_4 are:

$$\mathbf{d}_1 = f_4(0,0,1) \qquad \mathbf{d}_3 = f_4(1,2,3)$$
$$\mathbf{d}_2 = f_4(0,1,2) \qquad \mathbf{d}_4 = f_4(2,3,3)$$

The Bézier representation of F_4 is obtained as follows:

$$\begin{aligned}
\mathbf{b}_{4,0} &= f_4(1,1,1) = \frac{1}{2} f_4(0,1,1) + \frac{1}{2} f_4(1,1,2) = \\
&= \frac{1}{2}\left(\frac{1}{2} f_4(0,0,1) + \frac{1}{2} f_4(0,1,2)\right) + \frac{1}{2} f_4(1,1,2) = \frac{1}{2}\left(\frac{1}{2}\mathbf{d}_1 + \frac{1}{2}\mathbf{d}_2\right) + \frac{1}{2}\mathbf{b}_{4,1} \\
\mathbf{b}_{4,1} &= f_4(1,1,2) = \frac{2}{3} f_4(0,1,2) + \frac{1}{3} f_4(1,2,3) = \frac{2}{3}\mathbf{d}_2 + \frac{1}{3}\mathbf{d}_3 \\
\mathbf{b}_{4,2} &= f_4(1,2,2) = \frac{1}{3} f_4(0,1,2) + \frac{2}{3} f_4(1,2,3) = \frac{1}{3}\mathbf{d}_2 + \frac{2}{3}\mathbf{d}_3 \\
\mathbf{b}_{4,3} &= f_4(2,2,2) = \frac{1}{2} f_4(1,2,2) + \frac{1}{2} f_4(2,2,3) = \\
&= \frac{1}{2} f_4(1,2,2) + \frac{1}{2}\left(\frac{1}{2} f_4(1,2,3) + \frac{1}{2} f_4(2,3,3)\right) = \frac{1}{2}\mathbf{b}_{4,2} + \frac{1}{2}\left(\frac{1}{2}\mathbf{d}_3 + \frac{1}{2}\mathbf{d}_4\right)
\end{aligned}$$

4.5 Algorithms for B-spline Curves

In this section, $F(u) = \sum_{i=0}^{m-1} N_i^n(u)\,\mathbf{d}_i$ is a B-spline curve of degree n over a knot vector $t_0 \le \cdots \le t_{m+n}$, where multiple knots are allowed.

4.5.1 The de Boor Algorithm

The de Boor algorithm is a method for evaluating B-spline curves.
Given: control points $\mathbf{d}_0, \ldots, \mathbf{d}_{m-1}$ and knot vector $T = (t_i)$ of a B-spline curve $F(u) = \sum_{i=0}^{m-1} N_i^n(u)\,\mathbf{d}_i$ and a parameter intervall $t_j \le u < t_{j+1}$.
Find: value of F at u

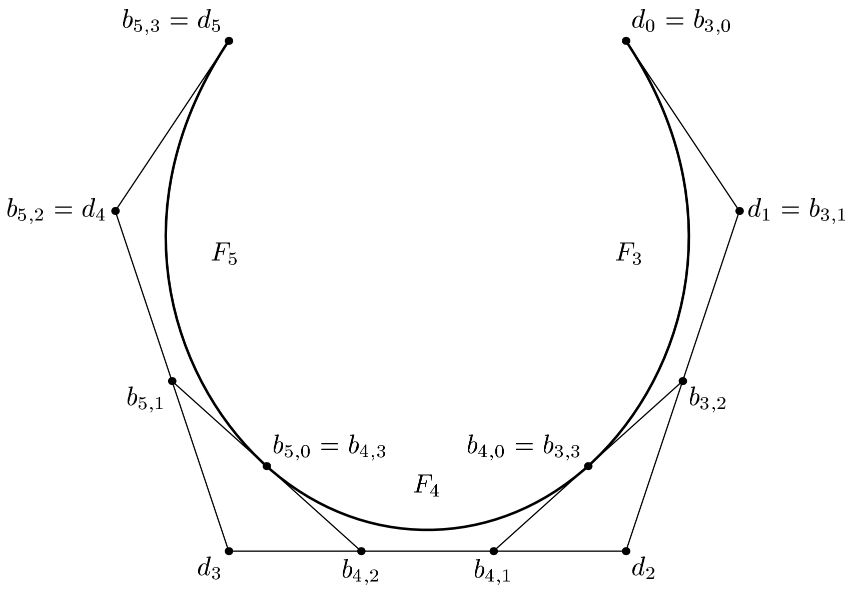


Figure 4.10: Bézier representation of a B-spline curve with endpoint interpolation

Algorithm 4.1 (de Boor)

$$\begin{aligned} \mathbf{d}_i^0 &:= \mathbf{d}_i\,, \qquad i = j-n, \dots, j \\ \mathbf{d}_i^k &:= \frac{t_{i+n+1-k} - u}{t_{i+n+1-k} - t_i}\,\mathbf{d}_{i-1}^{k-1} + \frac{u - t_i}{t_{i+n+1-k} - t_i}\,\mathbf{d}_i^{k-1}\,, \\ &\quad \text{with} \qquad k = 1, \dots, n \qquad \text{and} \qquad i = j-n+k, \dots, j \end{aligned} \tag{4.12}$$

Then: $F(u) = \mathbf{d}_j^n$.

The evaluation scheme for the de Boor algorithm is depicted in Figure 4.12.The coefficients can be visualized as divide ratios of line segments (see Figure 4.11).

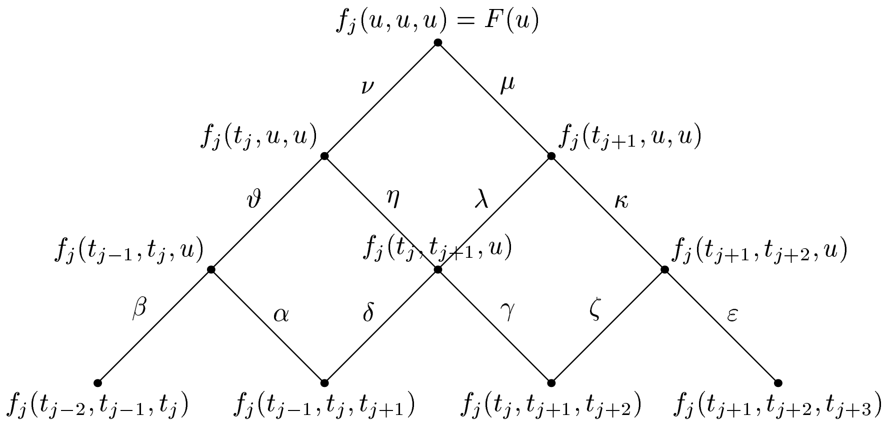


Figure 4.11: Evaluation schema of the de Boor algorithm for $n = 3$

Remark: The de Boor algorithm is the analogue to the de Casteljau algorithm for Bézier curves.

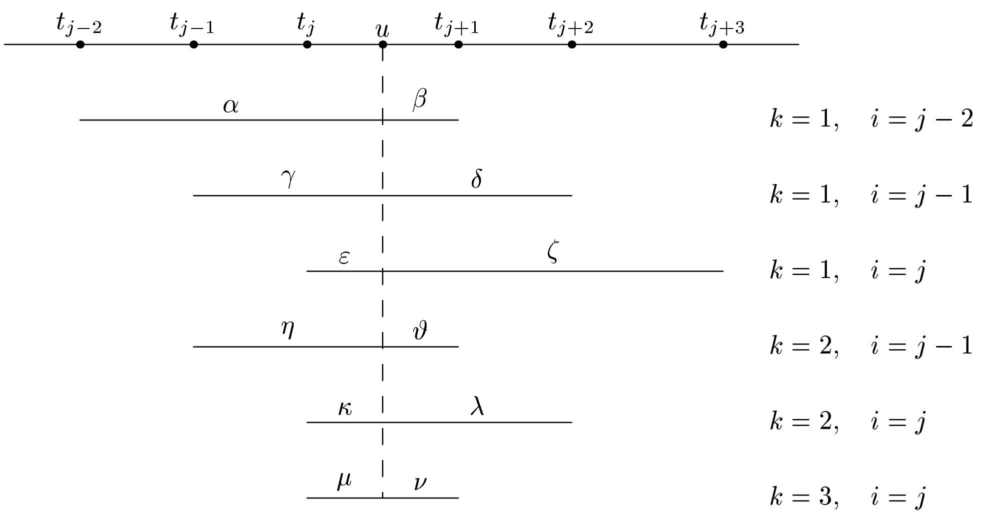


Figure 4.12: Visualization of the coefficients for the de Boor algorithm ($n = 3$)

Example: $n = 3,\ m = 5,\ T = (0, 0, 0, 0, 1, 2, 3, 3, 3, 3)\ , I_4 = [t_4, t_5) = [1, 2)\ ,\ u = \frac{3}{2} \in I_4$ (see fig. 4.13, 4.14 and 4.15).

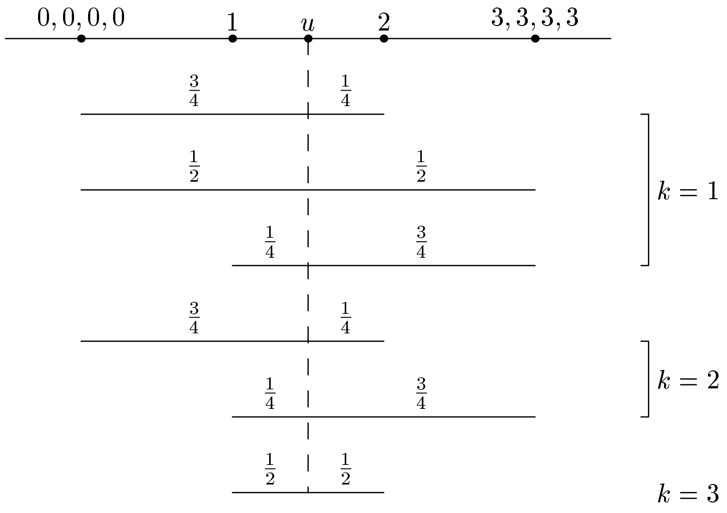


Figure 4.13: Example for the de Boor algorithm (1): coefficients

Now consider the following knot vector:

$$0 = t_1 = \cdots = t_n < t_{n+1} = t_{n+2} = \cdots = t_{2n-1} = 1.$$

Here: $u \in [0, 1]$, t_0 and t_n have all the multiplicity n. Then:

$$N_i^k(u) = u\, N_i^{k-1} + (1-u)\, N_{i+1}^{k-1}(u).$$

This is the de Casteljau algorithm!

Corollary 4.1

– As we only consider affine transformations, assumption of the interval $u \in [0, 1]$ is not a loss of generality.

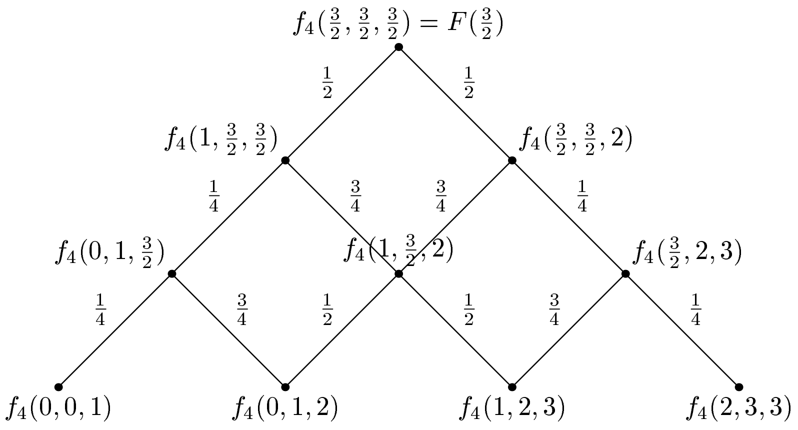


Figure 4.14: Example for the de Boor algorithm (2): evaluation

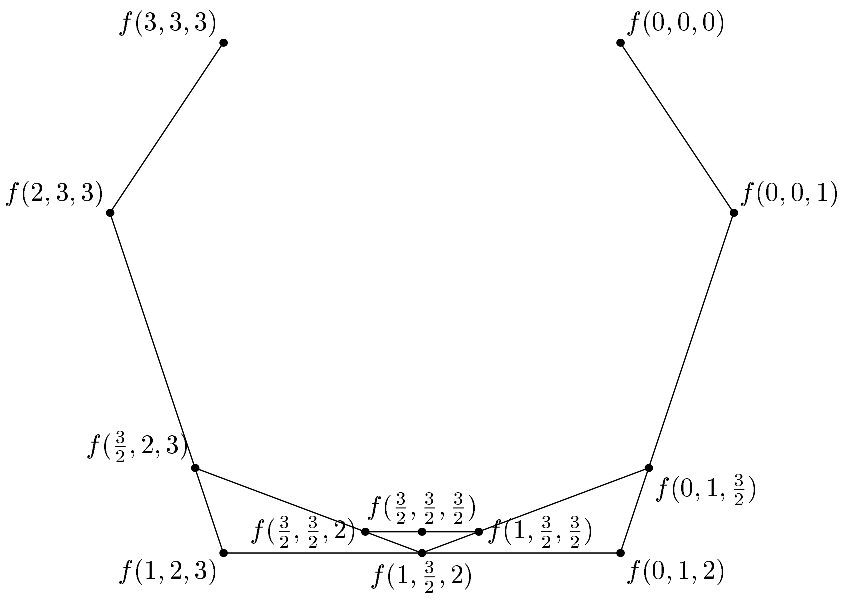


Figure 4.15: Example for the de Boor algorithm (3): graphical construction

- If two neighbouring knots have the multiplicity n, then the corresponding B-spline curve is a Bézier curve between these knots.
- By insertion of all knots until multiplicity n we can construct the Bézier polygon.

4.5.2 Conversion of B-spline Curves into Bézier Splines

Given: B-spline curve $F(u) = \sum_{i=0}^{m-1} N_i^n(u)\,\mathbf{d}_i$ over a knot vector $t_0 \leq t_1 \leq \cdots \leq t_{m+n}$.

Find: Bézier representation for each polynomial segment $F\,|_{[t_j,t_{j+1}]}$.

There are two possibilities:

(1) Insert knots t_j and t_{j+1}, until both have the multiplicity n.

(2) Use the polar form.

Example: Knot vector: $T = (\ldots, -2, -1, 0, 1, 2, 3, \ldots)$. Consider interval $[1, 2]$.

The control points $f(-2,-1,0), f(-1,0,1), f(0,1,2), f(1,2,3), \ldots$ are given.

To find the Bézier points $f(1,1,1), f(1,1,2), f(1,2,2)$ and $f(2,2,2)$, construct suitable affine combinations, e. g.

$$f(1,1,1) = \frac{1}{6}\,f(-1,0,1) + \frac{2}{3}\,f(0,1,2) + \frac{1}{6}\,f(1,2,3).$$

4.5.3 The Boehm Algorithm

How can we insert a new knot into a knot vector without changing the curve? Insertion of knots is useful for:

- adding local detail
- as basis of other algorithms (e. g. de Boor's algorithm for curve evaluation)
- to convert B-splines into Bézier splines.

Insertion of an additional knot $\bar{t}$, let's say at $t_j \leq \bar{t} < t_{j+1}$, gives a new knot vector $\bar{t}_0 \leq \cdots \leq \bar{t}_{j-1} \leq \bar{t}_j \leq \bar{t}_{j+1} \leq \bar{t}_{j+2} \leq \cdots \leq \bar{t}_{m+n+1}$.

$$\begin{array}{ccccccccccc} T & = & (t_0, & \ldots, & t_{j-1}, & t_j, & \boxed{\bar{t}}, & t_{j+1}, & t_{j+2}, & \ldots, & t_{n+m}) \\ & & \downarrow & & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & & \downarrow \\ \bar{T} & = & (\bar{t}_0, & \ldots, & \bar{t}_{j-1}, & \bar{t}_j, & \boxed{\bar{t}_{j+1}}, & \bar{t}_{j+2}, & \bar{t}_{j+3}, & \ldots, & \bar{t}_{n+m+1}) \end{array}$$

With the theorem of Curry-Schoenberg (theorem 4.3) we get: every B-spline curve F over the knot vector T has also a unique representation over the knot vector $\bar{T}$.

$$F(u) = \sum_{i=0}^{m-1} N_i^n(u)\,\mathbf{d}_i = \sum_{i=0}^{m} \bar{N}_i^n(u)\,\bar{\mathbf{d}}_i$$

How to compute the new control points $\bar{\mathbf{d}}_i$ from the given $\mathbf{d}_i$? An answer is given by the Boehm[2] algorithm.
With theorem 4.2, we get $\mathbf{d}_l = f_j(t_{l+1}, \dots, t_{l+n})$ and also $\bar{\mathbf{d}}_l = f_j(\bar{t}_{l+1}, \dots, \bar{t}_{l+n})$. Then:

$$\begin{array}{lllllllll}
\bar{\mathbf{d}}_{j-3} & = & f(\bar{t}_{j-2}, & \bar{t}_{j-1}, & \bar{t}_j) & = & f(t_{j-2}, & t_{j-1}, & t_j) & = & \mathbf{d}_{j-3} \\
\bar{\mathbf{d}}_{j-2} & = & f(\bar{t}_{j-1}, & \bar{t}_j, & \bar{t}_{j+1}) & = & f(t_{j-1}, & t_j, & \bar{t}) & & \\
\bar{\mathbf{d}}_{j-1} & = & f(\bar{t}_j, & \bar{t}_{j+1}, & \bar{t}_{j+2}) & = & f(t_j, & \bar{t}, & t_{j+1}) & & \\
\bar{\mathbf{d}}_j & = & f(\bar{t}_{j+1}, & \bar{t}_{j+2}, & \bar{t}_{j+3}) & = & f(\bar{t}, & t_{j+1}, & t_{j+2}) & & \\
\bar{\mathbf{d}}_{j+1} & = & f(\bar{t}_{j+2}, & \bar{t}_{j+3}, & \bar{t}_{j+4}) & = & f(t_{j+1}, & t_{j+2}, & t_{j+3}) & = & \mathbf{d}_j
\end{array}$$

By construction of suitable affine combinations we get e. g.

$$\begin{aligned}
\bar{\mathbf{d}}_{j-2} = f(t_{j-1}, t_j, \bar{t}) &= \frac{t_{j+1} - \bar{t}}{t_{j+1} - t_{j-2}} f(t_{j-2}, t_{j-1}, t_j) + \frac{\bar{t} - t_{j-2}}{t_{j+1} - t_{j-2}} f(t_{j-1}, t_j, t_{j+1}) = \\
&= \frac{t_{j+1} - \bar{t}}{t_{j+1} - t_{j-2}} \mathbf{d}_{j-3} + \frac{\bar{t} - t_{j-2}}{t_{j+1} - t_{j-2}} \mathbf{d}_{j-2}
\end{aligned}$$

The other control points are obtained similar. More general:

Algorithm 4.2 (Boehm)

$$\bar{\mathbf{d}}_i = \begin{cases} \mathbf{d}_i & i \le j - n \\ \frac{t_{i+n} - \bar{t}}{t_{i+n} - t_i} \mathbf{d}_{i-1} + \frac{\bar{t} - t_i}{t_{i+n} - t_i} \mathbf{d}_i & j - n + 1 \le i \le j \\ \mathbf{d}_{i-1} & j + 1 \le i \end{cases} \tag{4.13}$$

The coefficients can be visualized as divide ratios of line segments. In the following schema, there is $\bar{\mathbf{d}}_{j-2} = \alpha\, \mathbf{d}_{j-3} + \beta\, \mathbf{d}_{j-2}$, $\bar{\mathbf{d}}_{j-1} = \gamma\, \mathbf{d}_{j-2} + \delta\, \mathbf{d}_{j-1}$ and $\bar{\mathbf{d}}_j = \varepsilon\, \mathbf{d}_{j-1} + \zeta\, \mathbf{d}_j$.

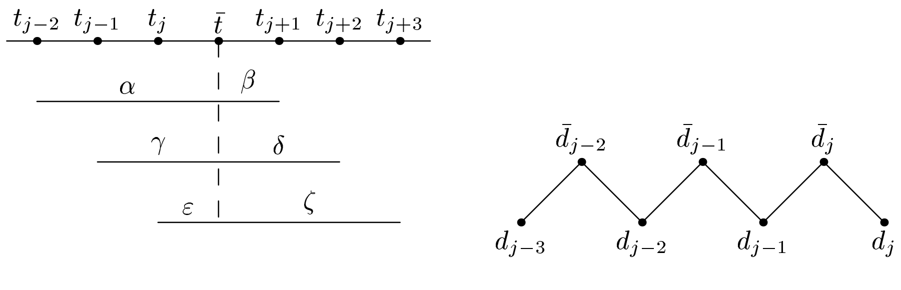


Figure 4.16: Schema for the Boehm algorithm

[2]Wolfgang Boehm, 1980

Example: $n = 3\,,\ T = (0,1,2,4,5,6),\ \bar{t} = 3.$

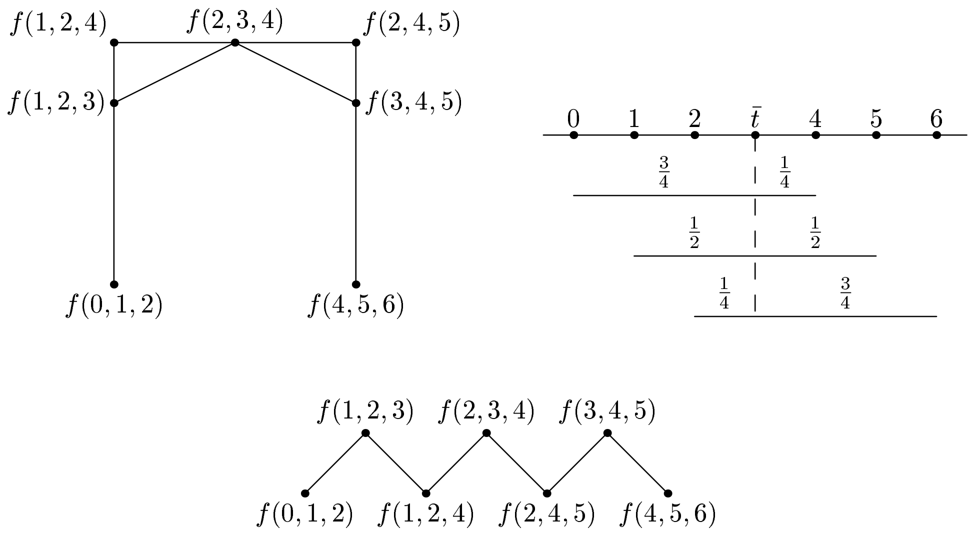


Figure 4.17: Example for the Boehm algorithm

Remark:

- According to Theorem 4.3 we can insert inner knots up to multiplicity n and at the end points up to mulitplicity $n+1$.
- Insertion of an existing knot (i. e. increasing the multiplicity of the knot to $\bar{\mu}$) gives only $n - \bar{\mu} + 1$ new control points.
- Insertion of knots until multiplicity n makes the B-spline curve interpolate control points. This gives the de Boor algorithm to evaluate B-spline curves.

4.5.4 Knot Deletion

Knot deletion is the inverse process to knot insertion. In general, it is not possible to remove a knot without changing the curve. Therefore, try to remove a knot and find a curve that is as close as possible to the original curve (see degree elevation $\leftrightarrow$ degree reduction for Bézier curves).
Knot insertion is a linear process:

$$\begin{pmatrix} \bar{\mathbf{d}}_{j-3} \\ \bar{\mathbf{d}}_{j-2} \\ \bar{\mathbf{d}}_{j-1} \\ \bar{\mathbf{d}}_{j} \\ \bar{\mathbf{d}}_{j+1} \end{pmatrix} = \begin{pmatrix} 1 & & & \\ \alpha_{j-2} & \beta_{j-2} & & \\ & \alpha_{j-1} & \beta_{j-1} & \\ & & \alpha_j & \beta_j \\ & & & 1 \end{pmatrix} \begin{pmatrix} \mathbf{d}_{j-3} \\ \mathbf{d}_{j-2} \\ \mathbf{d}_{j-1} \\ \mathbf{d}_{j} \end{pmatrix} \tag{4.14}$$

Short: $\bar{\mathbf{d}} = \mathbf{A}\,\mathbf{d}$. Find control points $\mathbf{d}$ such that $||\bar{\mathbf{d}} - \mathbf{A}\,\mathbf{d}||$ is minimal. With theorem 3.6 we have to solve the linear system

$$\mathbf{A}^T \mathbf{A}\,\mathbf{d} = \mathbf{A}^T\,\bar{\mathbf{d}} \tag{4.15}$$

and get

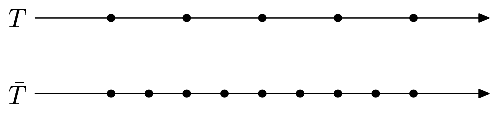
$$\mathbf{A}^T\mathbf{A} = \begin{pmatrix} 1+\alpha_{j-2}^2 & \alpha_{j-2}\beta_{j-2} & & \\ \alpha_{j-2}\beta_{j-2} & \alpha_{j-1}^2+\beta_{j-2}^2 & \alpha_{j-1}\beta_{j-1} & \\ & \alpha_{j-1}\beta_{j-1} & \alpha_j^2+\beta_{j-1}^2 & \alpha_j\beta_j \\ & & \alpha_j\beta_j & \beta_j^2+1 \end{pmatrix}$$

4.5.5 Subdivision for uniform B-spline curves

Intention: Create a series of polygons that converge fast to the curve. The subdivison of B-spline curves is achieved by insertion of many additional knots or by doubling the number of knots.

Let $F(u)=\sum_{i=0}^{m-1} N_i^n(u)\,\mathbf{d}_i$ a B-spline curve of degree n over the knot vector $T=(\cdots \le t_i \le t_{i+1} \le \cdots)$ with $t_{i+1}-t_i=h$ for every i or simply $t_i = i\cdot h.$ A knot vector with this property is called *uniform* knot vector.

Insert new knots at $i\cdot\frac{h}{2}$:



Then: $F(u)=\sum_{i=0}^{2m-1} N_i^n(u)\,\bar{\mathbf{d}}_i$ over the new knot vector $\bar{T}$.

Repeating this process gives a sequence of polygons that coverges *fast* to $F(u)$ (compared with subdivision for Bézier curves).
The $\bar{\mathbf{d}}_i$ are obtained again with suitable affine combinations. For $n=2$ this gives Chaikin's[3] algorithm.

Example 1: $n=2$ and knot vector with multiplicity $n+1$ at the endpoints
$T=(0,0,0,1,2,3,4,5,6,7,7,7)$
Knots are inserted where the curve has full support, in this example at $u=\frac{5}{2},\frac{7}{2},\frac{9}{2}$.

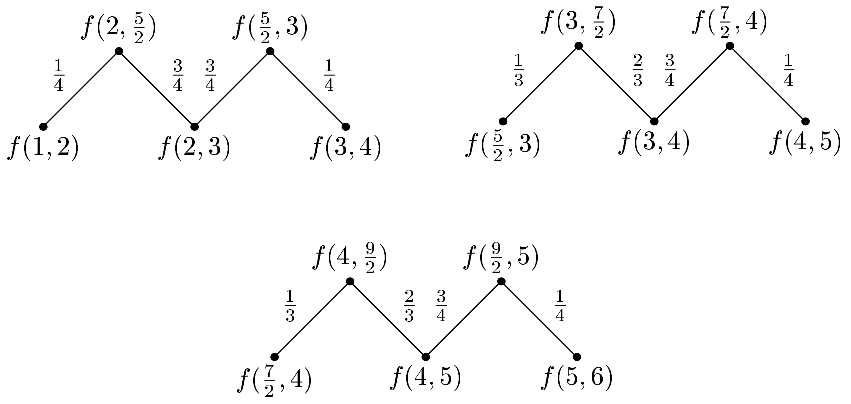


Figure 4.18: Knot insertion at $u=\frac{5}{2},\frac{7}{2},\frac{9}{2}$

[3]G. Chaikin, 1995

According to Figure 4.18 we obtain the following new control points:

$$\begin{aligned}
\bar{\mathbf{d}}_0 &= f(1,2) = \mathbf{d}_0 \\
\bar{\mathbf{d}}_1 &= f(2,\frac{5}{2}) = \frac{1}{4}(\mathbf{d}_0 + 3\mathbf{d}_1) \\
\bar{\mathbf{d}}_2 &= f(\frac{5}{2},3) = \frac{1}{4}(3\mathbf{d}_1 + \mathbf{d}_2) \\
\bar{\mathbf{d}}_3 &= f(3,\frac{7}{2}) = \frac{1}{3}\left(\frac{3}{4}\mathbf{d}_1 + \frac{1}{4}\mathbf{d}_2\right) + \frac{2}{3}\mathbf{d}_2 = \frac{1}{4}(\mathbf{d}_1 + 3\mathbf{d}_2) \\
\bar{\mathbf{d}}_4 &= f(\frac{7}{2},4) = \frac{1}{4}(3\mathbf{d}_2 + \mathbf{d}_3) \\
\bar{\mathbf{d}}_5 &= f(4,\frac{9}{2}) = \frac{1}{3}\left(\frac{3}{4}\mathbf{d}_3 + \frac{1}{4}\mathbf{d}_4\right) + \frac{2}{3}\mathbf{d}_3 = \frac{1}{4}(\mathbf{d}_2 + 3\mathbf{d}_3) \\
\bar{\mathbf{d}}_6 &= f(\frac{9}{2},5) = \frac{1}{4}(3\mathbf{d}_3 + \mathbf{d}_4)
\end{aligned}$$

Therefore we can calculate the new control points from m old control points as follows:

$$\begin{aligned}
\bar{\mathbf{d}}_0 &= \mathbf{d}_0 \\
\bar{\mathbf{d}}_{2i} &= \frac{1}{4}(3\mathbf{d}_i + \mathbf{d}_{i+1}) \qquad \text{for } i = 1,\dots,m-2 \\
\bar{\mathbf{d}}_{2i+1} &= \frac{1}{4}(\mathbf{d}_i + 3\mathbf{d}_{i+1}) \qquad \text{for } i = 1,\dots,m-3 \\
\bar{\mathbf{d}}_{m+2} &= \mathbf{d}_{m-1}
\end{aligned}$$

The graphical evaluation is also called "Corner Cutting". You can see the result after two steps of iteration in Figure 4.19.

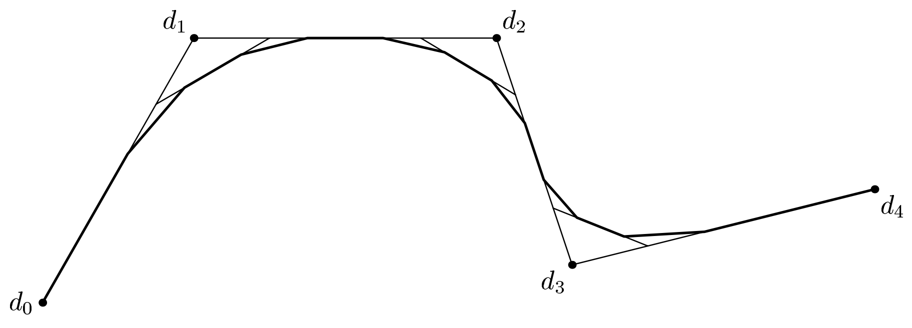


Figure 4.19: Geometrical construction: cutting the corners

Example 2: $n = 2$ and *open* knot vector e.g. $T = (0,1,2,3,4,5,6,7)$
Evaluation according to the following scheme (see Figure 4.20):

$$\begin{aligned}
\bar{\mathbf{d}}_{2i} &= \frac{1}{4}(3\mathbf{d}_i + \mathbf{d}_{i+1}) \qquad \text{for } i = 0,\dots,m-2 \\
\bar{\mathbf{d}}_{2i+1} &= \frac{1}{4}(\mathbf{d}_i + 3\mathbf{d}_{i+1}) \qquad \text{for } i = 0,\dots,m-2
\end{aligned}$$

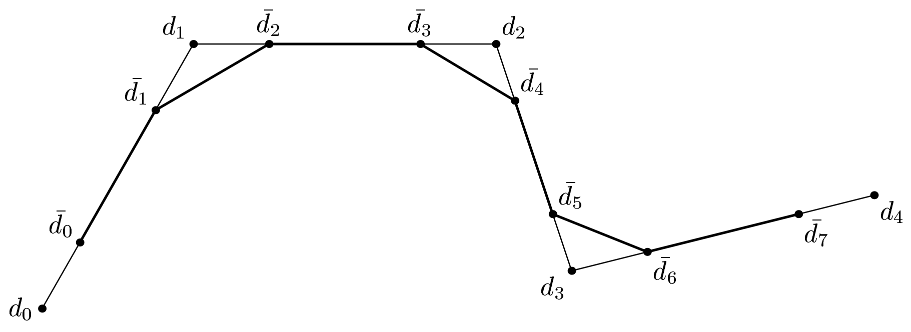


Figure 4.20: Subdivision with open knot vector and $n = 2$

For $n = 3$ the algorithm is more complicated, as new control points can depend on up to three old control points:
Example 3: $n = 3$ and knot vector with multiplicity $n + 1$ at the endpoints, m control points
Evaluation according to the following scheme:

$$
\begin{aligned}
\bar{\mathbf{d}}_0 &= \mathbf{d}_0 \\
\bar{\mathbf{d}}_1 &= \frac{1}{2}(\mathbf{d}_0 + \mathbf{d}_1) \\
\bar{\mathbf{d}}_2 &= \frac{1}{4}(3\mathbf{d}_1 + \mathbf{d}_2) \\
\bar{\mathbf{d}}_3 &= \frac{1}{16}(3\mathbf{d}_1 + 11\mathbf{d}_2 + 2\mathbf{d}_3) \\
\bar{\mathbf{d}}_{2i} &= \frac{1}{2}(\mathbf{d}_i + \mathbf{d}_{i+1}) && \text{for } i = 4, \dots, m-4 \\
\bar{\mathbf{d}}_{2i+1} &= \frac{1}{8}(\mathbf{d}_i + 6\mathbf{d}_{i+1} + \mathbf{d}_{i+2}) && \text{for } i = 5, \dots, m-5 \\
\bar{\mathbf{d}}_{2(m-4)+1} &= \frac{1}{16}(3\mathbf{d}_{m-1} + 11\mathbf{d}_{m-2}\mathbf{d}_{m-3}) \\
\bar{\mathbf{d}}_{2(m-3)} &= \frac{1}{4}(3\mathbf{d}_{m-1} + \mathbf{d}_{m-2}) \\
\bar{\mathbf{d}}_{2(m-3)+1} &= \frac{1}{2}(\mathbf{d}_m + \mathbf{d}_{m-1}) \\
\bar{\mathbf{d}}_{2(m-2)} &= \mathbf{d}_m
\end{aligned}
$$

Example 4: $n = 3$ and *open* knot vector, m control points
Evaluation according to the following scheme:

$$
\begin{aligned}
\bar{\mathbf{d}}_{2i} &= \frac{1}{2}(\mathbf{d}_i + \mathbf{d}_{i+1}) && \text{for } i = 0, \dots, m-2 \\
\bar{\mathbf{d}}_{2i+1} &= \frac{1}{8}(\mathbf{d}_i + 6\mathbf{d}_{i+1} + \mathbf{d}_{i+2}) && \text{for } i = 0, \dots, m-3
\end{aligned}
$$

4.6 Interpolation and Approximation

Given:

- M data points $\mathbf{c}_0, \dots, \mathbf{c}_{M-1}$ with M corresponding parameter values $x_0, \dots, x_{M-1}$
- an m-dimensional space with basis functions $N_0^n, \dots, N_{m-1}^n$ over a knot vector $T = (t_0 \leq \dots \leq t_{m+n})$

Find: Curve $F(u) = \sum_{i=0}^{m-1} N_i^n(u)\mathbf{d}_i$ that interpolates / approximates the data points
Remark: The given points $\mathbf{c}_0, \dots, \mathbf{c}_{M-1}$ are to be interpolated / approximated at the parameter values $x_0, \dots, x_{M-1}$. Imagine the knot vector is the time axis and the parameter u the time: The curve we want to find, should run through the point $\mathbf{c}_i$ at time x_i (see Figure 4.21).

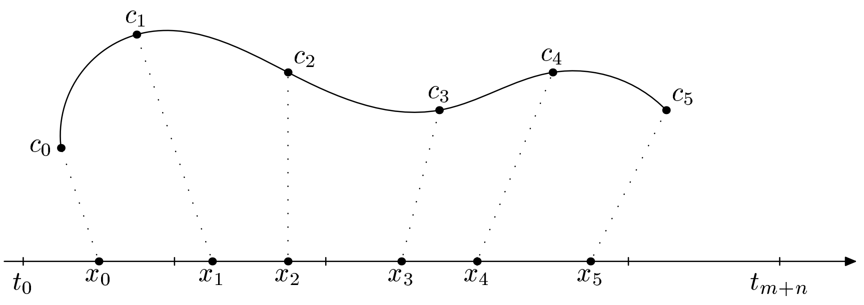


Figure 4.21: Illustration of parameter values

There are three cases to distinguish:

(1) $M = m$: number of conditions = number of degrees of freedom $\Rightarrow$ interpolation

(2) $M > m$: overdetermined system of equations $\Rightarrow$ approximation

(3) $M < m$: underdetermined system of equations $\Rightarrow$ "constrained" interpolation/approximation

4.6.1 B-spline Interpolation

Given:

- m points $\mathbf{c}_0, \dots, \mathbf{c}_{m-1}$ with m parameter values $x_0, \dots, x_{m-1}$
- an m-dimensional space: $F(u) = \sum_{i=0}^{m-1} N_i^n(u)\,\mathbf{d}_i$ with knot vector T

Find: control points $\mathbf{d}_0, \dots, \mathbf{d}_{m-1}$ such that the curve $F(u) = \sum_{i=0}^{m-1} N_i^n(u)\,\mathbf{d}_i$ interpolates the given points $\mathbf{c}_0, \dots, \mathbf{c}_{m-1}$ at the parameter values $x_0, \dots, x_{m-1}$, i. e. $F(x_i) \overset{!}{=} \mathbf{c}_i\ \forall i$.
In matrix notation:

$$\underbrace{\begin{pmatrix} N_0^n(x_0) & \cdots & N_{m-1}^n(x_0) \\ \vdots & & \vdots \\ N_0^n(x_{m-1}) & \cdots & N_m^n(x_{m-1}) \end{pmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} \mathbf{d}_0 \\ \vdots \\ \mathbf{d}_{m-1} \end{pmatrix}}_{\mathbf{d}} = \underbrace{\begin{pmatrix} \mathbf{c}_0 \\ \vdots \\ \mathbf{c}_{m-1} \end{pmatrix}}_{\mathbf{c}} \tag{4.16}$$

Therefore, the interpolation problem is always solvable in case the matrix $\left(N_j^n(x_i)\right)_{i,j}$ is invertible.
Remark:

- Matrix $\mathbf{A}$ has at most $n+1$ non-zero entries per row (support of N_i^n: $\operatorname{supp}(N_i^n) := [t_i, t_{i+n+1}]$).
- Matrix $\mathbf{A}$ is a band diagonal matrix. There are efficent numerical methods to solve a linear system consisting of a band diagonal matrix and a vector.

Theorem 4.5 (Schoenberg-Whitney)

$$\text{Matrix A is invertible} \quad \Leftrightarrow \quad t_i \le x_i < t_{i+n+1} \; , \qquad i = 0, ..., m-1$$

Proof: See [1]. A choice of the parameter values x_i that always fulfills the requirements of Theorem 4.5 are the *Greville abscissas*.

Definition 4.3
The parameter values $t_i^* := \frac{1}{n}\left(t_{i+1} + \cdots + t_{i+n}\right)$ are called *Greville abscissas*.

Remark:

- For the *Greville abscissas* holds: $F(u) = u = \sum_i N_i^n(u)\, t_i^*$
- $x_i = t_i^*$ fulfills $t_i \le x_i < t_{i+n+1}$, if all inner knots in T have multiplicity $\mu_i \le n$. Reason for this: $n \cdot t_i \le t_{i+1} + \cdots + t_{i+n} \lneq n \cdot t_{i+n+1}$.
- Applications typically only give the $\mathbf{c}_0, \ldots, \mathbf{c}_{m-1}$, but no parameterization. In this case, choose a parameterization that is suitable for the application.

4.6.2 Special case: Cubic spline interpolation at the knots

Given:

- degree $n = 3$
- a m-dimensional space
- $m-2$ data points $\mathbf{c}_0, \ldots, \mathbf{c}_{m-3}$
- knot vector $T = (t_0 \le \ldots \le t_{m+3})$

Find: cubic B-spline curve $F(u)$ over T with $F(t_{i+3}) = \mathbf{c}_i$
That means: we choose those knots as parameter values that lie in the area where the curve has full support. Then the conditions of Theorem 4.5 are fulfilled. But the number of conditions is smaller than the number of degrees of freedom. Hence for the uniqueness of the curve we need additional conditions.

Possible additional conditions:

(1) Specification of tangents at the end points: $F'(t_3) = \xi_0\,,\ F'(t_m) = \xi_1$

(2) Natural end condition: $F''(t_3) = 0\,,\ F''(t_m) = 0$

(3) In the case of closed curves (i. e. $F(t_3) = F(t_m)$):
$F'(t_3) = F'(t_m)\,,\ F''(t_3) = F''(t_m)$
The degrees of freedom are used to create C^2-continuity at the point $\mathbf{c}_0 = \mathbf{c}_{m-3}$

Example 1: Interpolation at the knots of an *arbitrary* knot vector by a cubic spline with additional condition 2

$$F'(u) = \sum_{i=0}^{m-2} N^2_{i+1}(u) \frac{3}{t_{i+4} - t_{i+1}} (\mathbf{d}_{i+1} - \mathbf{d}_i)$$

$$F''(u) = \sum_{i=0}^{m-3} N^1_{i+2}(u) \frac{2}{t_{i+4} - t_{i+2}} \left(\frac{3}{t_{i+5} - t_{i+2}} (\mathbf{d}_{i+2} - \mathbf{d}_{i+1}) - \frac{3}{t_{i+4} - t_{i+1}} (\mathbf{d}_{i+1} - \mathbf{d}_i) \right)$$

$$F''(t_3) = \underbrace{\frac{6}{(t_4 - t_2)(t_4 - t_1)}}_{a_0} \mathbf{d}_0 \underbrace{- \frac{6}{t_4 - t_2} \left(\frac{1}{t_5 - t_2} + \frac{1}{t_4 - t_1} \right)}_{a_1} \mathbf{d}_1 + \underbrace{\frac{6}{(t_4 - t_2)(t_5 - t_2)}}_{a_2} \mathbf{d}_2$$

$$F''(t_m) = \underbrace{\frac{6}{(t_{m+1} - t_{m-1})(t_{m+1} - t_{m-2})}}_{b_{m-3}} \mathbf{d}_{m-3} -$$

$$\underbrace{- \frac{6}{t_{m+1} - t_{m-1}} \left(\frac{1}{t_{m+2} - t_{m-1}} + \frac{1}{t_{m+1} - t_{m-2}} \right)}_{b_{m-2}} \mathbf{d}_{m-2} + \underbrace{\frac{6}{(t_{m+1} - t_{m-1})(t_{m+2} - t_{m-1})}}_{b_{m-1}} \mathbf{d}_{m-1}$$

Linear system to solve:

$$\begin{pmatrix} a_0 & a_1 & a_2 & & & \\ N^3_0(t_3) & N^3_1(t_3) & N^3_2(t_3) & & & \\ & N^3_1(t_4) & N^3_2(t_4) & N^3_3(t_4) & & \\ & \ddots & \ddots & \ddots & & \\ & & N^3_{m-3}(t_m) & N^3_{m-2}(t_m) & N^3_{m-1}(t_m) & \\ & & b_{m-3} & b_{m-2} & b_{m-1} & \end{pmatrix} \begin{pmatrix} \mathbf{d}_0 \\ \mathbf{d}_1 \\ \mathbf{d}_2 \\ \vdots \\ \mathbf{d}_{m-2} \\ \mathbf{d}_{m-1} \end{pmatrix} = \begin{pmatrix} 0 \\ \mathbf{c}_0 \\ \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_{m-3} \\ 0 \end{pmatrix}$$

Example 2: Interpolation at the knots of an *end point interpolating* knot vector

$T = (t_0 = \ldots = t_3 < \ldots < t_m = \ldots = t_{m+3})$ by a cubic spline with additional condition 2

$$\begin{aligned} t_0 = \ldots = t_3 \quad &\Rightarrow\ N^3_1(t_3) = N^3_2(t_3) = 0\,,\ N^3_0(t_3) = 1 &&\Rightarrow\ \mathbf{d}_0 = \mathbf{c}_0 \\ t_m = \ldots = t_{m+3} \quad &\Rightarrow\ N^3_{m-3}(t_m) = N^3_{m-2}(t_m) = 0\,,\ N^3_{m-1}(t_m) = 1 &&\Rightarrow\ \mathbf{d}_{m-1} = \mathbf{c}_{m-1} \end{aligned}$$

$$\Rightarrow\ a_0\mathbf{d}_0 + a_1\mathbf{d}_1 + a_2\mathbf{d}_2 = 0 \ \Leftrightarrow\ -a_1\mathbf{d}_1 - a_2\mathbf{d}_2 = a_0\mathbf{d}_0 = a_0\mathbf{c}_0$$

$$\Leftrightarrow\ \underbrace{-\frac{a_1}{a_0}}_{a_1'}\mathbf{d}_1 \underbrace{-\frac{a_2}{a_0}}_{a_2'}\mathbf{d}_2 = \mathbf{c}_0$$

analogue: $b'_{m-3}\mathbf{d}_{m-3} + b'_{m-2}\mathbf{d}_{m-2} = c_{m-3}$

The linear system that we have to solve is derived by modification of the linear system of Example 1:

$$\begin{pmatrix} a_1' & a_2' & & & \\ N_1^3(t_4) & N_2^3(t_4) & N_3^3(t_4) & & \\ & N_2^3(t_5) & N_3^3(t_5) & N_4^3(t_5) & \\ & & \ddots & \ddots & \\ & & N_{m-4}^3(t_{m-1}) & N_{m-3}^3(t_{m-1}) & N_{m-2}^3(t_{m-1}) \\ & & & b'_{m-3} & b'_{m-2} \end{pmatrix} \begin{pmatrix} \mathbf{d}_1 \\ \mathbf{d}_2 \\ \mathbf{d}_3 \\ \vdots \\ \mathbf{d}_{m-3} \\ \mathbf{d}_{m-2} \end{pmatrix} = \begin{pmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \mathbf{c}_2 \\ \vdots \\ \mathbf{c}_{m-3} \\ \mathbf{c}_{m-2} \end{pmatrix}$$

with

$$a_1' = -\frac{a_1}{a_0} = \frac{t_4 - t_3}{t_5 - t_3} + 1 \qquad b'_{m-3} = -\frac{b_{m-3}}{b_{m-1}} = -\frac{t_m - t_{m-1}}{t_m - t_{m-2}}$$

$$a_2' = -\frac{a_2}{a_0} = -\frac{t_4 - t_3}{t_5 - t_3} \qquad b'_{m-2} = -\frac{b_{m-2}}{b_{m-1}} = \frac{t_m - t_{m-1}}{t_m - t_{m-2}} + 1$$

Example 3: Interpolation at the knots of an *end point interpolating* and *uniform* knot vector $T = (0, 0, 0, 0, 1, \dots, m-2, m-3, m-3, m-3, m-3)$ by a cubic spline with additional condition 2
The linear system $\mathbf{A} \cdot \mathbf{d} = \mathbf{c}$ is:

$$\begin{pmatrix} 3/2 & -1/2 & & & & \\ 1/4 & 7/12 & 1/6 & & & \\ & 1/6 & 4/6 & 1/6 & & \\ & & \ddots & \ddots & & \\ & & 1/6 & 4/6 & 1/6 & \\ & & & 1/4 & 7/12 & 1/6 \\ & & & & 3/2 & -1/2 \end{pmatrix} \begin{pmatrix} \mathbf{d}_1 \\ \mathbf{d}_2 \\ \vdots \\ \vdots \\ \vdots \\ \mathbf{d}_{m-3} \\ \mathbf{d}_{m-2} \end{pmatrix} = \begin{pmatrix} \mathbf{c}_0 \\ \mathbf{c}_1 \\ \vdots \\ \vdots \\ \vdots \\ \mathbf{c}_{m-3} \\ \mathbf{c}_{m-2} \end{pmatrix}$$

Remark: In the case of end point interpolating splines the matrix $\mathbf{A}$ is tridiagonal (see Example 2 and 3).

4.6.3 B-spline approximation

If there are given more data points than control points available in the m-dimensional space, then interpolation of the data points is usually impossible. The linear system $\mathbf{Ad} = \mathbf{c}$ is overdetermined and therefore not solvable.

We can use the Gauss Normal Equation (Theorem 3.6) to minimize the error $|\mathbf{Ad}-\mathbf{c}|$ and with it, to approximate the data points at the best:

$$\begin{aligned} ||\mathbf{Ad}-\mathbf{c}||^2 = (\mathbf{Ad}-\mathbf{c})^T(\mathbf{Ad}-\mathbf{c}) = \mathbf{d}^T\mathbf{A}^T\mathbf{Ad} - 2\mathbf{d}^T\mathbf{A}^T\mathbf{c} + \mathbf{c}^T\mathbf{c} \;\; &\text{is minimal} \\ \Leftrightarrow \qquad & 1.\,\text{derivation} = 0 \\ \Leftrightarrow \qquad & \mathbf{A}^T\mathbf{Ad} = \mathbf{A}^T\mathbf{c} \\ \Leftrightarrow \qquad & \mathbf{d} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{c} \end{aligned}$$

Thus, the approximation has exactly one solution if and only if the matrix $\mathbf{A}^T\mathbf{A}$ is invertible:

$$\begin{aligned} (\mathbf{A}^T\mathbf{A})^{-1} \text{ exists} \qquad \Leftrightarrow \qquad & \text{Theorem of Schoenberg-Whitney is fulfilled} \\ \Leftrightarrow \qquad & \text{columns of } \mathbf{A} \text{ are linear independent} \end{aligned}$$

4.6.4 Parametrization

An important item of interpolation and approximation of data points is the choice of the parameter values x_i (parametrization). By choosing a proper parametrization great deflections and loops of the curve can be avoided in some degree.
The easiest way to determine the x_i is to set $x_i = i$. This is called the *uniform* or *equidistant* parametrization. The disadvantage is the disregard of the position of the data points. For this reason the interpolating curve will have unwanted loops and deflections in most cases.
Interpretation: Imagine the parameter value u is the time. Because of the C^2-continuity the speed and the acceleration is continuous. On account of the uniform parametrization large distances between data points must be managed in the same time as distances between data points that are close to each other. Therefore the speed is higher between points with large distance, but speed and acceleration are continuous and we receive loops and deflections of the curve between data points with short distance.

One way to take the positions of the data points into account is to choose the distance of the parameters proportional to the distance of the data points:

$$\begin{aligned} s_i = ||\mathbf{c}_i - \mathbf{c}_{i-1}||^p\,, \qquad & i = 1,\dots,m-1 \quad \text{with} \quad p \in [0,1] \\ x_0 = 0,\; x_{i+1} = x_i + s_{i+1}\,, \quad & i = 1,\dots,m-1 \end{aligned}$$

$$\text{This parametrization is called} \begin{cases} \text{uniform for } \; p = 0 \\ \text{centripetal for } \; p = 1/2 \\ \text{chordal for } \; p = 1 \end{cases} .$$

The chordal parametrization is depicted in Figure 4.22: The distances between the parameter values x_i and x_{i+1} are equal to the distances between the data points c_i und c_{i+1}.
Affine transformation does not change the parametrization. Hence, the parameter values can determined as follows for a given interval $[a,b]$:

$$\overline{x}_i = \frac{x_i}{x_{m-1}}(b-a) + a \quad \text{for} \quad i = 0,\dots,m-1 \qquad \Rightarrow \qquad \overline{x}_0 = a\,,\; \overline{x}_{m-1} = b$$

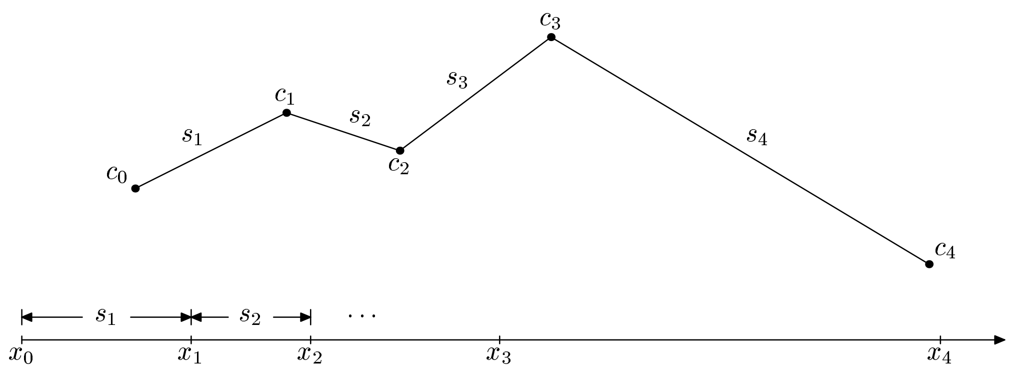


Figure 4.22: Chordal parametrisation

Remark: It is also possible to choose the knot vector T depending on the parameter values x_i. Cubic case: $t_i = \frac{1}{3}(x_{i-3} + x_{i-2} + x_{i-1})$

5 Rational Curves

A more general approach are *rational curves*. Using the *polynomial* curves we have examined so far it is not possible to represent conic sections (e. g. circle arcs). This becomes possible if we extend the schemes to rational curves. Of special interest in this chapter are rational Bézier curves and rational B-spline curves (NURBS).
A rational curve can be described in two ways:

- as a quotient of polynomials
- as central projection of a $(d+1)$-dimensional curve onto a d-dimensional plane.

Example: A two-dimensional point (x, y) is extended by another dimension to $(w\,x, w\,y, w)$ and can be represented by a line through the origin:

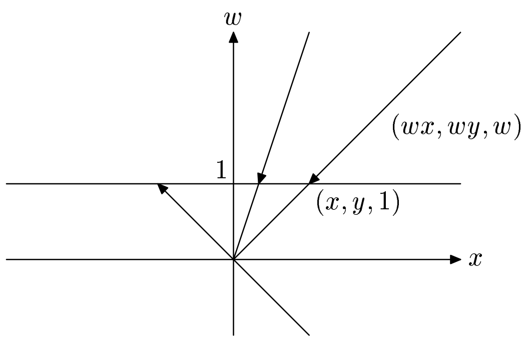


Figure 5.1: Homogeneous coordinates

By projection onto the plane $w = 1$ (division by w) we get back the 2D point.

Remark: This representation is called *homogeneous coordinates*. The aim is to represent d-dimensional affine maps as linear homogeneous transformations (i. e. matrices) in the $(d+1)$-dimensional space.

5.1 Rational Bézier Curves

Definition 5.1 (Rational Bézier curve)
A *rational Bézier curve* in $\mathbb{R}^d$ $(d = 2,\ 3)$ is the projection of a polynomial Bézier curve in $\mathbb{R}^{d+1}$.

Definition 5.2 (Rational Bézier curve, analytic definition)
Given:

- control points $\mathbf{b}_0, \ldots, \mathbf{b}_n \in \mathbb{R}^d$ (e. g. $\mathbb{R}^2$, $\mathbb{R}^3$)

  (projections of the $(d+1)$-dimensional control points $\mathbf{b}_0^h, \ldots, \mathbf{b}_n^h$)

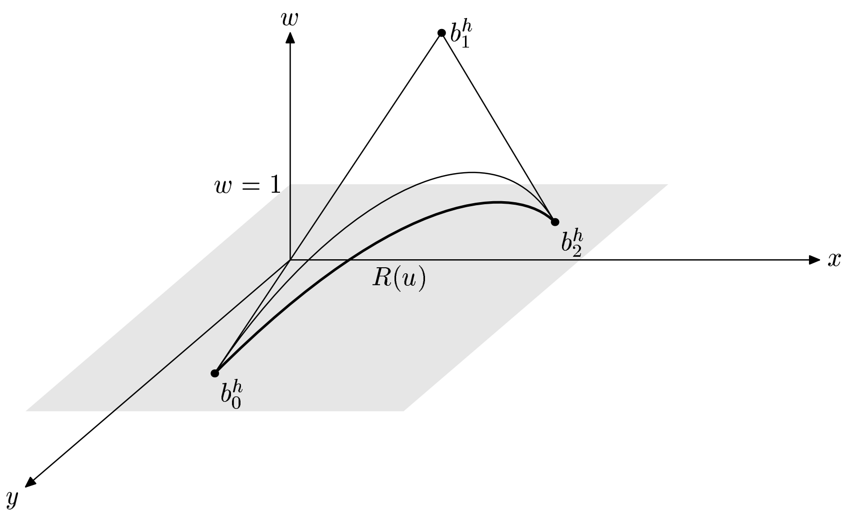


Figure 5.2: Interpretation of rat. curves with homogeneous coordinates (proj. $\mathbb{R}^3 \to \mathbb{R}^2$)

– *weights* $w_0, \dots, w_n \geq 0,\ w_i \in \mathbb{R}$

(homogeneous $(d+1)$ component of the control points)

We can assume without loss of generality: $w_0 = w_n = 1$.

Then:

$$R(u) := \frac{\sum_{i=0}^{n} B_i^n(u) \cdot w_i \cdot \mathbf{b}_i}{\sum_{i=0}^{n} B_i^n(u) \cdot w_i}, \quad u \in [0,1] \tag{5.1}$$

is called *rational Bézier curve* over the interval $[0,1]$.

5.1.1 Properties of rational Bézier curves

Let $F(u)$ a rational Bézier curve and $w_0 = w_n = 1,\ w_i \geq 0$.

(1) If all $w_i = 1$, then we get the well known polynomial Bézier curves.

(2) $F(u) \subseteq [\mathbf{b}_0, \dots, \mathbf{b}_n]$ (convex hull property)

(3) $F(0) = \mathbf{b}_0,\ F(1) = \mathbf{b}_n$ (endpoint interpolation)

(4) The curve is tangent to the control polygon in the endpoints.

(5) Affine (even projective) invariance φ affine yields:

$$\begin{tikzcd}
(\mathbf{b}_0^h, \dots, \mathbf{b}_n^h) \arrow[r, "\varphi"] \arrow[d] & \left(\varphi(\mathbf{b}_0^h), \dots, \varphi(\mathbf{b}_n^h)\right) \arrow[d] \\
R^h(u) = \sum_{i=0}^{n} B_i^n(u)\, \mathbf{b}_i^h \arrow[r, "\varphi"] & \varphi\left(\sum_{i=0}^{n} B_i^n(u)\, \mathbf{b}_i^h\right)
\end{tikzcd}$$

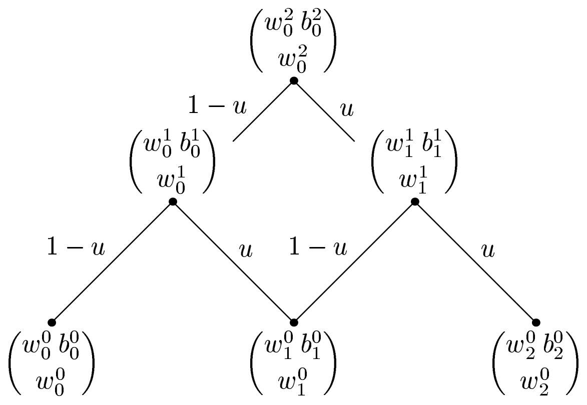
I. e. $\varphi(R(u)) = \frac{\sum_i B_i^n(u) \cdot w_i \cdot \varphi(\mathbf{b}_i)}{\sum_i B_i^n(u) \cdot w_i}$

(6) $F(u)$ shows the variation diminishing property with respect to $(\mathbf{b}_0, \dots, \mathbf{b}_n)$.

5.1.2 Evaluation of rational Béziercurves

The de Casteljau algorithm in homogeneous coordinates can be used to evaluate and subdivide the curve.
1st possibility:
Evaluate de Casteljau in the next higher dimension and project the result onto the $(w = 1)$-plane. The computation is done according to the scheme in Figure 5.3.

$$\begin{pmatrix} w_i^j \, \mathbf{b}_i^j \\ w_i^j \end{pmatrix} = (1-u) \begin{pmatrix} w_i^{j-1} \, \mathbf{b}_i^{j-1} \\ w_i^{j-1} \end{pmatrix} + u \begin{pmatrix} w_{i+1}^{j-1} \, \mathbf{b}_{i+1}^{j-1} \\ w_{i+1}^{j-1} \end{pmatrix} \tag{5.2}$$



$\begin{pmatrix} w_0^2 \, b_0^2 \\ w_0^2 \end{pmatrix}$
$1-u$
u
$\begin{pmatrix} w_0^1 \, b_0^1 \\ w_0^1 \end{pmatrix}$
$\begin{pmatrix} w_1^1 \, b_1^1 \\ w_1^1 \end{pmatrix}$
$1-u$
u
$1-u$
u
$\begin{pmatrix} w_0^0 \, b_0^0 \\ w_0^0 \end{pmatrix}$
$\begin{pmatrix} w_1^0 \, b_1^0 \\ w_1^0 \end{pmatrix}$
$\begin{pmatrix} w_2^0 \, b_2^0 \\ w_2^0 \end{pmatrix}$

Figure 5.3: de Casteljau scheme for rational Bézier curves

Problem: The Algorithm is numerically unstable if the weight differ very much.

2nd possibility: Rational de Casteljau algorithm

$$b_i^k(u) = (1-u) \cdot \frac{w_i^{k-1}}{w_i^k} \cdot b_i^{k-1}(u) + u \cdot \frac{w_{i+1}^{k-1}}{w_i^k} \cdot b_{i+1}^{k-1}(u) \tag{5.3}$$

with $\quad w_i^k = (1-u) \cdot w_i^{k-1} + u \cdot w_{i+1}^{k-1}\,, \quad$ for $\quad i = 0, \dots, n-k; \; k = 1, \dots, n$

5.1.3 Influence and properties of the weights

Theorem 5.1 (Effect and properties of the weights)

(1) For $w_0 = w_1 = \cdots = w_n = 1$ the rational Bézier curve is equal to a polynomial Bézier curve.

(2) Multiplication of all weights with a common factor does not change the curve.

(3) More general: If for all ratios

$$\frac{w_i}{w_{i+1}} : \frac{w_{i+2}}{w_{i+3}} = \text{const.}$$

then the curve has the same shape but another parametrization.

(4) Increasing the weight w_k pulls the curve to the control point $\mathbf{b}_k$. Decreasing the weight w_k pushes the curve away from the control point $\mathbf{b}_k$ (see Figure 5.4).

(5) In the limit case:

$$\lim_{w_k \to \infty} F(u) = \mathbf{b}_k. \tag{5.4}$$

Remark: From the properties (2) and (3) follows that we can assume without loss of generality that $w_0 = w_n = 1$ (proof given in the exercices).

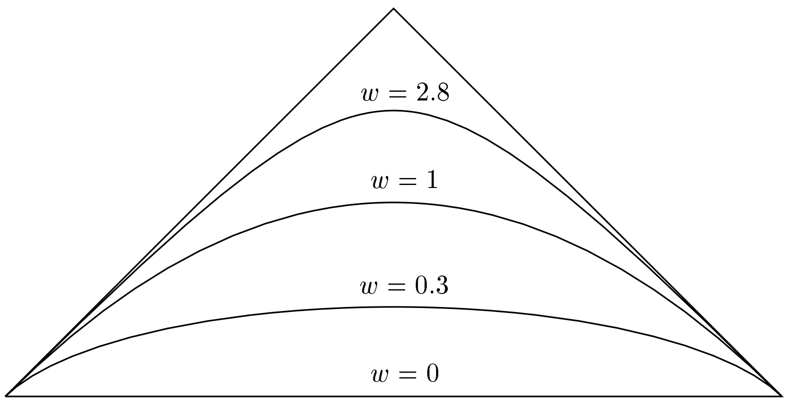


Figure 5.4: Effect of the weights ($n = 2$)

Supplement: Conic Sections

Definition 5.3 (Conic section)

– analytic: given a quadratic polynomial

$$G(x,y) = \underbrace{a\,x^2 + b\,x\,y + c\,y^2}_{\text{quadratisch}} + \underbrace{d\,x + e\,y}_{\text{linear}} + \underbrace{f}_{\text{konstant}}$$

then a conic section is defined as the point set

$$C_G(x,y) := \{(x,y)\,|\,G(x,y) = 0\}\,,$$

i. e. a conic section is the

– geometric: section of a plane with a double cone.

The coefficients a, b, c determine the type of the conic section.

$$a\,x^2 + b\,x\,y + c\,y^2 = (x,y)\begin{pmatrix} a & b/2 \\ b/2 & c \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}$$

The type of the conic section can be seen directly from the determinant of this matrix.

Graph	Angle between cone axis and plane	determinant of $\begin{pmatrix} a & b/2 \\ b/2 & c \end{pmatrix}$
ellipsis	greater than cone opening	greater than 0
parabola	equals cone opening	equals 0
hyperbola	less that cone opening	less than 0

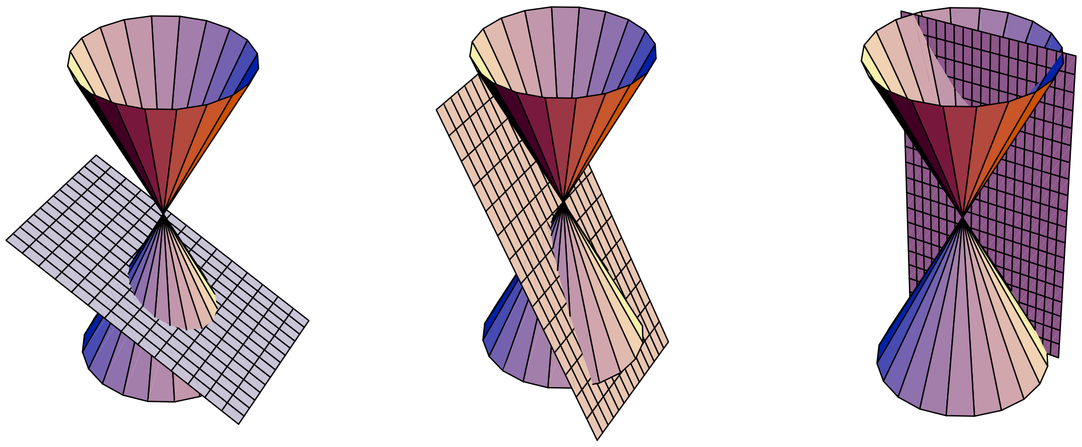


Figure 5.5: Conic sections: ellipsis, parabola und hyperbola

Example:

$G(x,y) = x^2 + y^2 - 1$	$C_G = \{(x,y) \,\vert\, x^2 + y^2 = 1\}$	unit circle
$G(x,y) = -x^2 + y$	$C_G = \{(x,y) \,\vert\, y = x^2\}$	standard parabola
$G(x,y) = x\,y - 1$	$C_G = \left\{(x,y) \,\vert\, y = \frac{1}{x}\right\}$	hyperbola

Conic sections can not be represented as polynomials but as rational functions.

Theorem 5.2

– Each rational quadratic Bézier curve is part of a conic section. For $w_0 = w_2 = 1$ and $w_1 > 0$ we have:

$$C_G : \left\{ \begin{array}{c} \text{ellipsis} \\ \text{parabola} \\ \text{hyperbola} \end{array} \right\} \Leftrightarrow \left\{ \begin{array}{l} 0 < w_1 < 1 \\ w_1 = 1 \\ w_1 > 1 \end{array} \right.$$

– Each part of a conic section can be represented as a rational quadratic Bézier curve.

5.1.4 Quadratic rational Bézier Curves as Conic Sections

Theorem 5.3

Given: a rational quadratic Bézier curve $F(u)$ over $[0,1]$ with

$$F(u) = \frac{(1-u)^2\,\mathbf{b}_0 + 2\,u\,(1-u)\,w_1\,\mathbf{b}_1 + u^2\,\mathbf{b}_2}{(1-u)^2 + 2\,u\,(1-u)\,w_1 + u^2}. \tag{5.5}$$

Choose coordinate system (ζ, η) such that

$$\mathbf{b}_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \ \mathbf{b}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \ \mathbf{b}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Then: $F(u)$ is part of a conic section with the defining equation

$$(1-\zeta-\eta)^2 = 4\,w_1^2 \cdot \zeta \cdot \eta \tag{5.6}$$

Proof: Let $\vec{\mathbf{e}}_1 := \mathbf{b}_0 - \mathbf{b}_1,\ \vec{\mathbf{e}}_2 := \mathbf{b}_2 - \mathbf{b}_1,\ N(u) := (1-u)^2 + 2\,u\,(1-u)\,w_1 + u^2$.

(1)
$$\begin{aligned} F(u) &= \frac{(1-u)^2}{N(u)}\,\mathbf{b}_0 + \frac{2\,u\,(1-u)\,w_1}{N(u)}\,\mathbf{b}_1 + \frac{u^2}{N(u)}\,\mathbf{b}_2 = \\ &= \frac{(1-u)^2}{N(u)}\,\mathbf{b}_0 + \frac{N(u)-(1-u)^2-u^2}{N(u)}\mathbf{b}_1 + \frac{u^2}{N(u)}\,\mathbf{b}_2 = \\ &= \frac{(1-u)^2}{N(u)}\,(\mathbf{b}_0-\mathbf{b}_1) + \frac{u^2}{N(u)}\,(\mathbf{b}_2-\mathbf{b}_1) + \mathbf{b}_1 = \frac{(1-u)^2}{N(u)}\,\vec{\mathbf{e}}_1 + \frac{u^2}{N(u)}\,\vec{\mathbf{e}}_2 + \mathbf{b}_1 \end{aligned}$$

Thus, in the coordinate system $\zeta,\ \eta$ we have

$$\zeta(u) = \frac{(1-u)^2}{N(u)} \qquad \eta(u) = \frac{u^2}{N(u)}$$

and additionally:

$$\frac{2\,u\,(1-u)\,w_1}{N(u)} = 1 - \zeta(u) - \eta(u)$$

$$\Rightarrow 4\,w_1^2\,\zeta\,\eta = \frac{4\,w_1^2\,(1-u)^2\,u^2}{N(u)^2} = \left(\frac{2\,w_1\,(1-u)\,u}{N(u)}\right)^2 = (1-\zeta-\eta)^2$$

(2) From equation 5.6 follows $\zeta^2 + \eta^2 + (2-4\,w_1^2)\zeta\eta - 2\,\zeta - 2\,\eta + 1 = 0$ and therefore

$$(\zeta,\eta)\begin{pmatrix} 1 & 1-2\,w_1^2 \\ 1-2\,w_1^2 & 1 \end{pmatrix}\begin{pmatrix}\zeta\\ \eta\end{pmatrix} + 1 = 0$$

$$\det\begin{pmatrix} 1 & 1-2\,w_1^2 \\ 1-2\,w_1^2 & 1 \end{pmatrix} = 4\,w_1^2\,(1-w_1)^2$$

$$\Rightarrow\ C_G : \left\{\begin{matrix}\text{ellipsis}\\ \text{parabola}\\ \text{hyperbola}\end{matrix}\right\} \quad\Leftrightarrow\quad \left\{\begin{matrix} 0 < |w_1| < 1 \\ |w_1| = 1 \\ |w_1| > 1\end{matrix}\right. \qquad \square$$

5.1.5 Segments of Conic Sections as quadratic rational Bézier Curves

Given:

- $G(x,y) = a\,x^2 + b\,x\,y + c\,y^2 + d\,x + e\,y + f$
- conic section $C_G = \{(x,y) \mid G(x,y) = 0\}$

– two points $\mathbf{b}_0 = (x_0, y_0),\ \mathbf{b}_2 = (x_2, y_2)$ on the same component of C_G (e. g. on the same branch of a hyperbola) with intersecting tangents.

Find: Representation of the corresponding segment of a conic section as a rational quadratic Bézier curve $(\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2;\ w_1;\ w_0 = w_2 = 1)$.

Procedure:

(1) Choose $\mathbf{b}_0, \mathbf{b}_2$ as given.

(2) Choose $\mathbf{b}_1$ as intersection point of the tangents T_0 and T_1:

$$T_i := \left(\frac{\partial G}{\partial y}, -\frac{\partial G}{\partial x}\right), \quad (x, y) = \mathbf{b}_i$$

Intersection point: solve $\mathbf{b}_0 + \alpha\, T_0 = \mathbf{b}_2 + \beta\, T_2$.

(3) Express the given polynomial $G(x, y)$ in the coordinate system (ζ, η) with

$\mathbf{b}_1 = (0, 0),\ \mathbf{b}_0 = (1, 0),\ \mathbf{b}_2 = (0, 1)$:

$$\begin{aligned} x &= (x_0 - x_1)\,\zeta + (x_2 - x_1)\,\eta + x_1 \\ y &= (y_0 - y_1)\,\zeta + (y_2 - y_1)\,\eta + y_1 \end{aligned}$$

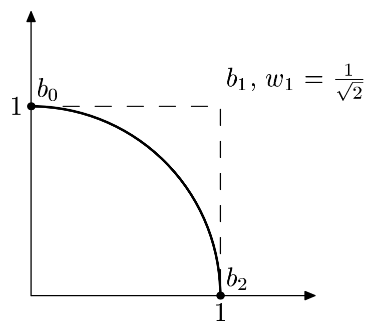
Inserting this into $G(x, y)$ gives a new polynomial (also of degree 2) with $\bar{G}(\zeta, \eta) = G(x, y)$. $\bar{G}$ describes the conic section in the coordinate system (ζ, η).

(4) Determine the weight w_1 using equation 5.6

$$\begin{aligned} \bar{G}(\zeta, \eta) &= \left(\frac{1}{4\,w_1^2}\,(1 - \zeta - \eta)^2 - \zeta\,\eta\right)\gamma = \\ &= \gamma\,\left(\frac{1}{4\,w_1^2}\,(\zeta^2 + \eta^2 - 2\,\zeta - 2\,\eta + 1) + \left(\frac{1}{2\,w_1^2} - 1\right)\zeta\,\eta\right) \end{aligned}$$

These are two equations for two variables $\Rightarrow$ comparison of the coefficients gives the solutions $w_1,\ \gamma$.

Example: Unit circle: $G(x, y) = x^2 + y^2 - 1,\ \mathbf{b}_0 = (1, 0),\ \mathbf{b}_2 = (0, 1),\ \frac{\partial G}{\partial x} = 2\,x,\ \frac{\partial G}{\partial y} = 2\,y$



(1) $\mathbf{b}_0 = (1, 0),\ \mathbf{b}_2 = (0, 1)$

(2) $T_0 = (2\,y, -2\,x)\,|_{(1,0)} = (0,-2); \quad T_2 = (2,0)$

$(1,0) + \alpha\,(0,-2) = (0,1) + \beta\,(2,0)$

$\Rightarrow\ \alpha = -\frac{1}{2},\ \beta = \frac{1}{2}\ \Rightarrow\ \mathbf{b}_1 = (1,1)$

(3) $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} \zeta \\ \eta \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

$\Rightarrow\ \bar{G}(\zeta,\eta) = (-\eta+1)^2 + (-\zeta+1)^2 - 1 = \zeta^2 + \eta^2 - 2\,\zeta - 2\,\eta + 1 + 1 - 1$

(4) Comparison of coefficients to determine w_1 and γ:

$1 = \frac{\gamma}{4\,w_1^2}; \quad 0 = \left(\frac{1}{2\,w_1^2} - 1\right)\gamma$

$w_1^2 = \frac{1}{2}\ \Rightarrow\ w_1 = \frac{1}{\sqrt{2}}$

Remark: For a circle arc of an arbitrary opening angle $\varphi < \pi$ we get $w_1 = \cos\frac{\varphi}{2}$.

5.1.6 Rational Bézier-Spline Curves

A rational Bézier-spline curve is a curve that is composed of single rational Bézier curves.
Example: Representation of the circle with four segments. Control points:

$$\begin{pmatrix}1\\0\end{pmatrix},\begin{pmatrix}1\\1\end{pmatrix},\begin{pmatrix}0\\1\end{pmatrix} \quad \begin{pmatrix}0\\1\end{pmatrix},\begin{pmatrix}-1\\1\end{pmatrix},\begin{pmatrix}-1\\0\end{pmatrix} \quad \begin{pmatrix}-1\\0\end{pmatrix},\begin{pmatrix}-1\\-1\end{pmatrix},\begin{pmatrix}0\\-1\end{pmatrix} \quad \begin{pmatrix}0\\-1\end{pmatrix},\begin{pmatrix}1\\-1\end{pmatrix},\begin{pmatrix}1\\0\end{pmatrix}$$

The weights are $w_0 = w_2 = 1$ and $w_1 = \frac{1}{\sqrt{2}}$ (see example in the previous section).
The corresponding 3D curve is the intersection curve of a cone (opened at top) with a pyramid (opened at bottom).

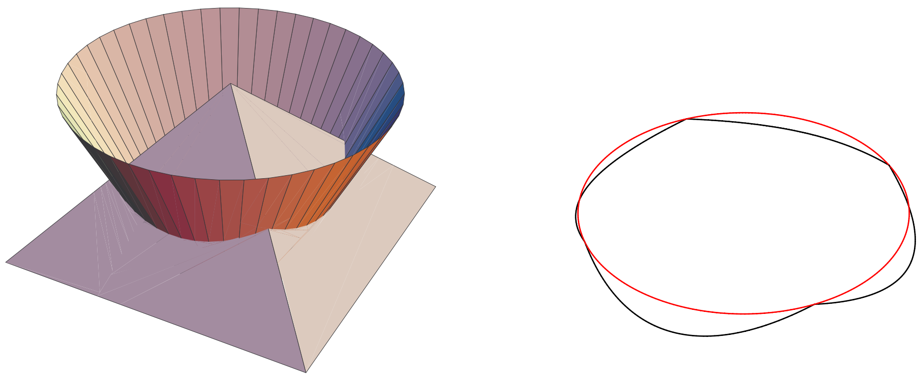


Figure 5.6: Intersection of a cone with a pyramid and projection of the intersection line

Remark: The higher dimensional curve is not smooth while the projected curve is smooth. This is quite typical for rational spline curves. By projection one can gain differentiability!

Theorem 5.4
Given: Rational Bézier curves $F : [0,1] \to \mathbb{R}^d$ and $G : [1,2] \to \mathbb{R}^d$ with control points $\mathbf{b}_0, \dots, \mathbf{b}_n$, $\mathbf{c}_0, \dots, \mathbf{c}_n$ and weights $w_0, \dots, w_n$, $v_0, \dots, v_n$. Additionally, set $w_0 = w_n = v_0 = v_n = 1$.
The curve composed from F and G is continuously differentiable $\Leftrightarrow$

$$\mathbf{b}_n = \mathbf{c}_0 = \frac{w_{n-1}}{w_{n-1} + v_1}\,\mathbf{b}_{n-1} + \frac{v_1}{w_{n-1} + v_1}\,\mathbf{c}_1. \tag{5.7}$$

Geometric interpretation: $\mathbf{b}_n = \mathbf{c}_0$ divides the line segment from $\mathbf{b}_{n-1}$ to $\mathbf{c}_1$ by the ratio $v_1 : w_{n-1}$.

Proof: We have $F'(1) = n\,w_{n-1}(\mathbf{b}_n - \mathbf{b}_{n-1})$ and $G'(1) = n\,v_1(\mathbf{c}_1 - \mathbf{c}_0)$. Because $\mathbf{c}_0 = \mathbf{b}_n$, we can summarize:

$$\mathbf{c}_0\,(v_1 + w_{n-1}) = v_1\,\mathbf{c}_1 + w_{n-1}\,\mathbf{b}_{n-1}.$$

□

5.2 Rational B-Spline Curves (NURBS)

	no local support	**local support**
rational	rational Bézier curves	NURBS
polynomial	Bézier curves	B-splines

Table 5.1: Summary: polynomial and rational curve schemes

“NURBS” = **n**on-**u**niform **r**ational **B-s**pline.

Definition 5.4 (Rational B-spline curve)
A *rational B-spline curve* in $\mathbb{R}^d$ is the projection of a B-spline curve in $\mathbb{R}^{d+1}$.

Definition 5.5 (Rational B-spline curve, analytic definition)
Given:

– degree n,

– knot vector T,

– control points $\mathbf{d}_0, \dots, \mathbf{d}_{m-1}$,

– weights $w_0, \dots, w_{m-1} \geq 0$ with $w_0 = w_{m-1} = 1$.

Then:

$$F(u) = \frac{\sum_{i=0}^{m-1} N_i^n(u)\, w_i\, \mathbf{d}_i}{\sum_{i=0}^{m-1} N_i^n(u)\, w_i} \tag{5.8}$$

is called *rational B-spline curve*.

Advantages:

- compared with rational Bézier-spline curves: C^{k-1}-continuity of rational B-spline curves of kth degree
- compared with B-spline curves: more degrees of freedom

Disadvantages: F is linearly dependent of the $\mathbf{d}_i$, but *not* of the w_i.
$\Rightarrow$ For tasks of approximation and interpolation NURBS are improper, because you can't build a linear system of equations.

6 Elementary Differential Geometry for Curves

We are not (really) interested in the parametric representation of curves but in the “shape” of a curve. What characterizes shape?

6.1 Reparametrization

Definition 6.1 (Regular curve)
Let $F : [a, b] \to \mathbb{R}^d$ a differentiable curve $(d = 2; 3)$. Then: F is called *regular* $\Leftrightarrow F'(u) \neq 0$ for all $u \in [a, b]$. $F'(u)$ is the (unique) *tangent direction* at the point $F(u)$.

Now we consider a *reparametrization* or *parameter transformation* of a regular curve:

$$\varphi : [s, t] \to [a, b] \quad \varphi(s) = a,\ \varphi(t) = b \qquad \Rightarrow\ \tilde{F}(x) = F(\varphi(x))$$

Definition 6.2 (Equivalence of curves)
Let $F : [a, b] \to \mathbb{R}^d$ and $\tilde{F} : [s, t] \to \mathbb{R}^d$ regular curves. Then: F and $\tilde{F}$ are called *equivalent* (notation: $F \sim \tilde{F}$) $\Leftrightarrow$ there exists a differentiable mapping $\varphi : [s, t] \to [a, b]$ with $\varphi(u) > 0$ for all $u \in [s, t]$ such that $\tilde{F}(u) = F(\varphi(u))$.

Example:

$$\begin{aligned}
&\tilde{F} : [0, \tfrac{1}{2}] \to \mathbb{R}^2 : && u \mapsto (4u, 2u) \\
&F : [0, 1] \to \mathbb{R}^2 : && v \mapsto (2\,v, v) \\
&\varphi : [0, \tfrac{1}{2}] \to [0, 1] : && u \mapsto 2u, \quad \varphi'(u) = 2 > 0.\ \Rightarrow \quad F \sim \tilde{F}
\end{aligned}$$

Example: Arc length

Let $F : [a, b] \to \mathbb{R}^d$ a regular curve. If we denote the length of the curve segment $\{F(u) : \alpha \leq u \leq \beta\}$ with ℓ_α^β, then:

$$\begin{aligned}
\ell_\alpha^\beta &= \lim_{|u_{i+1}-u_i| \to 0} \left\{ \sum_i \|F(u_{i+1}) - F(u_i)\| \right\} = \\
&= \lim_{|u_{i+1}-u_i| \to 0} \left\{ \sum_i \|F'(u_i)(u_{i+1} - u_i)\| \right\} = \int_\alpha^\beta \|F'(u)\|\, du
\end{aligned} \tag{6.1}$$

This leads to a nice parametrization (arc length parametrization). In this special case we have $\ell_\alpha^\beta = \beta - \alpha$ for all α, β and therefore

$$\int_\alpha^\beta \|F'(u)\|\, du = \beta - \alpha \ \Rightarrow\ \|F'(u)\| = 1 \quad \text{for all } u.$$

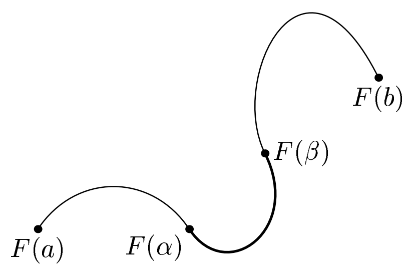


Figure 6.1: Arc length

Example: Circle: $F(u) = \begin{pmatrix} r\ \cos u \\ r\ \sin u \end{pmatrix}$, $u = 0, \dots, 2\pi$

Parameter transformation: $v := 2\pi r\, u$ yields $\tilde{F}(v) = \begin{pmatrix} r\ \cos \frac{2\pi v}{r} \\ r\ \sin \frac{2\pi v}{r} \end{pmatrix}$, $v = 0, \dots, 1.$

$\Rightarrow\ \|F'(u)\| = r, \quad \|\tilde{F}'(v)\| = 1.$

Theorem 6.1
Every regular curve can be reparametrized by arc length parametrization. This can be achieved by the following parameter transformation:

$$\text{Let } \Phi(u) := \int_a^u \|F'(s)\|\, ds, \text{ then } \tilde{F}(v) := F(\Phi^{-1}(v)) \tag{6.2}$$

is a reparametrization according to the arc length.

Example: In the previous example we had $\Phi(u) = r\,u$, thus $\tilde{F}(v) = F(\frac{r}{v}), \quad v = 0, \dots, 2\pi r.$

Remark: Bézier curves are rarely arc length parametrized.

6.2 Planar Curves

Definition 6.3 (2D Frenet frame)
Given: Let $F : [a, b] \to \mathbb{R}^2$ a regular parametric curve.

The local coordinate system $\{e_1(u), e_2(u)\}$ with

$$e_1(u) = \frac{F'(u)}{\|F'(u)\|} \tag{6.3}$$

$$e_2(u) = e_1(u) \text{ rotated by } 90^\circ \tag{6.4}$$

is called *2D Frenet frame*[1] of F.

[1]Jean Frédéric Frenet (1816-1900)

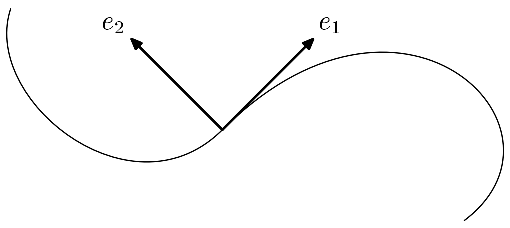


Figure 6.2: The 2D Frenet frame

Theorem 6.2 (Frenet)
For the derivatives of $e_1(u)$ and $e_2(u)$ the following holds:

$$e_1'(u) = w(s)\, e_2(u)$$
$$e_2'(u) = -w(s)\, e_1(u)$$

The quantity $\kappa(u) = \frac{w(u)}{||F'(u)||}$ is called *curvature* of the curve.
For arc length parametrized curves:

$$\kappa(v) = w(v) = \begin{cases} +||F''(v)|| & \text{curve bends to the left} \\ -||F''(v)|| & \text{curve bends to the right} \end{cases}$$

Sketch of Proof: The theorem states that e_1 and e_1' (respectively e_2 und e_2') are perpendicular; this is the case if and only if $||e_1(u)|| =$const., because from $0 = 2\,\langle e_1 \mid e_1' \rangle = \langle e_1 \mid e_1 \rangle'$ follows $||e_1(u)||' = 0$ and $||e_1(u)|| =$const. This is true by the normalization of e_1.

Remark:

– $\rho(u) = \frac{1}{||\kappa(u)||}$ is the radius of the osculating circle with the midpoint

$$M = F(s) + \frac{1}{||\kappa(u)||}\, e_2(u).$$

Theorem 6.3
For a curve point $F(u) = \begin{pmatrix} x(u) \\ y(u) \end{pmatrix}$ it holds true that

$$\kappa(u) = \frac{x'(u)\, y''(u) - x''(u)\, y'(u)}{\left(x'(u)^2 + y'(u)^2\right)^{3/2}} \tag{6.5}$$

Example: Ellipsis: $F(u) := \begin{pmatrix} x(u) \\ y(u) \end{pmatrix} = \begin{pmatrix} a\,\cos u \\ b\,\sin u \end{pmatrix}$

$$\begin{aligned} x' &= -a\,\sin u \quad & x'' &= -a\,\cos u \\ y' &= b\,\cos u \quad & y'' &= -b\,\sin u \end{aligned}$$

$$\Rightarrow \quad \kappa(u) = \frac{a\,b\,\sin^2 u + a\,b\,\cos^2 u}{(a^2\,\sin^2 u + b^2\,\cos^2 u)^{3/2}} = \frac{a\,b}{(a^2\,\sin^2 u + b^2\,\cos^2 u)^{3/2}}$$

Theorem 6.4
The curvature characterizes the curve uniquely short of rotation and translation. Additionally:

- $\kappa = 0 \;\Leftrightarrow\; F$ is a line segment.
- $\kappa = \kappa_0 =$ const. $\neq 0 \;\Leftrightarrow\; F$ is a circle arc (with radius $\frac{1}{\kappa_0}$).

Remark: Distinguish between *geometric* and *parametric* properties of a curve. *Geometric* means independent of the parametrization. Example:

geometric	parametric
unit tangent vector	tangent vector $F'(u)$
Frenet frame	curvature vector $F''(u)$
curvature κ	all derivatives

6.3 Space Curves

Definition 6.4 (3D Frenet frame)
Given: curve $F : [a,b] \to \mathbb{R}^3$. The local coordinate system $\{e_1(u), e_2(u), e_3(u)\}$ with

$$\begin{aligned} e_1(u) &= \frac{F'(u)}{\|F'(u)\|} && \text{unit tangent vector} \\ e_2(u) &= e_3(u) \times e_1(u) && \text{main normal vector} \\ e_3(u) &= \frac{F'(u) \times F''(u)}{\|F'(u) \times F''(u)\|} && \text{binormal vector} \end{aligned} \tag{6.6}$$

shows the following properties:

- $\{e_1(u), e_2(u), e_3(u)\}$ is orthonormal
- $\mathrm{span}\{e_1(u), e_2(u)\} = \mathrm{span}\{F'(u), F''(u)\}$ with the same orientation
- $\{e_1(u), e_2(u), e_3(u)\}$ is positively oriented (right handed system)

Theorem 6.5 (Frenet-Serret)
For the derivatives the following holds:

$$\begin{aligned} e_1'(u) &= && w_1(u)\,e_2(u) && \\ e_2'(u) &= -w_1(u)\,e_1(u) && && +w_2(u)\,e_3(u) \\ e_3'(u) &= && -w_2(u)\,e_2(u) && \end{aligned} \tag{6.7}$$

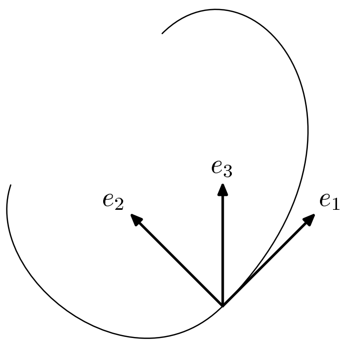


Figure 6.3: The 3D Frenet frame

In matrix notation:

$$\begin{pmatrix} e_1' \\ e_2' \\ e_3' \end{pmatrix} = \begin{pmatrix} 0 & w_1 & 0 \\ -w1 & 0 & w_2 \\ 0 & -w2 & 0 \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix}$$

Definition 6.5 (Curvature and torsion)

$$\text{curvature:} \quad \kappa(u) := \frac{w_1(u)}{||F'(u)||} \tag{6.8}$$

$$\text{torsion:} \quad \tau(u) := \frac{w_2(u)}{||F''(u)||} \tag{6.9}$$

Theorem 6.6
Formulas to compute κ and τ:

quantity	arc length parameter	arbitrary parameter
$\kappa(u)$	$\|\|F''(u)\|\|$	$\dfrac{\|\|F'(u) \times F''(u)\|\|}{\|\|F'(u)\|\|^3}$
$\tau(u)$	$\dfrac{\det(F'(u), F''(u), F'''(u))}{\|\|F''(u)\|\|^2}$	$\dfrac{\det(F'(u), F''(u), F'''(u))}{\|\|F'(u) \times F''(u)\|\|^2}$

(6.10)

Example: Spiral: $F(u) = \begin{pmatrix} x(u) \\ y(u) \\ z(u) \end{pmatrix} = \begin{pmatrix} r \cos u \\ r \sin u \\ Z\,t \end{pmatrix}$

Derivatives:

$$\begin{array}{lll} x' = -r \sin u & x'' = -r \cos u & x''' = r \sin u \\ y' = r \cos u & y'' = -r \sin u & y''' = -r \cos u \\ z' = Z & z'' = 0 & z''' = 0 \end{array}$$

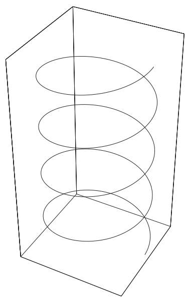


Figure 6.4: Spiral

$$e_1 = \tfrac{1}{\sqrt{r^2+Z^2}} \begin{pmatrix} -r\ \sin u \\ r\ \cos u \\ Z \end{pmatrix}, \quad e_3 = \tfrac{1}{\sqrt{r^2+Z^2}} \begin{pmatrix} Z\ \sin u \\ -Z\ \cos u \\ r \end{pmatrix}$$

$$e_2 = e_1 \times e_3 = \tfrac{1}{r^2+Z^2} \begin{pmatrix} -Z^2\ \cos u - r^2\ \cos u \\ -Z^2\ \sin u - r^2\ \sin u \\ 0 \end{pmatrix} = \begin{pmatrix} -\cos u \\ -\sin u \\ 0 \end{pmatrix}$$

$$\kappa(u) = \frac{r\sqrt{r^2+Z^2}}{\left(\sqrt{r^2+Z^2}\right)^3} = \frac{r}{r^2+Z^2}$$

$$\tau(u) = \frac{1}{r^2+Z^2} \begin{vmatrix} -r\ \cos u & -r\ \cos u & r\ \sin u \\ r\ cos u & -r\ \sin u & -r\ \cos u \\ Z & 0 & 0 \end{vmatrix} = \frac{r^2\, Z}{r^2\,(r^2+Z^2)} = \frac{Z}{r^2+Z^2}$$

Remark:

– The *osculating plane* is $\text{span}\{e_1, e_2\} = \text{span}\{F'(u), F''(u)\}$ (see osculating circle for planar curves).

– The *normal plane* is $\text{span}\{e_2, e_3\}$.

– We have assumed throughout that $\{F'(u), F''(u)\}$ are linearly independent! (i. e. $F''(u) \neq 0$, $\kappa > 0$)

Theorem 6.7 (Base theorem for space curves)
Curvature and torsion characterize a space curve uniquely short of rotation und translation. Additionally:

– $\tau = 0 \;\Leftrightarrow\; e_3(u) = 0 \;\Leftrightarrow\; F$ is a planar curve.

6.4 Geometric and Parametric Continuity

Remember: continuity of composed Bézier splines

Two polynomial curves $F : [a, b] \to \mathbb{R}^d$ and $G : [b, c] \to \mathbb{R}^d$ are called (parametric) C^k-continuous at b, if $F(b) = G(b),\ F'(b) = G'(b), \ldots, F^{(k)}(b) = G^{(k)}(b)$.

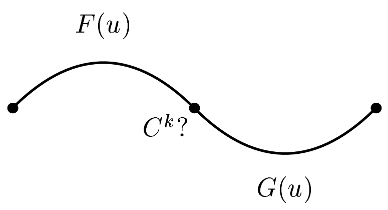


Figure 6.5: Continuity of Bézier splines

Definition 6.6 (Geometric continuity)
Two polynomial curves $F:[a,b]\to\mathbb{R}^d$ and $G:[b,c]\to\mathbb{R}^d$ are called (geometric) G^k-continuous at b, if the arc length parametrizations $\tilde{F}$ and $\tilde{G}$ are C^k-continuous.

Remark: C^k-continuity depends on parametrisierung ab, G^k-continuity does not!

Theorem 6.8 (Charakterization of geometric continuity)
The following statements are equivalent:

(1) F and G are G^k-continuous

(2) either F or G can be reparametrized such that F and $\tilde{G}$ (or $\tilde{F}$ and G) are C^k-continuous.

(3) There exist constants $\beta_1,\ldots,\beta_k$ ("Beta-constraints") such that

$$G^k(b)=\sum_{0\le j\le l}\ \sum_{\substack{i_1+\cdots+i_k=j\\ i_1+2i_2+\cdots+k\,i_k=k}}\frac{k!}{i_1!\,1!\,i_2!\,2!\cdots i_k!\,k!}\beta_1\cdots\beta_k\,F^{(j)}(b)\tag{6.11}$$

Proof: Let Φ and Ψ parameter transformations such that $\tilde{F}=F\circ\Phi$ und $\tilde{G}=G\circ\Psi$.
$(1)\ \Leftrightarrow\ (2)$: $\tilde{F}$ and $\tilde{G}$ are C^k-continuous $\Leftrightarrow$ F and $\tilde{G}\circ\Psi\circ\Phi^{-1}=G\circ(\Psi\circ\Phi^{-1})$ are C^k-continuous.

$(2)\ \Leftrightarrow\ (3)$: Assume: G and $F\circ\Phi$ are C^k-continuous at b.

$$k=1:\quad G'(b)=(F\circ\Phi)'(b)=F'(\Phi(b))\cdot\Phi'(b)=F'(\Phi(b))\cdot\beta_1$$

$$k=2:\quad G''(b)=(F\circ\Phi)''(b)=F''(\Phi(b))\cdot\Phi'(b)^2+F'(\Phi(b))\cdot\Phi''(b)=F''\cdot\beta_1^2+F'\cdot\beta_2$$

$$k=3:\quad G'''(b)=(F\circ\Phi)'''(b)=F'''\,\beta_1^3+F''\,3\beta_1\beta_2+F''\,\beta_1\beta_2+F'\,\beta_3$$

Note that $\beta_i=\Phi^{(i)}(b)$.

In matrix notation:

$$\begin{pmatrix}G'\\G''\\G'''\\\vdots\\G^{(k)}\end{pmatrix}=\begin{pmatrix}\beta_1&&&&\\\beta_2&\beta_1^2&&&\\\beta_3&3\beta_1\beta_2&\beta_1^3&&\\\vdots&&&\ddots&\\\beta_k&\cdots&\cdots&\cdots&\beta_1^k\end{pmatrix}\begin{pmatrix}F'\\F''\\F'''\\\vdots\\F^{(k)}\end{pmatrix}$$

(3) $\Leftrightarrow$ (2): Define Φ locally by

$$\Phi(x) = b + \beta_1\,(x-b) + \frac{\beta_2}{2!} + \cdots + \frac{\beta_k}{k!}(x-b)^k$$

□

Remark: Normally, of interest are only:

- G_1 : $G'(b) = \beta_1\,F'(b)$ für $\beta_1 > 0$
- G_2 : $G''(b) = \beta_1^2\,F''(b) + \beta_2\,F'(b)$
- G_3 : $G'''(b) = \beta_1^3\,F'''(b) + 3\beta_1\beta_2\,F''(b) + \beta_3\,F'(b)$

6.4.1 Geometric Continuity of Bézier Curves

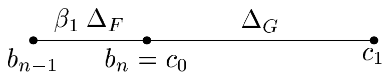
Given: Bézier curves $F : [a,b] \to \mathbb{R}^d$ and $G : [b,c] \to \mathbb{R}^d$ with $F(u) = \sum_{i=0}^n B_i^n(u)\,\mathbf{b}_i$ and $G(u) = \sum_{i=0}^n B_i^n(u)\,\mathbf{c}_i$ and the interval lengths $\Delta_F = b-a$ and $\Delta_G = c-b$.

Theorem 6.9 (Characterization of G^1-continuity (tangent continuity))
Equivalent are:

(1) F and G are G^1-continuous

(2) F and G have the same tangent direction (i. e. the same tangent unit vector) at b

(3) divide ratio:



Sketch of Proof: (1) $\Leftrightarrow$ (2): follows from $G'(b) = \beta_1\,F'(b)$
(1) $\Leftrightarrow$ (3): use equation for β-constraints and $F'(b) = \frac{n}{\Delta_F}\,(\mathbf{b}_n - \mathbf{b}_{n-1}),\ G'(b) = \frac{n}{\Delta_G}\,(\mathbf{c}_1 - \mathbf{c}_0)$

Theorem 6.10 (Characterization of G^2-continuity (curvature continuity))
Equivalent are:

(1) F and G are G^2-continuous

(2) F and G have the same unit tangent vector and the same curvature vector at b ($e_1^F = e_1^G,\ e_2^F = e_2^G,\ \rho^F = \rho^G$)

(3) There exist constants γ_1, γ_2 with $\gamma_1, \gamma_2 > 0$ such that the generalized A-frame property holds.

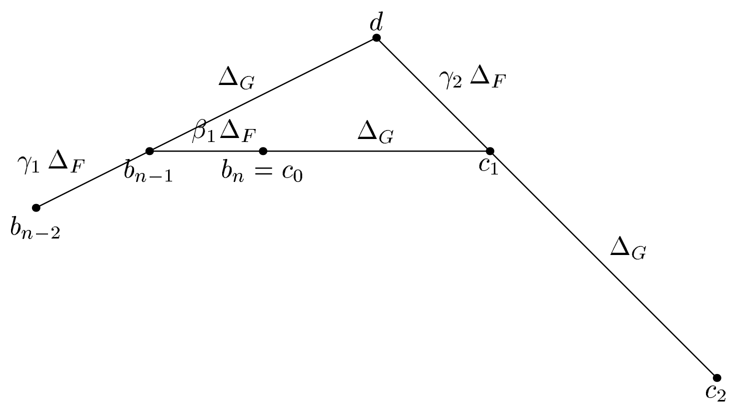


Figure 6.6: Generalized A-frame property

Sketch of Proof: (1) $\Leftrightarrow$ (3): Use β-constraints:

$F'(b) = \beta_1\,G'(b)$ and

$F''(b) = \beta_1^2\,G''(b) + \beta_2\,G'(b)$ and for the control points

$\frac{n}{\Delta_F}\,(\mathbf{b}_n - \mathbf{b}_{n-1}) = \frac{\beta_1\,n}{\Delta_G}\,(\mathbf{c}_1 - \mathbf{c}_0)$

$\frac{n(n-1)}{\Delta_F^2}\,(\mathbf{b} - 2\mathbf{b}_{n-1} + \mathbf{b}_{n-2}) = \frac{\beta_1^2\,n\,(n-1)}{\Delta_G^2}(\mathbf{c}_2 - 2\mathbf{c}_1 + \mathbf{c}_0) + \frac{\beta_2\,b}{\Delta_G}\,(\mathbf{c}_1 - \mathbf{c}_0)$

generalized A-frame condition:

$\mathbf{b}_n = \mathbf{c}_0$

$\mathbf{c}_1 = \mathbf{c}_0 + \frac{\Delta_G}{\sqrt{\gamma_1\gamma_2}\,\Delta_F}\,(\mathbf{b}_n - \mathbf{b}_{n-1})$

$\mathbf{b}_{n-1} + \frac{\Delta_G}{\gamma_1\,\Delta_F}\,(\mathbf{b}_{n-1} - \mathbf{b}_{n-2}) = \mathbf{c}_1 - \frac{\gamma_2\,\Delta_F}{\Delta_G}\,(\mathbf{c}_2 - \mathbf{c}_1)$

$\Rightarrow$

$$\gamma_1 = \frac{2\,(1+\beta_1)\,\beta_1^2}{\beta_2 + 2\,\beta_1 + 2\,\beta_1^2}$$

$$\gamma_2 = \frac{\beta_2 + 2\,\beta_1 + 2\,\beta_1^2}{2\,(1+\beta_1)}$$

respectively

$$\beta_1 = \sqrt{\gamma_1\gamma_2}$$

$$\beta_2 = 2\,(\gamma_2 - 1)\,\sqrt{\gamma_1\gamma_2} - 2\,\gamma_1\gamma_2 + 2\gamma_2$$

□

7 Tensor Product Surfaces

7.1 General Approach

Given: two curve schemes in 3D space:

$$F(u) = \sum_i N_i(u)\,\mathbf{b}_i \quad \text{and} \quad G(u) = \sum_i M_i(u)\,\mathbf{d}_i$$

with scalar basis functions $N_i(u)$ and $M_i(u)$.

The tensor product of these schemes is:

$$S(u,v) := \sum_i \sum_j N_i(u)\,M_i(v)\,\mathbf{c}_{i,j} \tag{7.1}$$

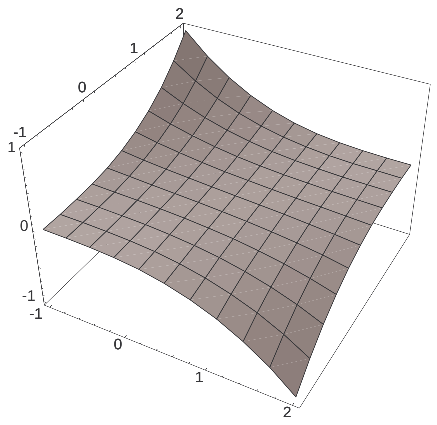


Figure 7.1: A tensor product Bézier surface

Of special interest are:

Definition 7.1 (Tensor Product Bézier Surface)

$$S(u,v) := \sum_{i=0}^{n} \sum_{j=0}^{m} B_i^n(u)\,B_j^m(v)\,\mathbf{b}_{i,j} \tag{7.2}$$

Definition 7.2 (Tensor Product B-Spline Surface)

$$F(u,v) := \sum_{i=0}^{N} \sum_{j=0}^{M} N_i^{n,T}(u)\,N_j^{m,S}(v)\,\mathbf{d}_{i,j} \tag{7.3}$$

with knot vectors $T = \{t_0, \ldots, t_{N+m}\}$ and $S = \{s_0, \ldots, s_{M+m}\}$.

Remark: We can factorize:

$$S(u,v) = \sum_i \sum_j N_i(u)\, M_j(v)\, \mathbf{c}_{i,j} = \sum_i N_i(u) \left(\sum_j M_j(v)\, \mathbf{c}_{i,j} \right) = \sum_j M_j(v) \left(\sum_i N_i(u)\, \mathbf{c}_{i,j} \right)$$

Each fixed u gives a control point of the v curve

$\Rightarrow$ The tensor product surface is a "curve of curves"!

With the same approach we get rational tensor product Bézier and B-spline surfaces: consider a TP Bézier or a TP B-spline surface in 4D space (homogeneous coordinates) and divide by the w component. This gives:

$$\mathbf{c}_{i,j} = \left(\frac{x_{i,j}}{w_{i,j}}, \frac{y_{i,j}}{w_{i,j}}, \frac{z_{i,j}}{w_{i,j}} \right)$$

Definition 7.3
The array of control points $\mathbf{b}_{i,j}$ of a tensor product surface is called the *control net* of the surface (analogon of control polygon for curves).

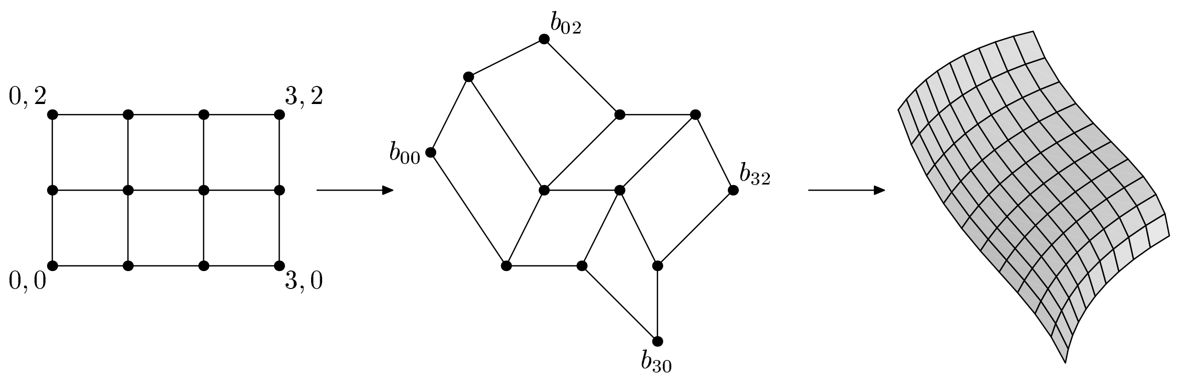


Figure 7.2: Parameter grid, control net and resulting surface

7.2 Tensor Product Bézier Surfaces

Given: a tensor product (TP) Bézier surface

$$S(u,v) := \sum_{i=0}^{n} \sum_{j=0}^{m} B_i^n(u)\, B_j^m(v)\, \mathbf{b}_{i,j}$$

Theorem 7.1 (Shape properties of TP Bézier surfaces)

(a) Convex hull property: $S(u,v) \in$ convex hull $\{\mathbf{b}_{i,j} \,|\, 0 \le i \le n, 0 \le j \le m\}$ for $u, v \in [0,1]$.

(b) Interpolation of the four corners of the control net:
$S(0,0) = \mathbf{b}_{00},\ S(0,1) = \mathbf{b}_{0m},\ S(1,0) = \mathbf{b}_{n0},\ S(1,1) = \mathbf{b}_{nm}.$

(c) Tangency in the corner points, e. g. the tangent plane at $S(0,0) = \mathbf{b}_{00}$ is spanned by $\mathbf{b}_{10} - \mathbf{b}_{00}$ and $\mathbf{b}_{01} - \mathbf{b}_{00}$.

(d) The boundary curves are Bézier curves, their control points are the boundary points of the control net (e. g. $S(0,v) = F(v)$ is Bézier curve).

(e) Affince invariance

(f) Variation diminishing property does *not* hold!

Sketch of Proof:

– affine invariance: $S(u,v) := \sum_{i=0}^{n} \sum_{j=0}^{m} B_i^n(u)\, B_j^m(v)\, \mathbf{b}_{i,j}$ must be a barycentric (affine) combination:

$$\sum_{j=0}^{m} \left(\sum_{i=0}^{n} B_i^n(u) \right) B_j^m(v) = \sum_{j=0}^{m} 1 \cdot B_j^m(v) = 1$$

– convex hull: $0 \le u, v \le 1$ and $B_i^n(u)\, B_j^m(v) \ge 0 \;\Rightarrow S(u,v)$ is a convex combination.

□

7.3 Algorithms for TP Bézier Surfaces

7.3.1 Evaluation with the TP Approach

Given:

– TP Bézier surface $S(u,v) = \sum_{j=0}^{m} B_j^m(v)\; (\sum_{i=0}^{n} B_i^n(u)\, \mathbf{b}_{i,j}) =: \sum_{j=0}^{m} B_j^m(v)\, \mathbf{b}_j(u)$

– parameter values u, v

Proceeding:

(1) Apply the univariate de Casteljau algorithm in u-direction $(n+1)$ times:

$$c_j = \sum_{i=0}^{n} B_i^n(u) b_{i,j}$$

(2) Apply the univariate de Casteljau algorithm in v-direction once:

$$S(u,v) = \sum_{j=0}^{m} B_j^m(v) c_j$$

or first in v-direction, afterwards in u-direction. The evaluation with the TP approach first in u-direction, then in v-direction is depicted in Figure 7.3.

Choice of the best order of evaluation (smallest number of needed arithmetic operations):

– If $m > n$: first in u-, afterwards in v-Richtung

– If $m < n$: first in v-, afterwards in u-Richtung

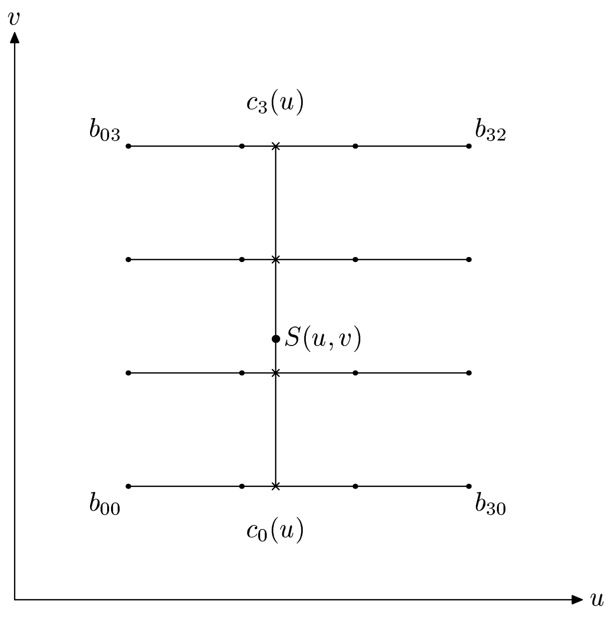


Figure 7.3: de Casteljau scheme for tensor product Bézier surfaces

7.3.2 The "direct" de Casteljau Algorithm

Idea: use bilinear interpolation instead of linear interpolation.

Theorem 7.2 (Bilinear interpolation formula)

$$
\begin{aligned}
S(u,v) &= (1-u)(1-v)\,\mathbf{b}_{00} + u\,(1-v)\,\mathbf{b}_{10} + (1-u)\,v\,\mathbf{b}_{01} + u\,v\,\mathbf{b}_{11} = \\
&= (\,(1-u), u\,) \begin{pmatrix} \mathbf{b}_{00} & \mathbf{b}_{01} \\ \mathbf{b}_{10} & \mathbf{b}_{11} \end{pmatrix} \begin{pmatrix} 1-v \\ v \end{pmatrix} \qquad (7.4)
\end{aligned}
$$

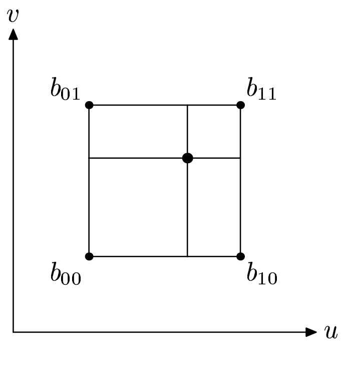


Figure 7.4: Bilinear interpolation

The bilinear interpolation formula (see Figure 7.4) is applied to the whole control net in the direct de Casteljau Algorithm (see Figure7.5). Therefore the evaluation scheme looks like a three-dimensional pyramid.

Algorithm 7.1 (Direct de Casteljau)

$$\mathbf{b}_{i,j}^{0,0} = \mathbf{b}_{i,j}$$

$$\mathbf{b}_{i,j}^{r,r} = (\,(1-u), u\,) \begin{pmatrix} \mathbf{b}_{i,j}^{r-1,r-1} & \mathbf{b}_{i,j+1}^{r-1,r-1} \\ \mathbf{b}_{i+1,j}^{r-1,r-1} & \mathbf{b}_{i+1,j+1}^{r-1,r-1} \end{pmatrix} \begin{pmatrix} 1-v \\ v \end{pmatrix} \tag{7.5}$$

$$r = 1, \ldots, n;\; i,j = 0, \ldots, n-r$$

Then: $\mathbf{b}_{0,0}^{n,n} = S(u,v)$.

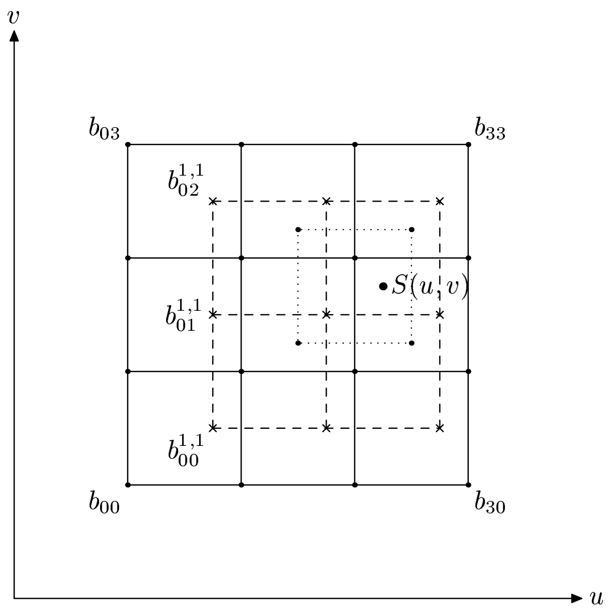


Figure 7.5: Direct de Casteljau scheme

Problem: If $n \neq m$ you have to use the univariate de Casteljau algorithm on from a certain step. This makes the implementation considerably more difficult.

Comparison of direct de Casteljau and TP approach according to the number of convex combinations (CC):

	TP approach	**direct de Castejau**
$n = m = 3$	$4 * 6 + 6 = 30$ CC	$14 * 3 = 42$ CC
$n = m = 5$	$6 * 15 + 15 = 105$ CC	$55 * 3 = 165$ CC

In practice the TP approach is used, because it is more efficient and you don't have to do any distinction of cases.

Remark: Other algorithms like degree elevation or subdivision can be derived from the corresponding algorithms for curves.

7.4 Tensor Product B-Spline Surfaces

Given:

- u knot vector $T = \{t_0 \le t_1 \le \cdots \le t_{N+n}\}_{t_i \le t_{i+n}}$
- v knot vector $S = \{s_0 \le s_1 \le \cdots \le s_{M+m}\}_{s_j \le s_{j+m}}$
- array of control points $\{\mathbf{d}_{i,j} \,|\, 0 \le i \le N-1,\ 0 \le j \le M-1\}$
  $\Rightarrow (u,v) \in [t_n, t_N] \times [s_m, s_M]$
- surface

$$F(u,v) := \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} N_i^{n,T}(u)\, N_j^{m,S}(v)\, \mathbf{d}_{i,j}$$

Theorem 7.3 (Shape properties of TP B-spline surfaces)

(a) Local convex hull property:

$(u,v) \in [t_{i+n}, t_{i+n+1}] \times [s_{j+m}, s_{j+m+1}] \quad \Rightarrow \quad F(u,v) \in$ convex hull$\{\mathbf{d}_{l,k} \,|\, i \le l \le i+n,\ j \le k \le j+m\}$

(b) Affine invariance

(c) Endpoint interpolation: if $t_0 = t_1 = \cdots = t_n < \cdots < t_N = \cdots = t_{N+n}$ and $s_0 = s_1 = \cdots = s_m < \cdots < s_M = \cdots = t_{M+m}$ then

$F(t_n, s_m) = \mathbf{d}_{0,0},\ F(t_N, s_m) = \mathbf{d}_{N-1,0},\ F(t_n, s_M) = \mathbf{d}_{0,M-1},\ F(t_N, s_M) = \mathbf{d}_{N-1,M-1}.$

In this case: tangency in the corner points, and the boundary curves are B-spline curves with the boundary points of the control net as curve control points. The tangential plane at the corners is spanned by the control points, e. g. $T_F(t_n, s_m) = d_{00} + \lambda(d_{10} - d_{00}) + \mu(d_{01} - d_{00})$.

(d) Local control: “a control point has influence” only on a part of the surface.

Changing $\mathbf{d}_{i,j}$ will change $F(u,v)$ only for $(u,v) \in [t_i, t_{i+n+1}] \times [s_j, s_{j+m+1}]$.

(e) Variation diminishing property does *not* hold!

Remark: Remember from curves: a Bézier curve is a special case of a B-spline curve. A Bézier curve over the interval $[\alpha, \beta]$ with the control points $\{\mathbf{b}_0, \ldots, \mathbf{b}_n\}$ is a B-spline curve over the knot vector $\{\underbrace{\alpha, \ldots, \alpha}_{n+1}, \underbrace{\beta, \ldots, \beta}_{n+1}\}$ with the same control points. The same property holds true for tensor product surfaces.

7.5 Algorithms for TP B-Spline Surfaces

7.5.1 Evaluation of TP B-Spline Surfaces

Given: $F(u,v) := \sum_{i=0}^{N-1} N_i^{n,T}(u) \sum_{j=0}^{M-1} N_j^{m,S}(v)\, \mathbf{d}_{i,j}$ with knot vectors S and T.
The proceeding is similar to the TP approach of TP Béziersurfaces:

(1) Evaluate N B-Spline curves of degree m with de Boor in v-direction.

(2) Evaluate one B-Spline curve of degree n with de Boor in u-direction.

7.5.2 Knot insertion

The Boehm knot insertion algorithm for increasing modelling parameters (i. e. control points) in a local area will produce overhead, because the refinement of the control net can only be done by adding new rows and columns.

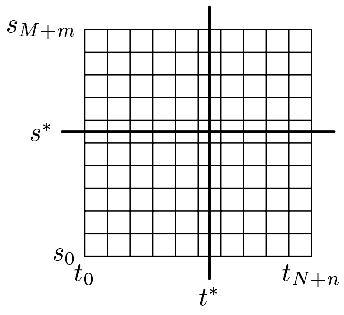


Figure 7.6: Knot insertion with the Boehm algorithm

One alternative are *hierarchical B-splines* (by Forsey and Bartels, Siggraph88). In this solution, a B-spline surface $\tilde{F}(u,v)$ is composed of a patch $F(u,v)$ over a coarse parameter grid and, local detail $G(u,v)$ defined over a locally refined parameter grid. You can see in Figure 7.7 that the grid is refined only where needed.

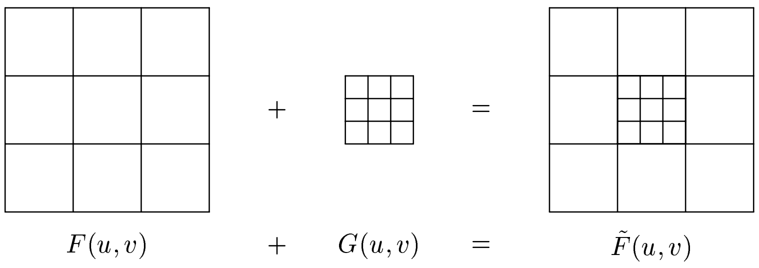


Figure 7.7: Building a hierarchical knot grid for B-spline patches

7.6 Interpolation and Approximation

7.6.1 General interpolation

Be $F(u,v) = \sum_i^N \sum_j^M A_i^n(u) B_j^m(u) \mathbf{c}_{ij}$. Consider $A_i(u)B_j(u)$ as basis functions $C_{ij}(u,v)$:

$$F(u,v) = \sum_{(i,j)}^{(N-1,M-1)} C_{ij}(u,v)\mathbf{c}_{ij}$$

Task: Interpolation/approximation of data points $\mathbf{d}_k$ with parameter values (u_k, v_k):

$$F(u_k,v_k) = \sum_{(i,j)=(0,0)}^{(N,M)} C_{ij}(u_k,v_k) c_{ij} \overset{!}{=} \mathbf{d}_k \ , \quad k = 1, \dots, ((N+1)*(M+1))$$

Interpolation and approximation are exactly the same as for Bézier and B-Spline curves.
Problem: For curves we have the Theorem of Schoenberg-Whitney, that says if the interpolation/approximation matrix is invertable. But there is no comparable theorem for surfaces.

7.6.2 Special case: Interpolation at the knots

Interpolation over grids:
Given: data points d_{ij} with parameter values (u_i, v_j) for $i = 0, \dots, N$, $j = 0, \dots, M$
Find: surface $F(u,v) = \sum_{i=0}^N \sum_{j=0}^M A_i^n(u) B_j^m(v) c_{ij}$ with $F(u_k, v_l) = \mathbf{d}_{kl}$
$k = 0, \dots, N$, $\quad l = 0, \dots, M$

$$\underbrace{\begin{pmatrix} A_0(u_0) & \dots & A_N(u_0) \\ \vdots & & \vdots \\ A_0(u_N) & \dots & A_N(u_N) \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} \mathbf{c}_{00} & \dots & \mathbf{c}_{0M} \\ \vdots & & \vdots \\ \mathbf{c}_{N0} & \dots & \mathbf{c}_{NM} \end{pmatrix}}_{C} \underbrace{\begin{pmatrix} B_0(v_0) & \dots & B_0(v_M) \\ \vdots & & \vdots \\ B_M(v_0) & \dots & B_M(v_M) \end{pmatrix}}_{B^T} =$$

$$= \underbrace{\begin{pmatrix} \mathbf{d}_{00} & \dots & \mathbf{d}_{0M} \\ \vdots & & \vdots \\ \mathbf{d}_{N0} & \dots & \mathbf{d}_{NM} \end{pmatrix}}_{D}$$

The resulting matrix system is very large. For $N = M = 30$ for example you have to do $900 * 900$ operations. Therefore the following proceeding is better:

(1) Solve $A * E = D$

(2) Solve $E = C * B^T$

Tasks (1) and (2) are interpolation problems of curves, for whom the corresponding algorithms can be applied.

Interpolation at the knots:
If $(u_k, v_l) = (t_i, s_j)$, then tasks (1) and (2) are tridiagonal matrices and therefore solvable in a fast way.

7.7 Trimming of parametric Patches

One disadvantage of tensor product Bézier and B-spline surfaces is that all patches are square-like; it is not possible to model surfaces that does not have a well-defined tangent plane (i. e. if $F'(u) = 0$).

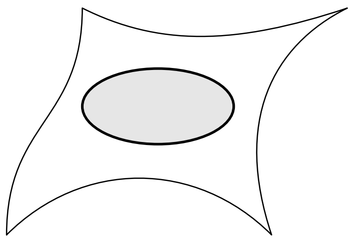


Figure 7.8: Trimming a patch

A solution are *trimmed patches*. Specify a set of closed contours $\Gamma_0, \dots, \Gamma_r$ in the parameter domain, where Γ_0 is the outer contour and $\Gamma_1, \dots, \Gamma_r$ are inner contours (see Figure 7.9). By definition: $F(u, v)$ shall belong to the trimmed surface if

– (u, v) is *inside* with respect to Γ_0 **and**

– (u, v) is *outside* with respect to $\Gamma_1, \dots, \Gamma_r$.

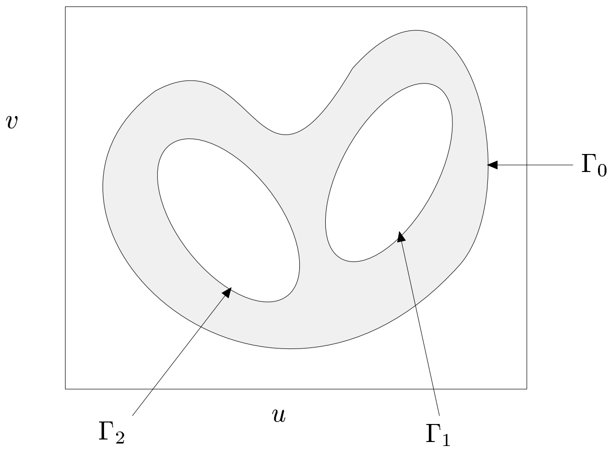


Figure 7.9: Specifying inner and outer contours in the parameter domain

How to determine if a point $\mathbf{p}$ is inside or outside the trimmed surface (see Figure 7.10)?

Theorem 7.4
Given: a closed contour $\Gamma : [a, b] \to \mathbb{R}^2$ with $\Gamma(a) = \Gamma(b)$:

– $\mathbf{p}$ is inside Γ :$\Leftrightarrow$ a ray emanating from $\mathbf{p}$ intersects the contour k times with k being *odd*.

– $\mathbf{p}$ is outside Γ :$\Leftrightarrow$ a ray emanating from $\mathbf{p}$ intersects the contour k times with k being *even*.

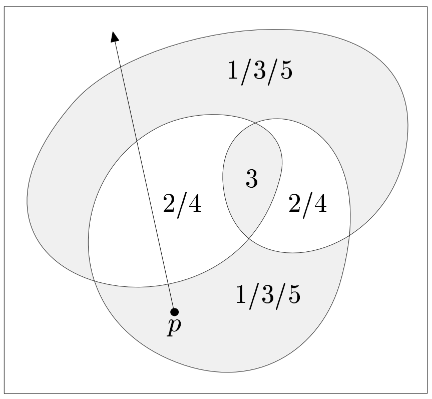


Figure 7.10: Determining inside and outside of a trimmed patch

8 Ray Tracing of TP-Bézier Surfaces (Bézier Clipping)

In this chapter, we want to discuss an algorithm to compute intersections between rational tensor product Bézier surfaces and rays. In particular, this is important for rendering Bézier patches using the ray tracing approach.

For more information see Nishita et al., Siggraph90.

Given:

– rational TP Bézier patch

$$F(u,v) := \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} B_i(u)\,B_j(v)\,w_{ij}\,\mathbf{p}_{ij}}{\sum_{i=0}^{n}\sum_{j=0}^{m} B_i(u)\,B_j(v)\,w_{ij}} \tag{8.1}$$

– ray $Q(t) = \mathbf{p}_0 + t\,\vec{\mathbf{q}}$

Find: intersection, i. e. find t and (u,v) such that $Q(t) = F(u,v)$.
Remark: This problem is analytically solvable only for low polynomial degrees. Therefore an iterative approximation algorithm as presented here is normally a better approach.

1st step:

Represent the ray as an intersection of two planes:

$$a_1\,x + b_1\,y + c_1\,z + d_1 = 0$$
$$a_2\,x + b_2\,y + c_2\,z + d_2 = 0$$

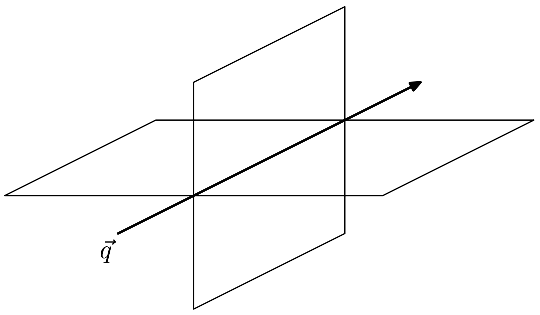


Figure 8.1: Bézier clipping, step 1

Example: Select plane normal vectors that are perpendicular to $\vec{\mathbf{r}} = (r_x, r_y, r_z)$, e. g. $(a_1, b_1, c_1) = (-r_y, r_x, 0)$ and $(a_2, b_2, c_2) = (0, -r_z, r_y)$. Then determine d_1 and d_2 by applying $\mathbf{p}_0$:

$$d_1 = -(a_1\,p_{0_x} + b_1\,p_{0_y} + c_1\,p_{0_z})$$
$$d_2 = -(a_2\,p_{0_x} + b_2\,p_{0_y} + c_2\,p_{0_z})$$

2nd step:

Find intersection of F with the two planes:

$$\left\langle \begin{pmatrix} a_k \\ b_k \\ c_k \end{pmatrix} \middle| \, F(u,v) \right\rangle + d_k = 0\,, \quad k = 1;2$$

$$\Leftrightarrow \left\langle \begin{pmatrix} a_k \\ b_k \\ c_k \end{pmatrix} \middle| \, \sum_{i=0}^{n}\sum_{j=0}^{m} B_i(u)\,B_j(v)\,w_{ij}\,\mathbf{p}_{ij} \right\rangle + d_k \sum_{i=0}^{n}\sum_{j=0}^{m} B_i(u)\,B_j(v)\,w_{ij} = 0$$

We get

$$R(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} r_{ij}\,B_i(u)\,B_j(v) = 0 \quad \text{and}$$

$$S(u,v) = \sum_{i=0}^{n}\sum_{j=0}^{m} s_{ij}\,B_i(u)\,B_j(v) = 0$$

with

$$r_{ij} := w_{ij} \left\langle \begin{pmatrix} a_1 \\ b_1 \\ c_1 \end{pmatrix} \middle| \, \mathbf{p}_{ij} \right\rangle + d_1\,w_{ij} \quad \text{and}$$

$$s_{ij} := w_{ij} \left\langle \begin{pmatrix} a_2 \\ b_2 \\ c_2 \end{pmatrix} \middle| \, \mathbf{p}_{ij} \right\rangle + d_2\,w_{ij}.$$

Remark: This is a projection of $F(u,v)$ along $Q(t)$ onto a plane that is perpendicular to $Q(t)$.

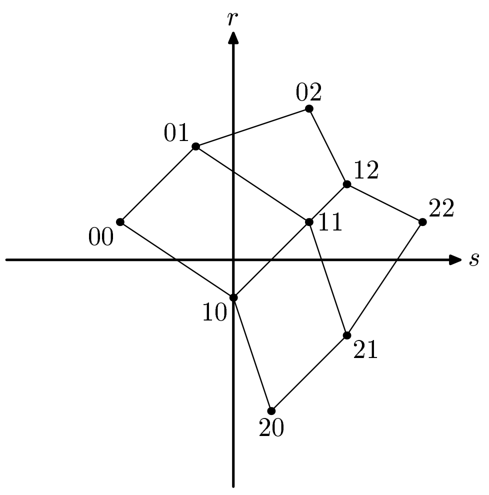


Figure 8.2: Bézier clipping, step 2

3rd step:

Choose two directions in (r, s) space:

L_u direction $\approx$ perpendicular to u direction, $u =$ const.
L_v direction $\approx$ perpendicular to v direction, $v =$ const.

In case of a biquadratic patch:

$$L_u = \frac{1}{2}\left[\begin{pmatrix} r_{20} - r_{00} \\ s_{20} - s_{00} \end{pmatrix} + \begin{pmatrix} r_{22} - r_{02} \\ s_{22} - s_{02} \end{pmatrix}\right]$$

$$L_v = \frac{1}{2}\left[\begin{pmatrix} r_{02} - r_{00} \\ s_{02} - s_{00} \end{pmatrix} + \begin{pmatrix} r_{22} - r_{20} \\ s_{22} - s_{20} \end{pmatrix}\right]$$

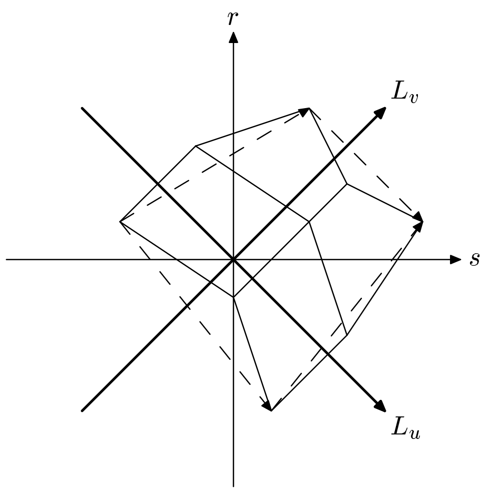


Figure 8.3: Bézier clipping, step 3

Alternate the directions L_u and L_v and iterate on the following steps:

4th step:

Determine the (signed) distances d_{ij} of the projected control points (r_{ij}, s_{ij}) to L_u. This gives a Bézier representation of the "distance-to-L_u"-function that can be used to determine the distance of an arbitrary point to L_u:

$$D(u, v) = \sum_{i=0}^{n}\sum_{j=0}^{m} d_{ij}\, B_i(u)\, B_j(v)$$

5th step:

Plot the values $\left(\frac{i}{n},\, d_{ij}\right)$ in a u-d-diagram (see figure 8.4)

6th step:

Determine the convex hull of the points in this diagram (by connecting maxima and minima of the columns) and intersect it with the u-axis. This gives an interval $[u_{\min},\, u_{\max}]$.

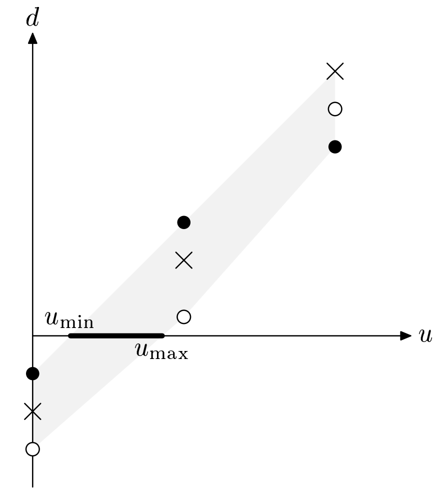


Figure 8.4: Bézier clipping, steps 5 and 6

7th step:

Clip away the parametric domain $u < u_{\min}$ and $u > u_{\max}$ by two subdivision steps (using the de Casteljau algorithm for Bézier patches).

Switch direction $(L_u \to L_v \to L_u \to \cdots)$ and repeat steps **4** to **7**. The parameter values of the intersection point must be within the box $[u_{\min},\, u_{\max}] \times [v_{\min},\, v_{\max}]$.

Stop when the size of the box is smaller than a threshold value or the convex hull in step **6** does not intersect with the axis; this indicates that there is no intersection of Q and F.

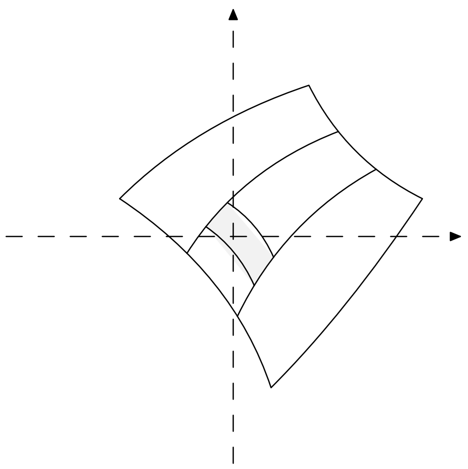


Figure 8.5: Bézier clipping, step 7

9 Triangular Bézier Patches

The tensor product surfaces discussed in the previous chapter have been defined over rectangular parameter domains. Now we will consider Bézier patches defined over *triangular* parameter domains. There are two major advantages:

– triangles give more topological flexibility

– tensor product surfaces have high polynomial degree, e. g. a biquadratic TP patch has actually degree 4 (see table below for all monomials occurring in a biquadratic TP patch).

v^2	v^2u	v^2u^2
v	vu	vu^2
1	u	u^2

9.1 Barycentric Coordinates in $\mathbb{R}^2$

Normally, a point $\mathbf{p} \in \mathbb{R}^2$ is given by an origin $\mathbf{O}$ and two directions $\mathbf{e}_1$ and $\mathbf{e}_2$ such that $\mathbf{p} = \mathbf{O} + \zeta\,\mathbf{e}_1 + \eta\,\mathbf{e}_2$. We call ζ and η the *standard coordinates* of $\mathbf{p}$ with respect to the coordinate system $\{\mathbf{e}_1, \mathbf{e}_2\}$. Now we introduce another type of 2D coordinates based on three points:

Definition 9.1 (Barycentric coordinates)
Given: Three points $\mathbf{R}$, $\mathbf{S}$ and $\mathbf{T}$ that are not on a line. An arbitrary point $\mathbf{p}$ can be represented as an affine combination

$$\mathbf{p} = \rho\,\mathbf{R} + \sigma\,\mathbf{S} + \tau\,\mathbf{T}\,, \quad \rho + \sigma + \tau = 1. \tag{9.1}$$

The coefficients ρ, σ, τ are called *barycentric coordinates* of $\mathbf{p}$ with respect to the triangle $\triangle(R, S, T)$.

Theorem 9.1
Given: Triangle $\triangle(R, S, T)$ and point $\mathbf{p}$. The barycentric coordinates ρ, σ, τ of $\mathbf{p}$ with respect to $\triangle(R, S, T)$ are unique and can be determined by

$$\rho = \frac{d(\mathbf{p}, \mathbf{S}, \mathbf{T})}{d(\mathbf{R}, \mathbf{S}, \mathbf{T})}, \quad \sigma = \frac{d(\mathbf{R}, \mathbf{p}, \mathbf{T})}{d(\mathbf{R}, \mathbf{S}, \mathbf{T})}, \quad \tau = \frac{d(\mathbf{R}, \mathbf{S}, \mathbf{p})}{d(\mathbf{R}, \mathbf{S}, \mathbf{T})}, \tag{9.2}$$

where

$$d(\mathbf{R}, \mathbf{S}, \mathbf{T}) := \det \begin{pmatrix} R_x & S_x & T_x \\ R_y & S_y & T_y \\ 1 & 1 & 1 \end{pmatrix}.$$

Sketch of Proof: Equation 9.1 can be rewritten as a linear system with tree variables:

$$\begin{pmatrix} R_x & S_x & T_x \\ R_y & S_y & T_y \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} \rho \\ \sigma \\ \tau \end{pmatrix} = \begin{pmatrix} p_x \\ p_y \\ 1 \end{pmatrix}$$

So we have

$$d(\mathbf{R},\mathbf{S},\mathbf{T}) = \det \begin{pmatrix} R_x - T_x & S_x - T_x \\ R_y - T_y & S_y - T_y \end{pmatrix} = \det(\mathbf{R}-\mathbf{T},\mathbf{S}-\mathbf{T}) \overset{!}{=} 0$$

$$\Leftrightarrow \ \mathbf{R}-\mathbf{T} \text{ and } \mathbf{S}-\mathbf{T} \text{ are linear dependent.}$$

The formulas for ρ, σ and τ follow from Cramer's rule. □

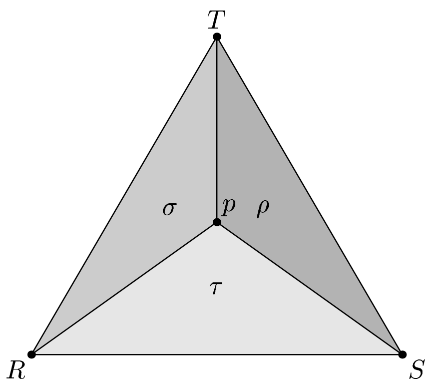


Figure 9.1: Barycentric coordinates

Remark:

- Because of $d(\mathbf{R},\mathbf{S},\mathbf{T}) = (\mathbf{S}-\mathbf{R}) \times (\mathbf{T}-\mathbf{R}) = 2 \cdot \text{area}(\mathbf{R},\mathbf{S},\mathbf{T})$ we can represent the barycentric coordinates also as ratios of areas, i. e.

$$\rho = \frac{\text{area}(\mathbf{p},\mathbf{S},\mathbf{T})}{\text{area}(\mathbf{R},\mathbf{S},\mathbf{T})}, \quad \sigma = \frac{\text{area}(\mathbf{R},\mathbf{p},\mathbf{T})}{\text{area}(\mathbf{R},\mathbf{S},\mathbf{T})}, \quad \tau = \frac{\text{area}(\mathbf{R},\mathbf{S},\mathbf{p})}{\text{area}(\mathbf{R},\mathbf{S},\mathbf{T})}$$

- $\mathbf{p}$ is inside $\triangle(\mathbf{R},\mathbf{S},\mathbf{T}) \ \Leftrightarrow \ \rho > 0, \ \sigma > 0$ and $\tau > 0$.
- If $\rho = \sigma = \tau$ then $\mathbf{p}$ is the barycenter of $\triangle(\mathbf{R},\mathbf{S},\mathbf{T})$.

9.2 Bernstein Polynomials over Triangles

A 1-dimensional analogon to barycentric coordinates: if c divides the interval $[a,b]$ such that $c = \alpha \cdot a + \beta \cdot b$ then $\alpha = \frac{c-b}{a-b}$ and $\beta = \frac{a-c}{a-b}$.
The k^{th} Bernstein polynomial of degree n over the interval $\Delta = [a,b]$ is defined as follows:

$$B_k^{\Delta,n}(c) = \frac{n!}{k!\,(n-k)!}\,\alpha^{n-k}\,\beta^k.$$

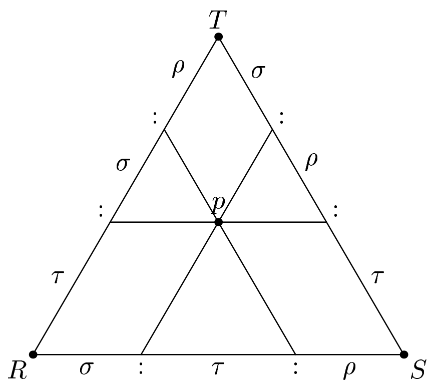


Figure 9.2: Divide ratios and barycentric coordinates

Definition 9.2 (Bernstein polynomial over triangles)
Given: Point $\mathbf{p}$ having barycentric coordinates ρ, σ, τ with respect to a triangle $\Delta := \triangle(\mathbf{R}, \mathbf{S}, \mathbf{T})$, a triple of indices i, j, k with degree n and $i + j + k = n$. Then:

$$B^{\Delta,n}_{i,j,k}(\mathbf{p}) := \frac{n!}{i!\,j!\,k!}\,\rho^i\,\sigma^j\,\tau^k \tag{9.3}$$

is called a *Bernstein polynomial* over the triangle Δ.

Example:

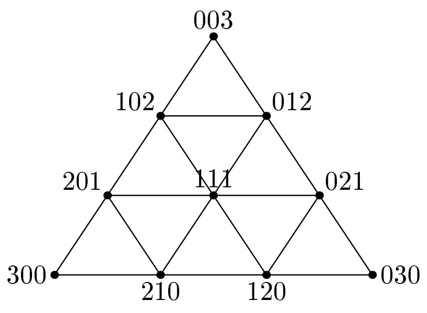


Figure 9.3: Arrangement of indices ijk for $n = 3$ Bernstein polynomials

Theorem 9.2
The Bernstein polynomials $\{B^n_{i,j,k} \,|\, i + j + k = n\}$ form a basis of $\mathbb{P}^2_n$ (= bivariate polynomials of degree $\leq n$).

Sketch of Proof: The number of Bernstein polynomials is $(n+1)+n+\cdots+1 = \frac{1}{2}(n+1)(n+2)$, which is the dimension of bivariate polynomials of degree $\leq n$. Example: for $n = 3$ we have 10 linear independent polynomials: $\{1, x, y, x^2, xy, y^2, x^3, x^2y, xy^2, y^3\}$. $\square$

Theorem 9.3 (Properties of triangular Bernstein Polynomials)
(1) If $\mathbf{p}$ is inside $\triangle(\mathbf{R}, \mathbf{S}, \mathbf{T})$, then $B^n_{i,j,k} > 0$.

(2) Partition of 1:

$$\sum_{i+j+k=n} B^n_{i,j,k}(\mathbf{p}) = 1 \tag{9.4}$$

(3) Recursion:

$$B^n_{i,j,k} = \rho\, B^n_{i-1,j,k} + \sigma\, B^n_{i,j-1,k} + \tau\, B^n_{i,j,k-1} \tag{9.5}$$

Sketch of Proof:

(1) follows from definition; $\rho, \sigma, \tau > 0$ if $\mathbf{p}$ is inside $\triangle(\mathbf{R}, \mathbf{S}, \mathbf{T})$.

(2) $1 = 1^n = (\rho + \sigma + \tau)^n = \displaystyle\sum_{i+j+k=n} \frac{n!}{i!\,j!\,k!}\,\rho^i\,\sigma^j\,\tau^k$

(3) $\rho\, B^n_{i-1,j,k} = \rho\, \dfrac{(n-1)!}{(i-1)!\,j!\,k!}\,\rho^{i-1}\,\sigma^j\,\tau^k = \dfrac{i}{n}\, B^n_{i,j,k}$.
Similar for $\sigma\, B^n_{i,j-1,k}$ and $\tau\, B^n_{i,j,k-1}$.

□

Theorem 9.4 (Polar Forms in 2D)
For each polynomial $F : \mathbb{R}^2 \to \mathbb{R}^d$ of degree n there exists a unique polar form $f : (\mathbb{R}^2)^n \to \mathbb{R}^d$ with

– f is symmetric

– f is multiaffine, i. e.
$f(\mathbf{z}_1, (\alpha\,\hat{\mathbf{z}} + \beta\,\tilde{\mathbf{z}}), \mathbf{z}_3, \ldots, \mathbf{z}_n) = \alpha\, f(\mathbf{z}_1, \hat{\mathbf{z}}, \mathbf{z}_3, \ldots, \mathbf{z}_n) + \beta\, f(\mathbf{z}_1, \tilde{\mathbf{z}}, \mathbf{z}_3, \ldots, \mathbf{z}_n)$

– f is diagonal, i. e. $F(\mathbf{z}) = f(\underbrace{\mathbf{z}, \ldots, \mathbf{z}}_{n})$.

Example: $\mathbf{z} := (x, y),\ \mathbf{z}_1 := (x_1, y_1),\ \mathbf{z}_2 := (x_2, y_2)$

$F(\mathbf{z})$	$f(\mathbf{z}_1, \ldots, \mathbf{z}_n)$
1	1
x	$\frac{1}{2}(x_1 + x_2)$
y	$\frac{1}{2}(y_1 + y_2)$
x^2	$x_1\, x_2$
$x\, y$	$\frac{1}{2}(x_1\, y_2 + x_2\, y_1)$
y^2	$y_1\, y_2$

9.3 Triangular Bézier Patches

Definition 9.3 (Triangular Bézier Patch)
Given: degree n, parameter domain $\Delta := \triangle(\mathbf{R}, \mathbf{S}, \mathbf{T})$, points $\mathbf{b}_{ijk}$ with $i + j + k = n$. Then:

$$F(\mathbf{z}) = \sum_{i+j+k=n} B^{\Delta,n}_{i,j,k}(\mathbf{z})\, \mathbf{b}_{ijk} \tag{9.6}$$

is called *triangular Bézier patch* over the triangle Δ.

The points $\mathbf{b}_{ijk}$ are called *control points* or *Bézier points*. They form the *control net*.

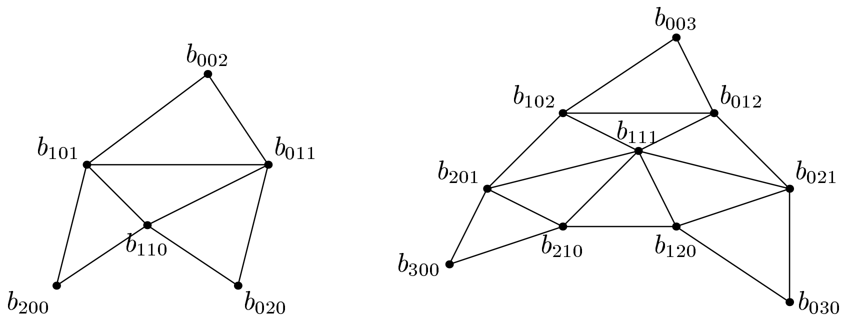


Figure 9.4: Triangular control nets for $n = 2$ and $n = 3$

Theorem 9.5 (Control Points and Polar Form)
Given: triangular Bézier patch $F(\mathbf{z}) = \sum_{i+j+k=n} B^{\Delta,n}_{i,j,k}(\mathbf{z})\, \mathbf{b}_{ijk}$. Then:

$$\mathbf{b}_{ijk} = f(\underbrace{R, \ldots, R}_{i}, \underbrace{S, \ldots, S}_{j}, \underbrace{T, \ldots, T}_{k}) \tag{9.7}$$

Sketch of Proof: Let ρ, σ, τ the barycentric coordinates of $\mathbf{z}$ such that $\mathbf{z} = \rho\,\mathbf{R} + \sigma\,\mathbf{S} + \tau\,\mathbf{T}$. Assume that $n = 3$, then

$$F(\mathbf{z}) = f(\mathbf{z}, \mathbf{z}, \mathbf{z}) = \rho\, f(\mathbf{R}, \mathbf{z}, \mathbf{z}) + \sigma\, f(\mathbf{S}, \mathbf{z}, \mathbf{z}) + \tau\, f(\mathbf{T}, \mathbf{z}, \mathbf{z}) = \cdots$$

$$= \sum_{i+j+k=3} \frac{n!}{i!\,j!\,k!}\, f(\underbrace{R, \ldots, R}_{i}, \underbrace{S, \ldots, S}_{j}, \underbrace{T, \ldots, T}_{k}).$$

□

Theorem 9.6 (Shape Properties of Triangular Bézier Patches)

(a) convex hull property: $F(\mathbf{z}) \in$ convex hull of $\{\mathbf{b}_{ijk}\}$ for $\mathbf{z} \in \triangle(\mathbf{R}, \mathbf{S}, \mathbf{T})$.

(b) end point interpolation: $F(\mathbf{R}) = \mathbf{b}_{n00}$, $F(\mathbf{S}) = \mathbf{b}_{0n0}$, $F(\mathbf{T}) = \mathbf{b}_{00n}$.

(c) end point tangency: e. g. the tangent plane at $\mathbf{b}_{n00}$ is

$$T_{n00} = \operatorname{span}\{\mathbf{b}_{n00} - \mathbf{b}_{(n-1)10}, \mathbf{b}_{n00} - \mathbf{b}_{(n-1)01}\}.$$

(d) The boundary curves are Bézier curves formed by the boundary control points.

(e) affine invariance

9.4 Algorithms for Triangular Bézier Patches

9.4.1 The de Casteljau Algorithm

To evaluate $F(\mathbf{z})$, we need the barycentric coordinates ρ, σ, τ of $\mathbf{z}$.

Algorithm 9.1 (de Casteljau for Bézier triangles)

$$\begin{aligned} \mathbf{b}^0_{ijk} &:= \mathbf{b}_{ijk}, \quad i+j+k=n \\ \mathbf{b}^l_{ijk} &:= \rho\, \mathbf{b}^{l-1}_{i+1,j,k} + \sigma\, \mathbf{b}^{l-1}_{i,j+1,k} + \tau\, \mathbf{b}^{l-1}_{i,j,k+1}, \quad l=1,\dots,n; \quad i+j+k=n-l \end{aligned} \tag{9.8}$$

Then: $\mathbf{b}^n_{000} = F(\mathbf{z})$ with $\mathbf{z} = \rho\,\mathbf{R} + \sigma\,\mathbf{S} + \tau\,\mathbf{T}$.

Sketch of Proof: According to theorem 9.5, we have

$$\mathbf{b}^l_{ijk} = f(\underbrace{\mathbf{R},\dots,\mathbf{R}}_{i}, \underbrace{\mathbf{S},\dots,\mathbf{S}}_{j}, \underbrace{\mathbf{T},\dots,\mathbf{T}}_{k}, \underbrace{\mathbf{z},\dots,\mathbf{z}}_{l=n-i-j-k}).$$

This is a solution for the recursion formula, it leads to $\mathbf{b}^n_{000} = f(\mathbf{z},\dots,\mathbf{z}) = F(\mathbf{z})$. □

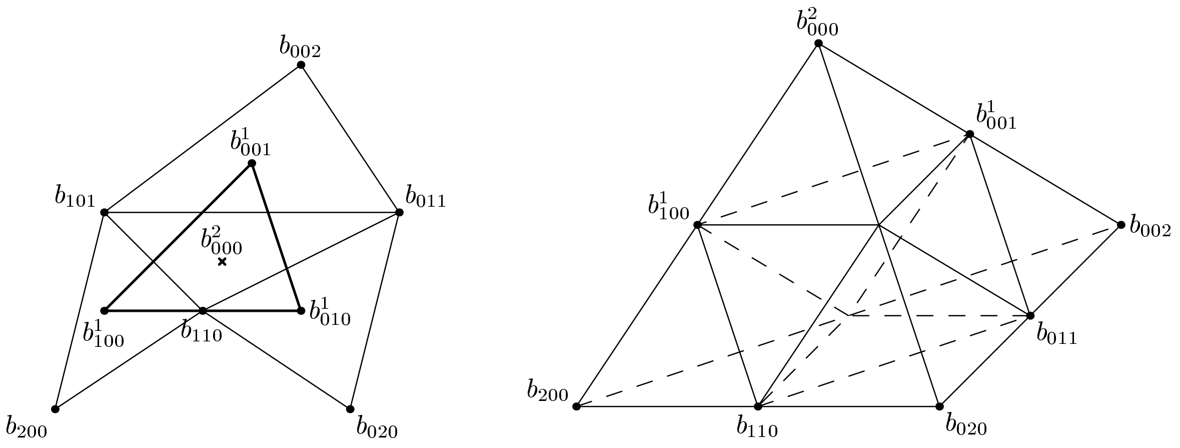


Figure 9.5: de Casteljau scheme for triangular Bézier patches

9.4.2 Degree Elevation

As an example, we first consider here the degree elevation from quadratic to cubic triangular Bézier patches:

$$F(\mathbf{z}) = \sum_{i+j+k=2} B^{\Delta,2}_{i,j,k}\, \mathbf{b}_{ijk} \overset{!}{=} \sum_{i+j+k=3} B^{\Delta,3}_{i,j,k}\, \bar{\mathbf{b}}_{ijk}$$

To find the $\bar{\mathbf{b}}_{ijk}$, we need a mapping for the polar forms $f_2(\mathbf{z}_1, \mathbf{z}_2) \leftrightarrow f_3(\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3)$:

$$f_3(\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3) = \frac{1}{3}\left(f_2(\mathbf{z}_1, \mathbf{z}_2) + f_2(\mathbf{z}_1, \mathbf{z}_3) + f_2(\mathbf{z}_2, \mathbf{z}_3)\right) \tag{9.9}$$

It is easy to see that this is symmetric, multiaffine and diagonal, so f_3 is a polar form.
Example:

$$\bar{\mathbf{b}}_{300} = f_3(\mathbf{R},\mathbf{R},\mathbf{R}) = \frac{1}{3}(3\,f_2(\mathbf{R},\mathbf{R})) = f_2(\mathbf{R},\mathbf{R}) = \mathbf{b}_{200}$$

$$\bar{\mathbf{b}}_{030} = f_3(\mathbf{S},\mathbf{S},\mathbf{S}) = f_2(\mathbf{S},\mathbf{S}) = \mathbf{b}_{020}$$

$$\bar{\mathbf{b}}_{003} = f_3(\mathbf{T},\mathbf{T},\mathbf{T}) = f_2(\mathbf{T},\mathbf{T}) = \mathbf{b}_{002}$$

$$\bar{\mathbf{b}}_{210} = f_3(\mathbf{R},\mathbf{S},\mathbf{S}) = \frac{1}{3}(f_2(\mathbf{R},\mathbf{R}) + 2\,f_2(\mathbf{R},\mathbf{S})) = \frac{1}{3}\mathbf{b}_{200} + \frac{2}{3}\mathbf{b}_{110}$$

$$\bar{\mathbf{b}}_{111} = f_3(\mathbf{R},\mathbf{S},\mathbf{T}) = \frac{1}{3}(f_2(\mathbf{R},\mathbf{S}) + f_2(\mathbf{R},\mathbf{T}) + f_2(\mathbf{S},\mathbf{T})) = \frac{1}{3}(\mathbf{b}_{110} + \mathbf{b}_{101} + \mathbf{b}_{011})$$

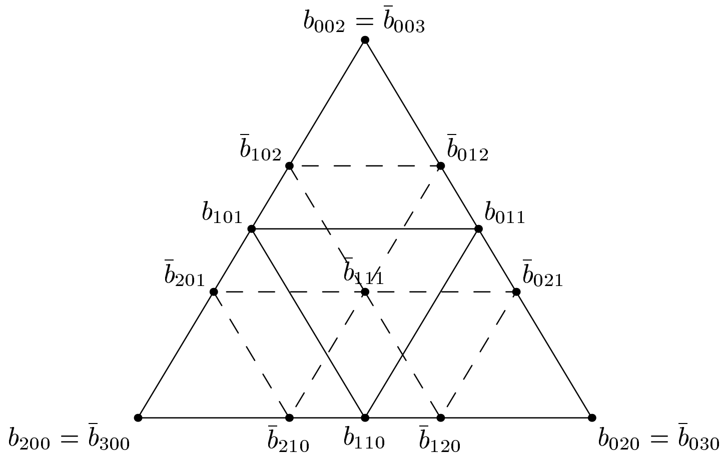


Figure 9.6: Degree elevation $n = 2 \rightarrow n = 3$

Algorithm 9.2 (Degree elevation)
Given: Bézier triangle $F(\mathbf{z}) = \sum_{i+j+k=n} B^{\Delta,n}_{i,j,k}\,\mathbf{b}_{ijk}$

Find: control points $\bar{\mathbf{b}}_{ijk}$ such that $F(\mathbf{z}) = \sum_{i+j+k=n+1} B^{\Delta,n+1}_{i,j,k}\,\bar{\mathbf{b}}_{ijk}$

$$\bar{\mathbf{b}}_{ijk} = \frac{1}{n+1}\left(i\,\mathbf{b}_{i-i,j,k} + j\,\mathbf{b}_{i,j-1,k} + k\,\mathbf{b}_{i,j,k-1}\right) \tag{9.10}$$

9.4.3 Subdivision

There are two possibilities to perform a subdivision for triangular Bézier patches:

– 1st possibility: divide $\triangle(\mathbf{R},\mathbf{S},\mathbf{T})$ into two triangles (“1-to-2 split”).

  E. g. dividing the line segment $\overline{\mathbf{RT}}$ with $\mathbf{Q}$ gives

$$\triangle(\mathbf{R},\mathbf{S},\mathbf{T}) \rightarrow \{\triangle(\mathbf{R},\mathbf{S},\mathbf{Q}), \triangle(\mathbf{Q},\mathbf{S},\mathbf{T})\}$$

– 2nd possibility: divide $\triangle(\mathbf{R}, \mathbf{S}, \mathbf{T})$ into four triangles ("1-to-4 split", see figure 9.7).

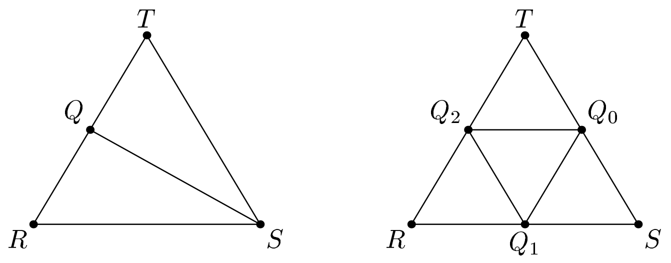


Figure 9.7: Splitting triangles: 1-to-2 and 1-to-4 split

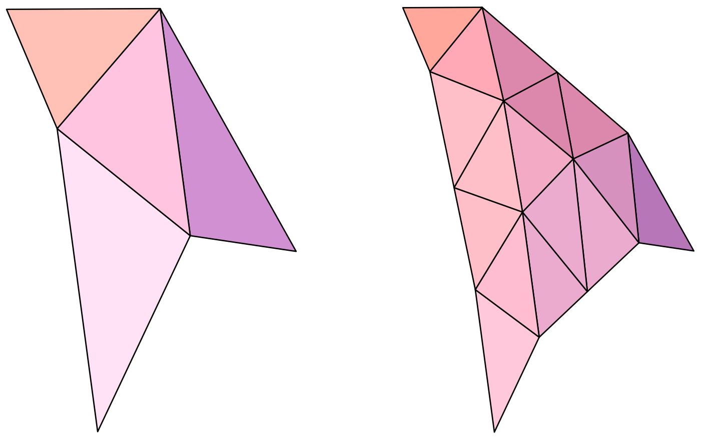


Figure 9.8: A control net and the first 1-to-4 subdivision step

Algorithm 9.3 (1-to-2 split)
Perform de Casteljau algorithm:

$$\mathbf{b}^l_{ijk} = f(\underbrace{\mathbf{R}, \ldots, \mathbf{R}}_{i}, \underbrace{\mathbf{S}, \ldots, \mathbf{S}}_{j}, \underbrace{\mathbf{T}, \ldots, \mathbf{T}}_{k}, \underbrace{\mathbf{z}, \ldots, \mathbf{z}}_{l=n-i-j-k})$$

for $\mathbf{z} = \mathbf{Q}$. Then: the outer faces of the resulting tetrahedron are control points of the two subdivided triangles (see figure 9.5).

Algorithm 9.4 (1-to-4 split, Farin-Prautzsch method)
Perform four 1-to-2 splits with de Casteljau (see figure 9.9).

Instead of subsequently using the de Casteljau algorithm for subdivision, it is also possible to pre-calculate the split points using polar forms:

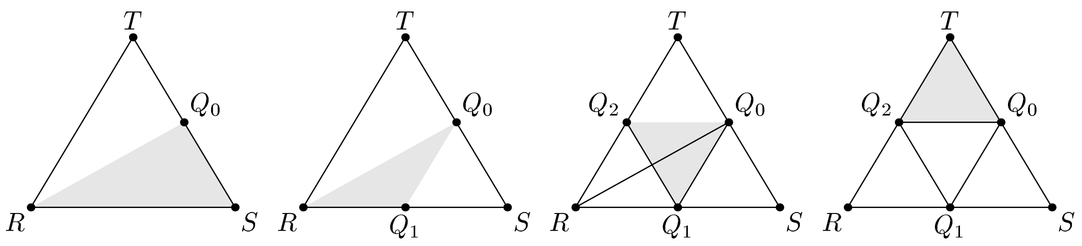


Figure 9.9: Farin-Prautzsch 1-to-4 subdivision scheme

Example: If we want to split at $\mathbf{Q}_1 = \frac{\mathbf{R}+\mathbf{S}}{2}$ and $\mathbf{Q}_2 = \frac{\mathbf{R}+\mathbf{T}}{2}$ we get:

$$f(\mathbf{R},\mathbf{R}) = \mathbf{b}_{200}$$

$$f\left(\frac{\mathbf{R}+\mathbf{T}}{2},\frac{\mathbf{R}+\mathbf{T}}{2}\right) = \frac{1}{4}\mathbf{b}_{200} + \frac{1}{2}\mathbf{b}_{101} + \frac{1}{4}\mathbf{b}_{002}$$

$$f\left(\frac{\mathbf{R}+\mathbf{S}}{2},\frac{\mathbf{R}+\mathbf{S}}{2}\right) = \frac{1}{4}\mathbf{b}_{200} + \frac{1}{2}\mathbf{b}_{110} + \frac{1}{4}\mathbf{b}_{020}$$

$$f\left(\mathbf{R},\frac{\mathbf{R}+\mathbf{S}}{2}\right) = \frac{1}{2}\mathbf{b}_{200} + \frac{1}{2}\mathbf{b}_{110}$$

$$f\left(\mathbf{R},\frac{\mathbf{R}+\mathbf{T}}{2}\right) = \frac{1}{2}\mathbf{b}_{200} + \frac{1}{2}\mathbf{b}_{101}$$

$$f\left(\frac{\mathbf{R}+\mathbf{S}}{2},\frac{\mathbf{R}+\mathbf{T}}{2}\right) = \frac{1}{4}\mathbf{b}_{200} + \frac{1}{4}\mathbf{b}_{110} + \frac{1}{4}\mathbf{b}_{101} + \frac{1}{4}\mathbf{b}_{011}$$

Supplement: Derivatives of Multivariate Functions

Considering multivariate (here: bivariate) functions, there are three types of derivatives:

- partial derivatives are $\frac{\partial F}{\partial u}$ and $\frac{\partial F}{\partial v}$
- directional derivative: the derivative of $F(u,v)$ at a certain point $\mathbf{p}$ in direction $\vec{\mathbf{w}} = (w_u, w_v)$ is

$$\frac{\partial F}{\partial \vec{\mathbf{w}}} = \lim_{h\to 0}\frac{F(\mathbf{p}+h\,\vec{\mathbf{w}}) - F(\mathbf{p})}{h} = w_u\,\frac{\partial F}{\partial u} + w_v\,\frac{\partial F}{\partial v}$$

- total derivative: the Jacobian matrix

$$J_F(\mathbf{p}) = \left(\frac{\partial F}{\partial u}(\mathbf{p}), \frac{\partial F}{\partial v}(\mathbf{p})\right)$$

For bivariate functions, the Jacobian matrix is a $2 \times n$ matrix with n the dimension of the space $(F : \mathbb{R}^m \to \mathbb{R}^n \;\Rightarrow\; J_F \in \mathbb{R}^{(n,m)})$.

9.5 C^k-Continuity of Triangular Bézier Patches

Given: two Bézier triangles F and G which have the parameter domains $\Delta_F = \triangle(\mathbf{R}, \mathbf{S}, \mathbf{T})$ and $\Delta_G = \triangle(\mathbf{Q}, \mathbf{S}, \mathbf{T})$.

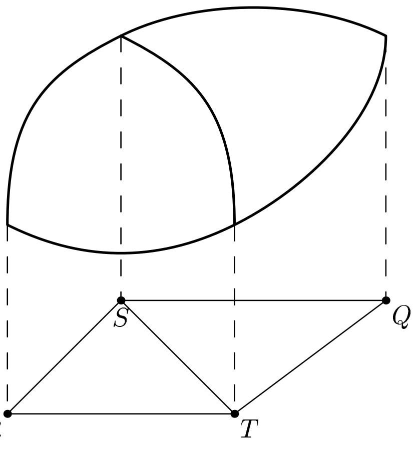


Figure 9.10: Attaching two Bézier triangles continuously

How to determine that F and G are C^k-continuous along the edge $\overline{\mathbf{ST}}$? Assume $\mathbf{p} \in \overline{\mathbf{ST}}$ a point on the edge in *cartesian* coordinates $\mathbf{p} = (p_u, p_v)$ (and assume that F and G are also given in cartesian representation).

- $k = 0$: $F(\mathbf{p}) = G(\mathbf{p})$.
- $k = 1$: C^0 and $\dfrac{\partial F}{\partial u} = \dfrac{\partial G}{\partial u}$ and $\dfrac{\partial F}{\partial v} = \dfrac{\partial G}{\partial v}$.
- $k = 2$: C^1 and $\dfrac{\partial^2 F}{\partial u^2} = \dfrac{\partial^2 G}{\partial u^2}, \quad \dfrac{\partial^2 F}{\partial v^2} = \dfrac{\partial^2 G}{\partial v^2}$ and $\dfrac{\partial^2 F}{\partial u\,\partial v} = \dfrac{\partial^2 G}{\partial u\,\partial v}$
- In general: $\dfrac{\partial^k F}{\partial u^i\,\partial v^j} = \dfrac{\partial^k G}{\partial u^i\,\partial v^j}$ for all $i + j = k$.

C^k-continuity can be characterized using polar forms. Assume f and g are polar forms of F and G.

- $k = 0$:

$$f(\mathbf{S}, \dots, \mathbf{S}) = g(\mathbf{S}, \dots, \mathbf{S})$$
$$f(\mathbf{S}, \dots, \mathbf{S}, \mathbf{T}) = g(\mathbf{S}, \dots, \mathbf{S}, \mathbf{T})$$
$$\vdots$$
$$f(\mathbf{T}, \dots, \mathbf{T}) = g(\mathbf{T}, \dots, \mathbf{T})$$

- $k = 1$:

$$f(\mathbf{S}, \dots, \mathbf{S}, \mathbf{z}) = g(\mathbf{S}, \dots, \mathbf{S}, \mathbf{z})$$
$$f(\mathbf{S}, \dots, \mathbf{S}, \mathbf{T}, \mathbf{z}) = g(\mathbf{S}, \dots, \mathbf{S}, \mathbf{T}, \mathbf{z})$$
$$\vdots$$

$f(\mathbf{T}, \dots, \mathbf{T}, \mathbf{z}) = g(\mathbf{T}, \dots, \mathbf{T}, \mathbf{z})$

for all $\mathbf{z} \in \mathbb{R}^2$. An equivalent condition for C^1-continuity: C^0 holds and the above equations hold for $\mathbf{z} = \mathbf{R}$.

– $k = 2$:

$f(\underbrace{\mathbf{S}, \dots, \mathbf{S}}_{i}, \underbrace{\mathbf{T}, \dots, \mathbf{T}}_{j}, \mathbf{z}_1, \mathbf{z}_2) = g(\underbrace{\mathbf{S}, \dots, \mathbf{S}}_{i}, \underbrace{\mathbf{T}, \dots, \mathbf{T}}_{j}, \mathbf{z}_1, \mathbf{z}_2)$

for all $\mathbf{z}_1$, $z_2 \in \mathbb{R}^2$. Equivalent: C^1 holds and $\mathbf{z}_1 = \mathbf{R}$ and $\mathbf{z}_2 = \mathbf{R}$.

In general:

Theorem 9.7
The Bézier triangles F and G with the parameter domains $\Delta_F = \triangle(\mathbf{R}, \mathbf{S}, \mathbf{T})$ and $\Delta_G = \triangle(\mathbf{Q}, \mathbf{S}, \mathbf{T})$ are C^k-continuous if

$$f(\underbrace{\mathbf{S}, \dots, \mathbf{S}}_{i}, \underbrace{\mathbf{T}, \dots, \mathbf{T}}_{j}, \underbrace{\mathbf{R}, \dots, \mathbf{R}}_{l}) = g(\underbrace{\mathbf{S}, \dots, \mathbf{S}}_{i}, \underbrace{\mathbf{T}, \dots, \mathbf{T}}_{j}, \underbrace{\mathbf{R}, \dots, \mathbf{R}}_{l}) \tag{9.11}$$

for all $0 \leq l \leq k$ and $i + j = n - l$.

Geometric interpretation (for $k = 1$):

$\mathbf{R} = \omega\,\mathbf{Q} + \sigma\,\mathbf{S} + \tau\,\mathbf{T}$ is a representation of $\mathbf{R}$ in barycentric coordinates with respect to $\triangle(\mathbf{S}, \mathbf{Q}, \mathbf{T})$. Then:

$$f(\mathbf{S}, \dots, \mathbf{S}, \mathbf{T}, \dots, \mathbf{T}, \mathbf{R}) = g(\mathbf{S}, \dots, \mathbf{S}, \mathbf{T}, \dots, \mathbf{T}, \mathbf{R}) =$$
$$= \omega\, g(\mathbf{S}, \dots, \mathbf{S}, \mathbf{T}, \dots, \mathbf{T}, \mathbf{Q}) + \sigma\, g(\mathbf{S}, \dots, \mathbf{S}, \mathbf{T}, \dots, \mathbf{T}, \mathbf{S}) + \tau\, g(\mathbf{S}, \dots, \mathbf{S}, \mathbf{T}, \dots, \mathbf{T}, \mathbf{T}).$$

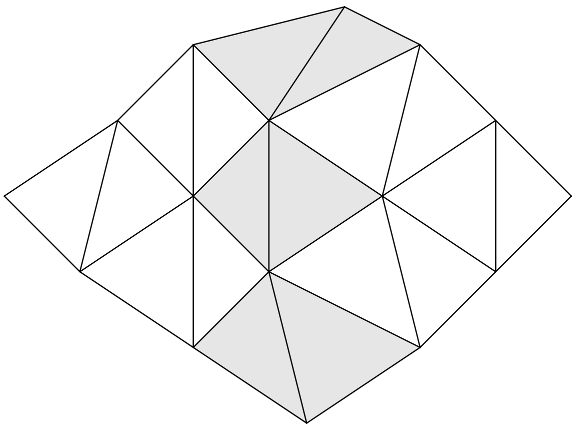


Figure 9.11: “Diamond condition” for $n = 3$

These quadrilaterals (highlighted in figure 9.11) are affine images of the parameter quadrilateral $(\mathbf{R}, \mathbf{S}, \mathbf{Q}, \mathbf{T})$, therefore they have to be *planar* (“diamond condition”).

9.6 Interpolation with piecewise Bézier Triangles (Split Surfaces)

One application of triangular Bézier patches is the interpolation of scattered data points using smooth triangular surfaces.
Given: data points $(x_1, y_1), \dots, (x_n, y_n)$ and corresponding data values $z_1, \dots, z_n$.

Find: a smooth (i. e. C^1) scalar function $f : \mathbb{R}^2 \to \mathbb{R}$ that interpolates the given data, i. e. $f(x_i, y_i) = z_i$.

Note that there exist many approaches to solve this problem.

General procedure:

- 1st step: triangulate the set of data points, e. g. using Delaunay triangulation (see lecture on Graphical Algorithms for details).
- 2nd step: split each triangle. We want to discuss here two methods of splitting, the
  - *Clough-Tocher* method that gives three cubic Bézier patches and the
  - *Powell-Sabin* method that gives six quadratic Bézier patches.
- 3rd step: estimate the normal vectors (or partial derivatives) at each data point. These are used to determine the control points.

9.6.1 Clough-Tocher

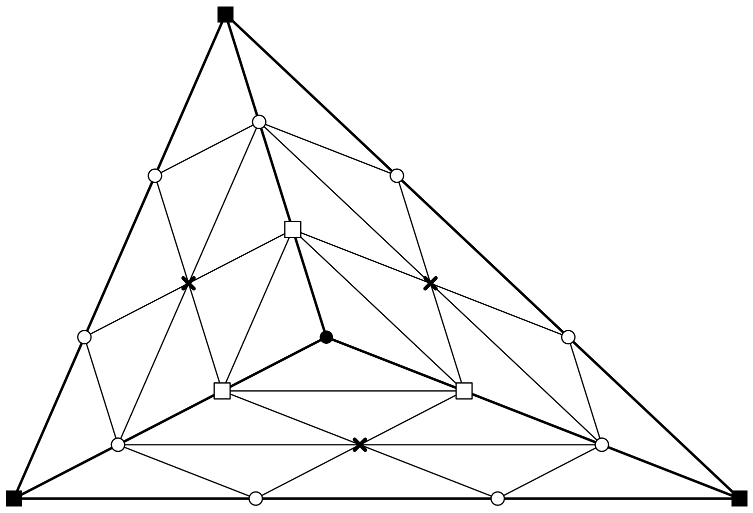


Figure 9.12: Clough-Tocher split surface scheme

This is how the control points in figure 9.12 are obtained:

- ■: given data points (x_i, y_i, z_i).
- ○: specified by the directional derivatives obtained in 3rd step.
- $\times$: Assume that the orthogonal derivative along an edge is linear and linearly interpolate the orthogonal derivative at the end points.

- □: “diamond condition” at the interior edges for C^1-continuity
- ●: choose the barycenter of the triangle.

9.6.2 Powell-Sabin

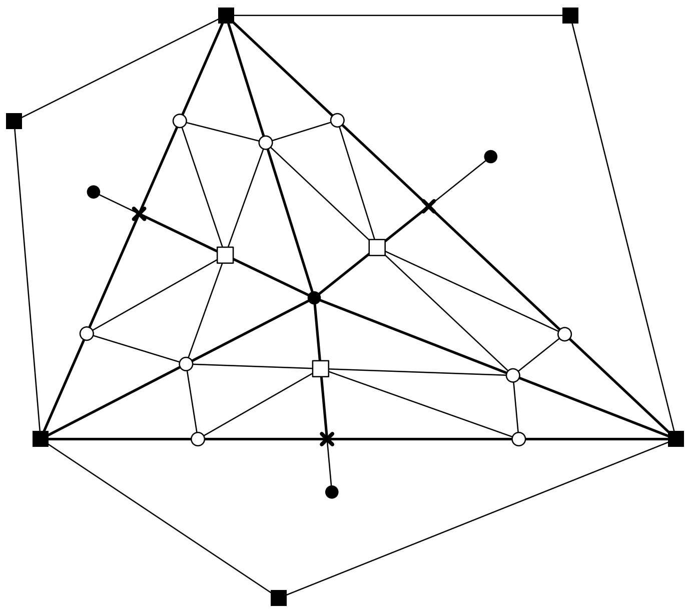


Figure 9.13: Powell-Sabin split surface scheme

Here, the control points according to figure 9.13 are obtained as follows:

- ■: given data points (x_i, y_i, z_i).
- ○: specified by the directional derivatives obtained in 3rd step.
- $\times$: “diamond condition” at the outer edges for C^1-continuity
- □: “diamond condition” at the interior edges for C^1-continuity
- ●: barycenters of the given triangles (incenters can also be used; the only condition is that ● has always the same barycentric coordinates with respect to its corresponding triangle).

10 Triangle meshes

Triangle meshes are used for modelling geometric objects. The representation of triangles is supported by the graphics hardware and there are efficient algorithms.
Triangle meshes consist of:

- a set of **vertices** $V = \{v_i\}, \quad v_i \in \mathbb{R}^3$
- a set of **edges** $E = \{e_{ij}\}, \quad e_{ij} \simeq \overline{v_i v_j} = [v_i, v_j] \simeq \{i, j\}$
- a set of **triangles** $T = \{t_{ijk}\}_{(i,j,k)}, \quad t_{ijk} \simeq \triangle(v_i, v_j, v_k) = [v_i, v_j, v_k] \simeq \{i, j, k\}$ with $i \neq j \neq k$

10.1 Basics

Notations:

- For vertices v_i the edges e and triangles t with $i \in e$ resp. $i \in t$ are called **adjacent**.
- For edges $e = \{i, j\}$ all triangles t with $e \subset t$ are called **adjacent**.
- **(1-)neighbourhood** of a vertice: *all* adjacent triangles and the edges and vertices they consist of (see Figure 10.1)
- **2-neighbourhood** of a vertice: neighbourhood of the neighbourhood (see Figure 10.1)
- **n-neighbourhood**: neighbourhood of the $(n - 1)$-neighbourhood
- **valence** η_v of a vertice v: number of adjacent edges resp. number of neighbour vertices

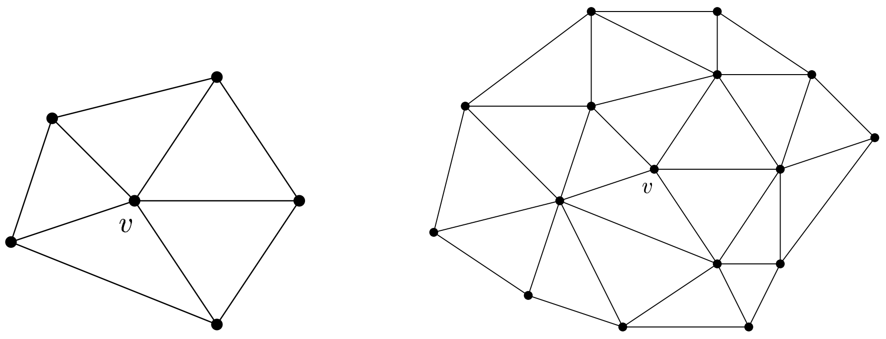


Figure 10.1: 1-neighbourhood and 2-neighbourhood of a vertice v

We only consider *manifold* triangulation, i. e. triangle meshes with the following properties:

(1) The intersection of two different triangles is

- a common edge,
- a common vertice or
- empty. (see Figure 10.2)

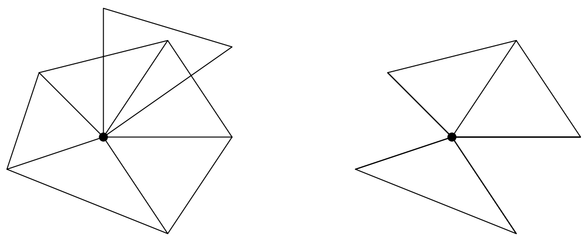


Figure 10.2: not possible in manifold triangulations

(2) Edges have

- *one* adjacent triangle $\Rightarrow$ border edge
- oder *two* adjacent triangles $\Rightarrow$ inner edge.

(3) If the adjacent triangles of a vertice build a

- *open fan* (see Figure 10.3), the vertice is a vertice at the border: $\eta_v = \#(\text{adjacent triangles}) + 1$
- *closed fan* (see Figure 10.3), the vertice is a inner vertice: $\eta_v = \#(\text{adjacent triangles})$

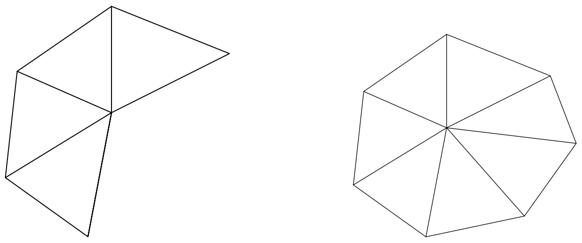


Figure 10.3: open and closed fan

(4) orientable surfaces, i.e.

- the normals can be oriented constantly
- *no* Moebius Strips and Klein Bottles (see Figure 10.4)

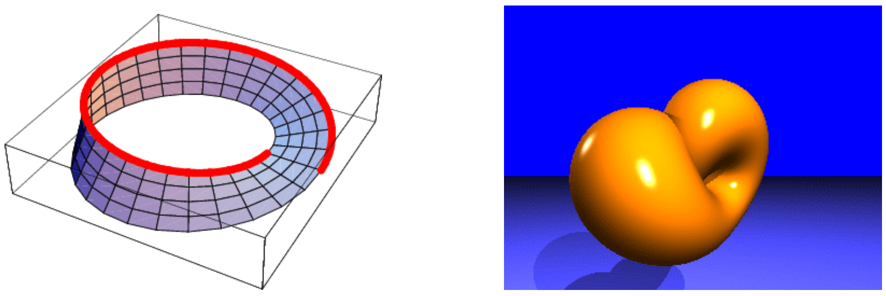


Figure 10.4: Moebius Strip and Klein Bottle

(5) The triangle is connected

$\Rightarrow$ the border edges build a close polygon.

Example: Tetrahedron

$$V = \{(1,1,1), (-1,1,-1), (-1,-1,1), (1,-1,-1)\}$$
$$E = \{\{0,1\}, \{0,2\}, \{0,3\}, \{1,2\}, \{1,3\}, \{2,3\}\}$$
$$T = \{\{0,1,2\}, \{0,2,3\}, \{0,3,1\}, \{3,2,1\}\}$$

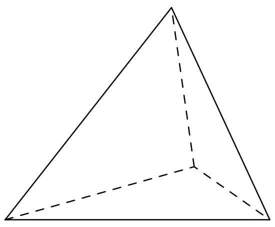


Figure 10.5: Tetrahedron

Theorem 10.1 (EULER formula for general polyhedrons)
If V is the number of vertices, E the number of edges, F the number of faces, G the genus of the surface (number of holes (see Figure 10.6)) and B the number of border polygons die Anzahl der Randpolygone, we have:

$$V - E + F = 2(1 - G) - B$$

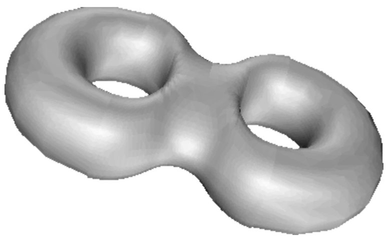


Figure 10.6: Two-torus: surface of genus 2

Example:

- sphere: $G = 0,\ B = 0 \quad \Rightarrow \quad V - E + F = 2$
- tetrahedron: $G = 0,\ B = 0,\ V = 4,\ E = 6,\ F = 4 \quad \Rightarrow \quad 4 - 6 + 4 = 2$
- torus: $G = 1,\ B = 0 \quad \Rightarrow \quad V - E + F = 0$

For the estimation of the required memory of a triangle mesh the following theorem is important.

Theorem 10.2
For large triangle meshes holds: $F \approx 2V,\ E \approx 3V$

Proof:

i) Each triangle has 3 edges, each inner edge belongs to 2 triangles.
$\Rightarrow 3F \approx 2E \quad \Leftrightarrow \quad F \approx 3V$

ii) $V - E + F \approx 0$ (because the genus and the number of border polygons is vanishing in large triangle meshes)
$\Rightarrow \quad V - E + \frac{2}{3}E \approx 0 \quad \Leftrightarrow \ V \approx \frac{1}{3}E$

□

10.2 Computer representation of triangle meshes

The information of triangle meshes consists of

– **geometry**: *where* are the vertices, edges and triangles?

– **topology:** *how* are the vertices connected to edges and triangles?

This information should be suitably stored for the application and as efficient as possible.

Remark: In the following holds:

Below there are presented three possible representations of triangle meshes in the computer:

(1) **explicit representation**
For all triangles t_{ijk} the vertices v_i, v_j, v_k are stored.
$\Rightarrow \qquad 2n \cdot 3 \cdot 3 = 18n$ floats

Remark: Maximal redundant, because the geometry is multiply stored.

(2) **"'shared vertex"'** or **"'indexed face set"'**

– list of coordinates of all vertices $\Rightarrow\ 3n$ floats
– list of all triangles by referencing the vertex list, i. e. for t_{ijk} we store $(i, j, k) \quad \Rightarrow$ $6\,n$ integers

Remark:

– Efficient because geometry and topology is separated. Therefore it is suitable for archiving the data.
– Edges are only stored implicitly. Explicit access is difficult.

  - Frequent operations are expensive (e. g. access to the neighbour of a vertex).
  - Vertices of the triangles are normally stored counter clockwise.

(3) **extended shared vertex**

  - list of vertices with reference to **one** adjacent triangle. (convention for border vertices: right border triangle) $\Rightarrow$ $3\,n$ floats $+\ 1\,n$ integers
  - list of triangles with reference to adjacent triangles (at the border: -1) $\Rightarrow$ $12n$ integers

Remark: The “ extended shared vertex” structure is suitable especially for the processing triangle meshes. E. g. neighbour vertices can be found easily because of the stored lists.

Example: tetrahedron

(1) explicit data structure

t	v_i	v_j	v_k
0	1.0, 1.0, 1.0	-1.0, 1.0, -1.0	-1.0, -1.0, 1.0
1	1.0, 1.0, 1.0	-1.0, -1.0, 1.0	1.0, -1.0, -1.0
2	1.0, 1.0, 1.0	1.0, -1.0, -1.0	-1.0, 1.0, -1.0
3	1.0, -1.0, -1.0	-1.0, -1.0, 1.0	-1.0, 1.0, -1.0

(2) shared vertex structure

v	x	y	z
0	1.0	1.0	1.0
1	-1.0	1.0	-1.0
2	-1.0	-1.0	1.0
3	1.0	-1.0	-1.0

t	i	j	k
0	0	1	2
1	0	2	3
2	0	3	1
3	3	2	1

(3) extended shared vertex structure

v	x	y	z	l
0	1.0	1.0	1.0	0
1	-1.0	1.0	-1.0	0
2	-1.0	-1.0	1.0	0
3	1.0	-1.0	-1.0	1

t	i	j	k	n_0	n_1	n_2
0	0	1	2	3	1	2
1	0	2	3	3	2	0
2	0	3	1	3	0	1
3	3	2	1	0	2	1

10.3 Parametrization of triangle meshes

Given: data points $v_i \in \mathbb{R}^3$
Find: parameter values $x_i \in \mathbb{R}^2$ for which the angles and splitting ratios of the triangle mesh are preserved.
Notations:

- border points: $v_i \in Bdy$
- (indices of) neighbour of v_i: $N(v_i)$

10.3.1 The spring modell

In order to parameterize the inner points we use the spring modell from physics. Interpretation: Springs are attached to the inner points and fixed along the edges at the neighbour points. The springs are idealised and have the minimum size 0. The mesh is getting flat and the springs are as minimal tensed as possible.
Parameterization of data points by spring modells requires the minimization of a quadratic functional. That corresponds to solving a linear system of equations.
The spring equation is:

$$F = \frac{1}{2} \sum_{(i,j)\in E} D_{ij} ||x_i - x_j||^2$$

Minimization:

$$\frac{\partial F}{\partial x_k} = \frac{1}{2} \sum_{(i,j)\in E} D_{ij} \frac{\partial}{\partial x_k} ||x_i - x_j||^2$$

$$\frac{\partial}{\partial x_k} ||x_i - x_j||^2 = \begin{cases} 2(x_i - x_j)\,, & k = i,\ j \in N(v_k) \\ -2(x_i - x_j)\,, & k = j,\ i \in N(v_k) \\ 0\,, & \text{sonst} \end{cases}$$

$$\frac{\partial F}{\partial x_k} = \sum_{j\in N(v_k)} D_{kj}(x_k - x_j) \overset{!}{=} 0$$

$$\begin{aligned}
&\Leftrightarrow x_k \sum_{j\in N(v_k)} D_{kj} - \sum_{j\in N(v_k)} D_{kj}x_j = 0\\
&\Leftrightarrow x_k \sum_{j\in N(v_k)} D_{kj} - \sum_{\substack{j\in N(v_k)\\ v_j\notin Bdy}} D_{kj}x_j = \sum_{\substack{j\in N(v_k)\\ v_j\in Bdy}} D_{kj}x_j\\
&\Leftrightarrow A\cdot x = b
\end{aligned}$$

with

$$A_{ij} = \begin{cases} \sum_{k\in N(v_i)} D_{ik}\,, & i=j\\ -D_{ij}\,, & j\in N(v_i)\\ 0\,, & \text{sonst}\end{cases}; \qquad x = \begin{pmatrix}\vdots\\ x_i\\ \vdots\end{pmatrix}; \qquad b_i = \sum_{\substack{j\in N(v_i)\\ v_j\in Bdy}} D_{ij}x_j$$

Remark: A is invertible because it is diagonal dominant.

Proceeding of parameterization:

(1) parameterization of the border vertices

(2) parameterization of the inner vertices

10.3.2 Parametrization of the border vertices

First the border vertices of the triangle mesh have to be parameterized by one of the following possibilities.

i) method for curves over a circle

ii) specify n corners and parameterize over a polygon with n corners

iii) projection of the border vertices onto the least squares plane

Remark: It is possible that method iii) leads to convolutions of the border polygon. The projected border polygon is not convex necessariliy.

10.3.3 Parametrization of the inner vertices

Given: vertices $v_i \in \mathbb{R}^3$
Find: parameter values $x_i \in \mathbb{R}^2$, which fulfill the following:

(1) identity property: parameterization is identical with the triangle mesh, that has been pressed into the plane.

(2) regularity resp. no convolutions

Choice of the D_{ij}:

a)

$$D_{ij} = \frac{1}{||v_i - v_j||^p} \qquad \text{are called} \begin{cases} p = 0 & \textit{uniform} \text{ parametrization} \\ p = 1/2 & \textit{zentripetal} \text{ parametrization} \\ p = 1 & \textit{chordal} \text{ parametrization} \end{cases}$$

Disadvantage: If the given triangle mesh lies in a plane, the mesh is getting deformed. The parameterization does not fulfill (1), but (2).

b) *discret harmonic* parameterization

$$D_{ij} = \frac{1}{2}(\cot\alpha + \cot\beta)$$

Disadvantage: parameterized triangle mesh might be convoluted.
The parameterization fulfills (1), but not (2).

c) *shape preserving* parameterization: fulfills (1) and (2)

i) Flattening of the 1-neighbourhood of a inner point v by exponential map (see Figure 10.3.3 (a) and (b)):

$$p = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \text{and} \quad q_i = ||w_i - v|| \begin{pmatrix} \cos\sigma_i \\ \sin\sigma_i \end{pmatrix}$$

$$\text{with} \quad \sigma_i = \rho \sum_{j=1}^{i-1} \gamma_j \,, \quad \rho = \frac{2\pi}{\sum_{i=1}^n \gamma_i} \quad \text{and} \quad \gamma_i = \sphericalangle(w_i, v, w_{i+1})$$

ii) determine D_{ij} by averaging the barycentric coordinates (see Section 9.1)
For each q_i we find one j, so that $p \in \delta(q_i, q_j, q_{j+1})$,
i. e. $p = \tau_i^i q_i + \tau_j^i q_j + \tau_{j+1}^i q_{j+1}$ with $\tau_i^i + \tau_j^i + \tau_{j+1}^i = 1$
all other $\tau_k^i = 0\,,\ k \neq i, j, j+1$
(see Figure 10.3.3 (c))
Finally: $D_{vw_i} = \frac{1}{n}\sum_{j=1}^n \tau_i^j$
Dieses D_{vw_i} ist das D_{kl} zwischen den Punkten $v_k = v$ und $v_l = w_i$.
Es ist klar, dass

i)

$$p = \sum_{i=1}^n D_{vw_i} \cdot q_i \,, \ \text{denn f"ur alle } i \text{ gilt } \ p = \sum_{j=1}^n \tau_j^i q_j$$

ii)

$$\sum_{i=1}^{n} D_{vw_i} = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\tau_i^j = \frac{1}{n}\sum_{j=1}^{n}\underbrace{\sum_{i=1}^{n}\tau_i^j}_{=1} = 1$$

$\Rightarrow$ p ist damit Konvexkombination der q_i

Remark: Diese D_{ij} sind i. d. R. *nicht* symmetrisch, d. h. $D_{ij} \neq D_{ji}$.

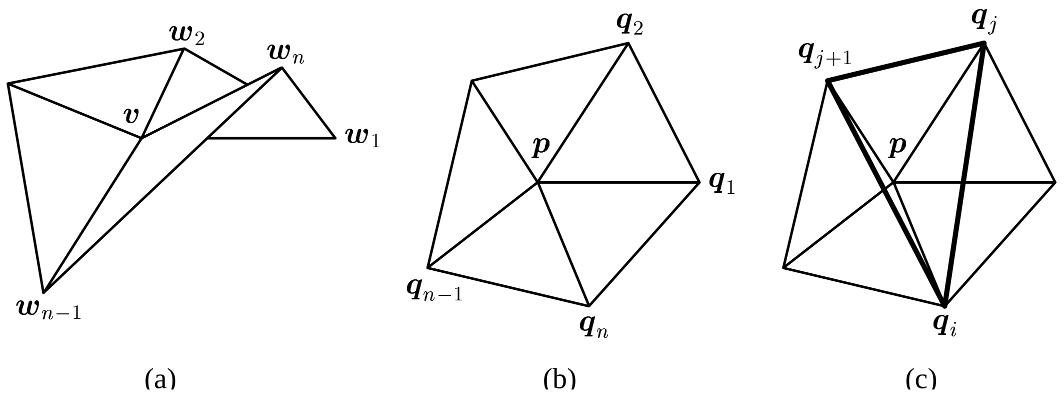


Figure 10.7: A local part of a triangular mesh (a) is mapped into the plane by expontial mapping (b) and the local weights are determined (c).

11 Subdivision Surfaces

Motivation: Comparison TP-Surfaces - Triangle Meshes

	TP-Surfaces	Triangle Meshes
memory requirements	+	-
smooth surface	+	-
display on screen	-	+
free forms	-	+
complex forms	-	+

Subdivision surfaces are a compromise between TP-surfaces and triangle meshes. In the following, we learn that Subdivision Surfaces offer a good possibility for surface modelling (e. g. in movies). The modelling process consists of two steps: First build a coarse triangle mesh of the object, then apply several subdivision steps. We receive a smooth surface, which is C^1- or C^2-continuous in the limit and interpolates or approximates the original mesh.
Basic idea: 1-to-4 split (see Figure 11.1)

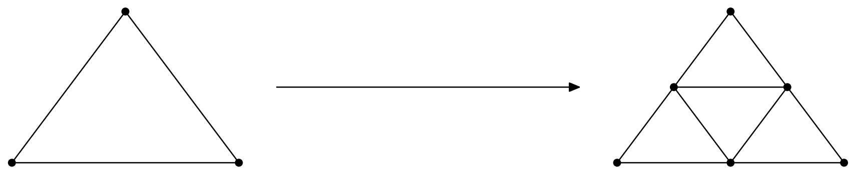


Figure 11.1: 1-to-4 split

Analogously: subdivision of curves

useful terms:

- "old" points: even vertices
- "new" points: odd vertices

There are two kinds of subdivision schemes:

(1) interpolative subdivision: Even vertices are kept, odd vertices are added.

(2) approximative subdivision: Also the position of the even vertices changes.

Remark:

- The more subdivision steps, the smaller the triangles (and the smoother the surface).
- In the limit (after ∞ subdivision steps): smooth surface

Advantages and disadvantages of subdivision surfaces:

+ arbitrary topology

+ scalability

+ compromise between triangle meshes and TP-surfaces

+ numerical stability

+ code simplicity

+ adaptivity

– no simple method with global C^2-continuity in the limit

– artifacts at vertices with high valence

Objectives for the development of subdivision schemes:

- efficient calculation of even and odd vertices
- simplicity and locality of this rules
- affine invariance $\Rightarrow$ convex combinations
- continuity of the limit surface

11.1 Approximative schemes: Loop-Subdivision

The Loop-Subdivision scheme is approximative. It is the most simple subdivision scheme. Even and odd vertices in regular triangle meshes are calculated with the following masks:

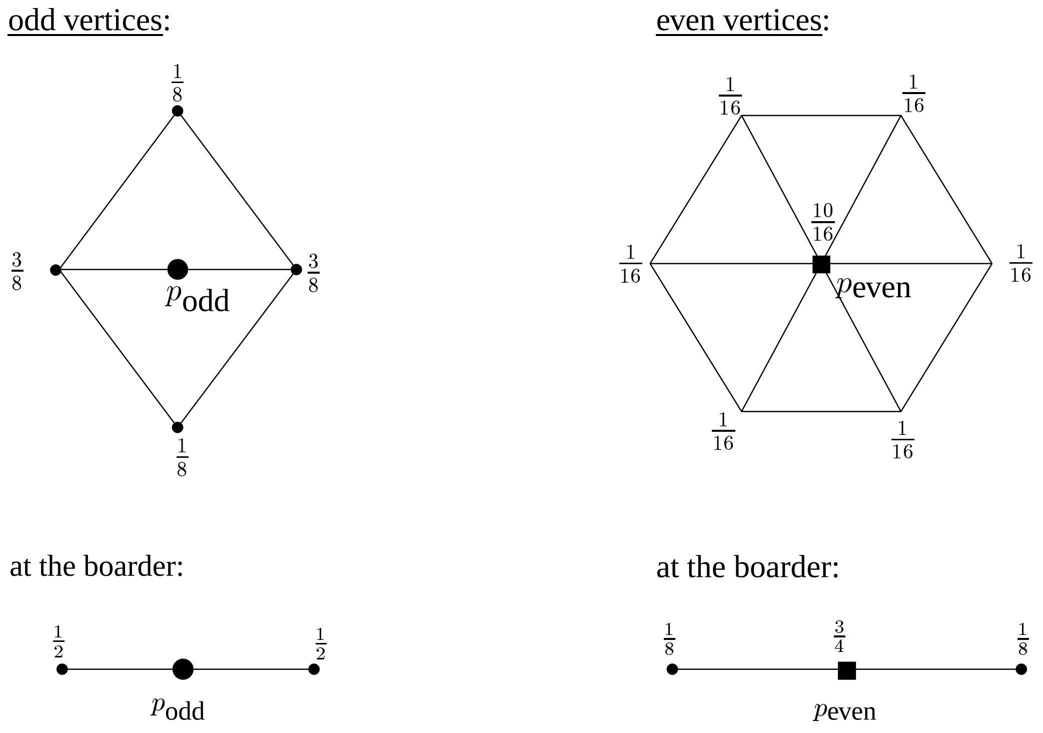
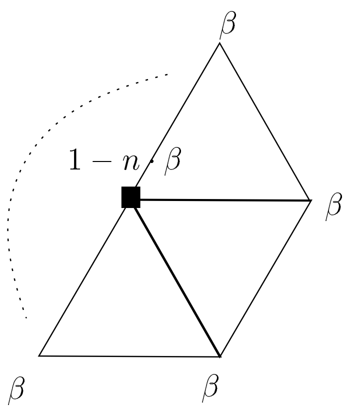


Figure 11.2: mask for the calculation of odd and even vertices

For irregular even vertices the Loop-Subdivision uses the scheme in Figure 11.3.



with n = valence of p_{even}

and $\beta = \frac{1}{n}\left(\frac{5}{8} - \left(\frac{3}{8} + \frac{1}{4}\cos\frac{2\pi}{n}\right)^2\right)$

Figure 11.3: masks for the calculation of irregular even vertices

At regular vertices the limit surface of the Loop-Subdivision scheme is C^2-continuous. At irregular vertices it is C^1-continuous. You can see a triangle mesh and three Loop-Subdivision steps in the Figures 11.4 und 11.5.

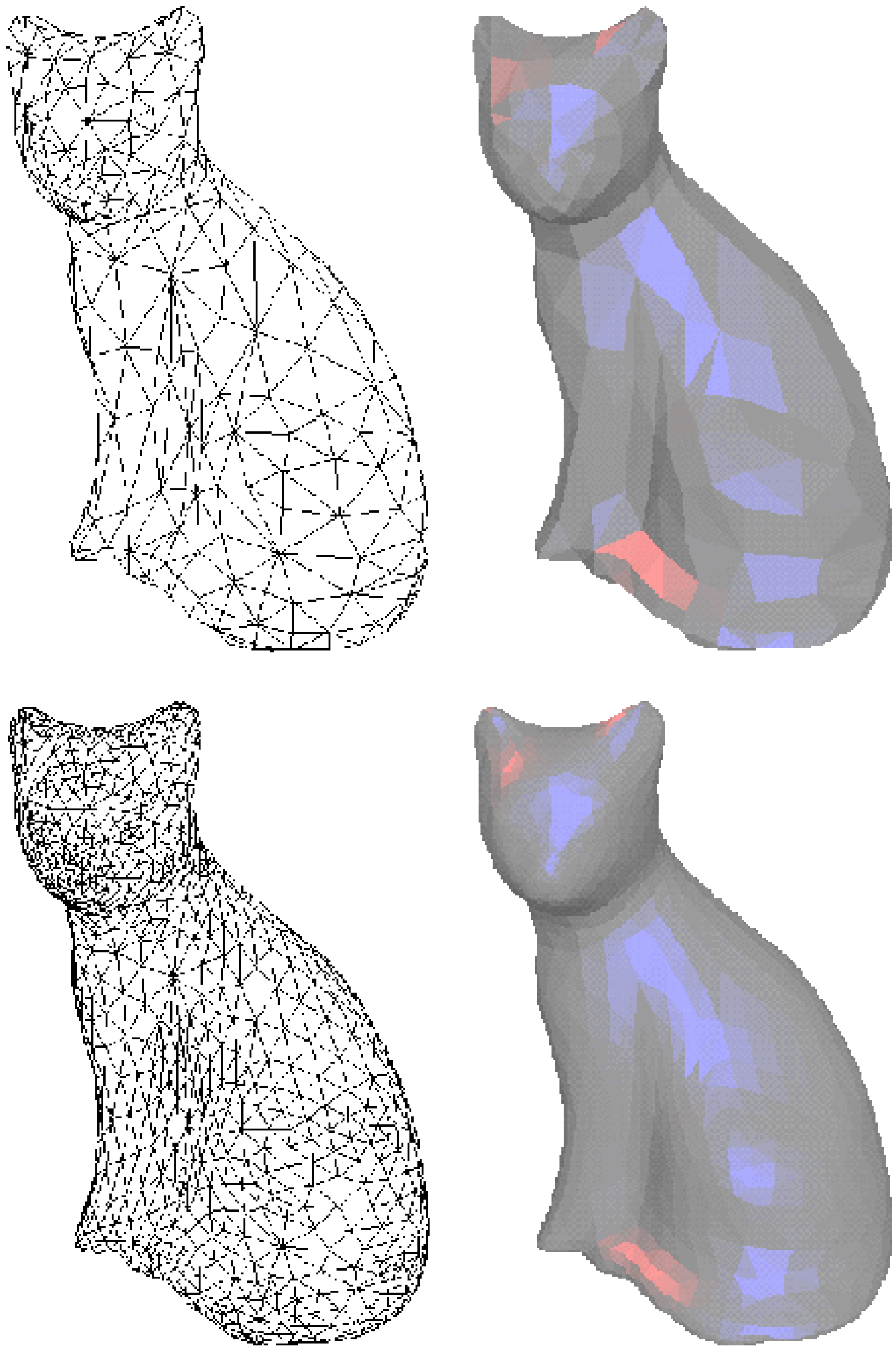


Figure 11.4: Loop-Subdivision: top: coarse triangle mesh, bottom: after 1 subdivision step

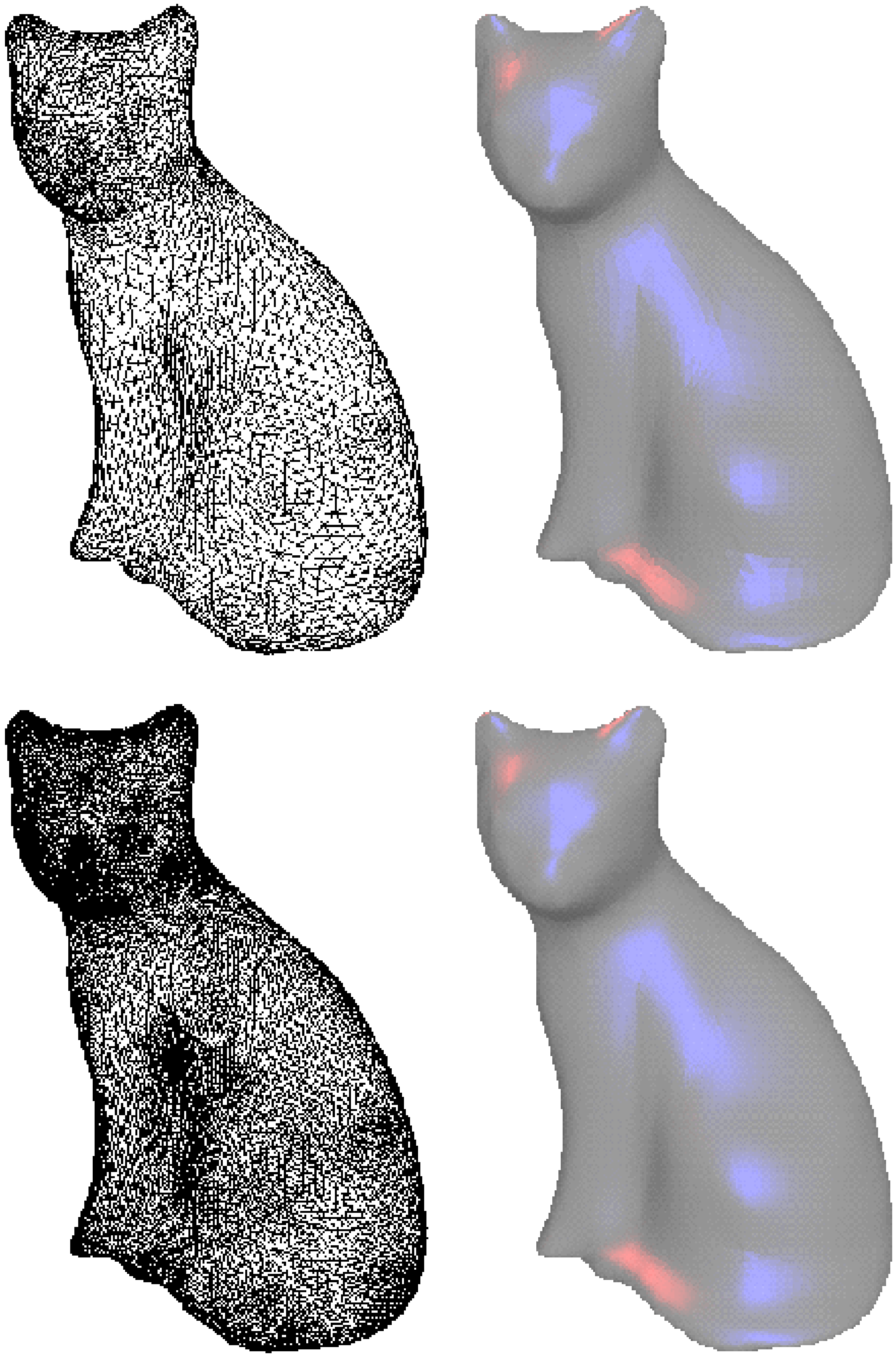


Figure 11.5: Loop-Subdivision: top: after 2 steps, bottom: after 3 steps

11.2 Interpolatory schemes: Modified Butterfly-Subdivision

Butterfly-Subdivsion is a interpolatory scheme. It reaches C^1-continuity of the limit surface. It only needs one rule for the calculation of the odd vertices, because the even vertices are interpolated.

Inner points:

(1) regular neighbour vertices

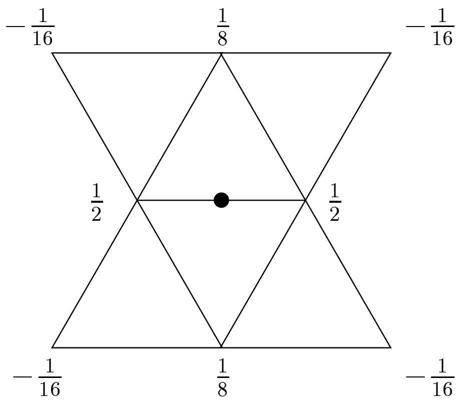
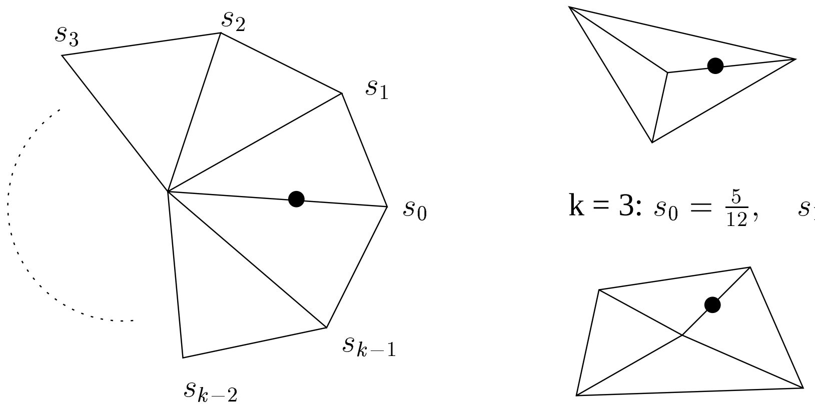


Figure 11.6: Butterfly: Mask for the calculation of the odd vertices

(2) one irregular neighbour vertice



k = 3: $s_0 = \frac{5}{12}, \quad s_{1,2} = -\frac{1}{12}$

k $\geq$ 5: $s_i = \frac{1}{k}\left(\frac{1}{4} + \cos\frac{2i\pi}{k} + \frac{1}{2}\cos 4i\pi k\right)$ k = 4: $s_0 = \frac{3}{8}, \quad s_2 = -\frac{1}{8}, \quad s_{1,3} = 0$

Figure 11.7: one irregular neighbour vertice with valence 3, 4 and 5

(3) two irregular neighbour vertices Use mask from (2) for both neighbours and average the results

Remark: Case 3 is only possible in the first subdivision step.

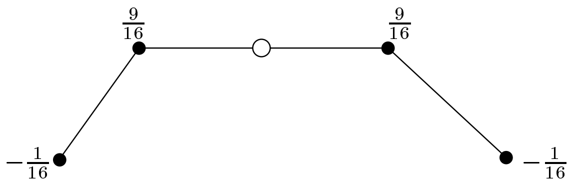


Figure 11.8: 4-point-scheme for boarder vertices

Boarder vertices: At the border the 4-point-scheme for curves is applied (Figure 11.8).
You can see a triangle mesh and three Butterfly-Subdivision steps in the Figures 11.9 und 11.10.

11.3 Limit points for approximative subdivision

In this section subdivision is written as a linear process. By analysing the so-called subdivision matrix the position of the vertices on the limit surface and propositions about the continuity of the limit surface can be deduced.

11.3.1 Limit points for the subdivision of cubic splines

The formulas for subdividing cubic splines from section 4.5.5 example 4 for open knot vectors are applied. These formulas for the calculation of the shifted vertices $v_{-1}^{k+1}, v_0^{k+1}, v_1^{k+1}$ can be displayed in matrix form as follows:

$$\begin{pmatrix} v_{-1}^{k+1} \\ v_0^{k+1} \\ v_1^{k+1} \end{pmatrix} = \underbrace{\begin{pmatrix} 1/2 & 1/2 & 0 \\ 1/8 & 3/4 & 1/8 \\ 0 & 1/2 & 1/2 \end{pmatrix}}_{\mathbf{S}} \begin{pmatrix} v_{-1}^{k} \\ v_0^{k} \\ v_1^{k} \end{pmatrix}$$

By repeated insertion we get:

$$\begin{pmatrix} v_{-1}^{k+1} \\ v_0^{k+1} \\ v_1^{k+1} \end{pmatrix} = \mathbf{S} \cdot \mathbf{S} \cdot \begin{pmatrix} v_{-1}^{k-1} \\ v_0^{k-1} \\ v_1^{k-1} \end{pmatrix} = \ldots = \mathbf{S^{k+1}} \cdot \begin{pmatrix} v_{-1}^{0} \\ v_0^{0} \\ v_1^{0} \end{pmatrix}$$

$$\begin{pmatrix} v_{-1}^{\infty} \\ v_0^{\infty} \\ v_1^{\infty} \end{pmatrix} = \lim_{k\to\infty} \mathbf{S^k} \begin{pmatrix} v_{-1}^{0} \\ v_0^{0} \\ v_1^{0} \end{pmatrix}$$

We have to know the matrix S^∞ for the calculation of the limit points. Therefore we have to diagonalize the subdivision matrix S:

$$S = R \cdot D \cdot R^{-1}$$

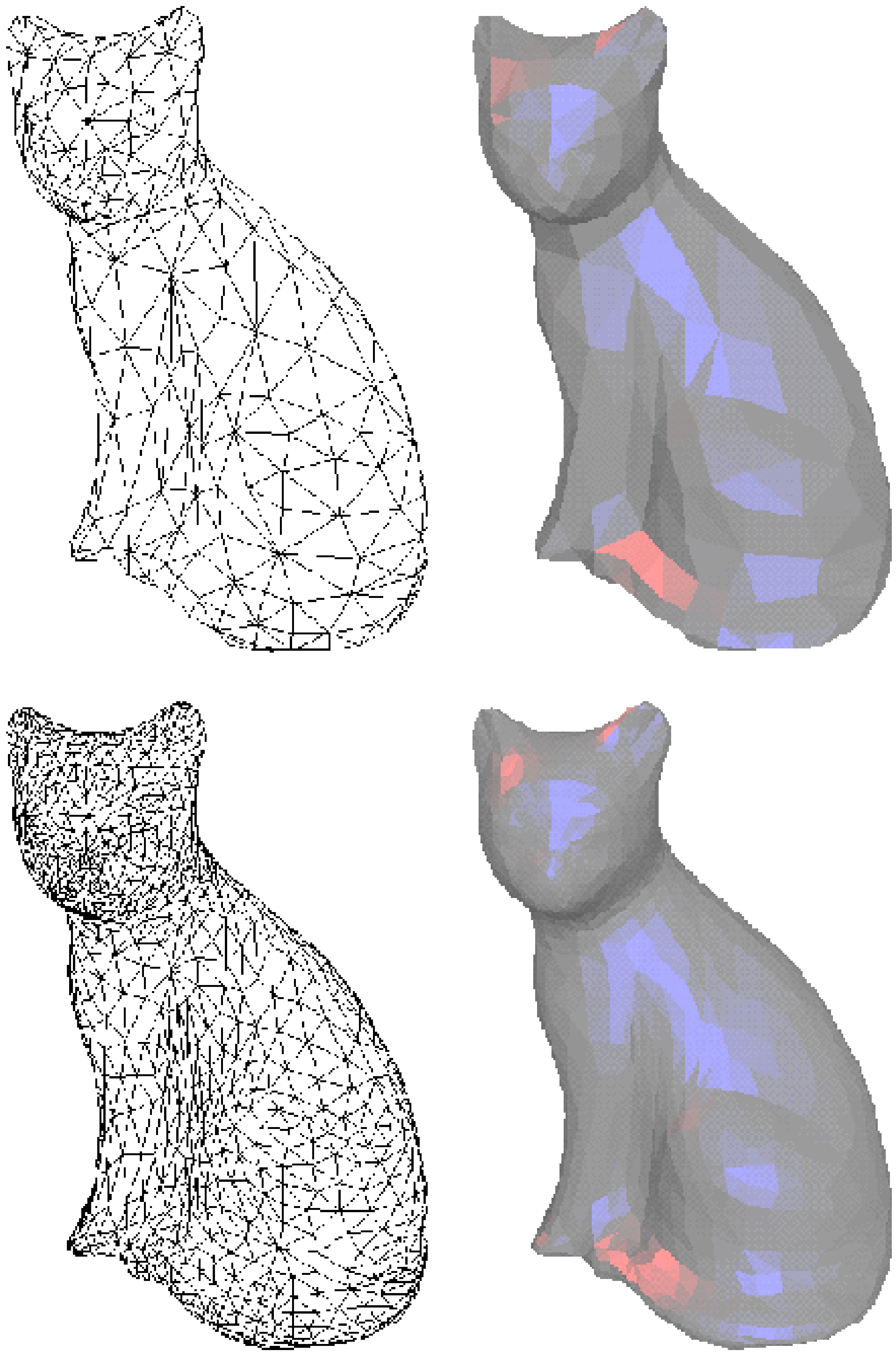


Figure 11.9: Butterfly-Subdivision: top: coarse triangle mesh, bottom: after 1 subdivision step

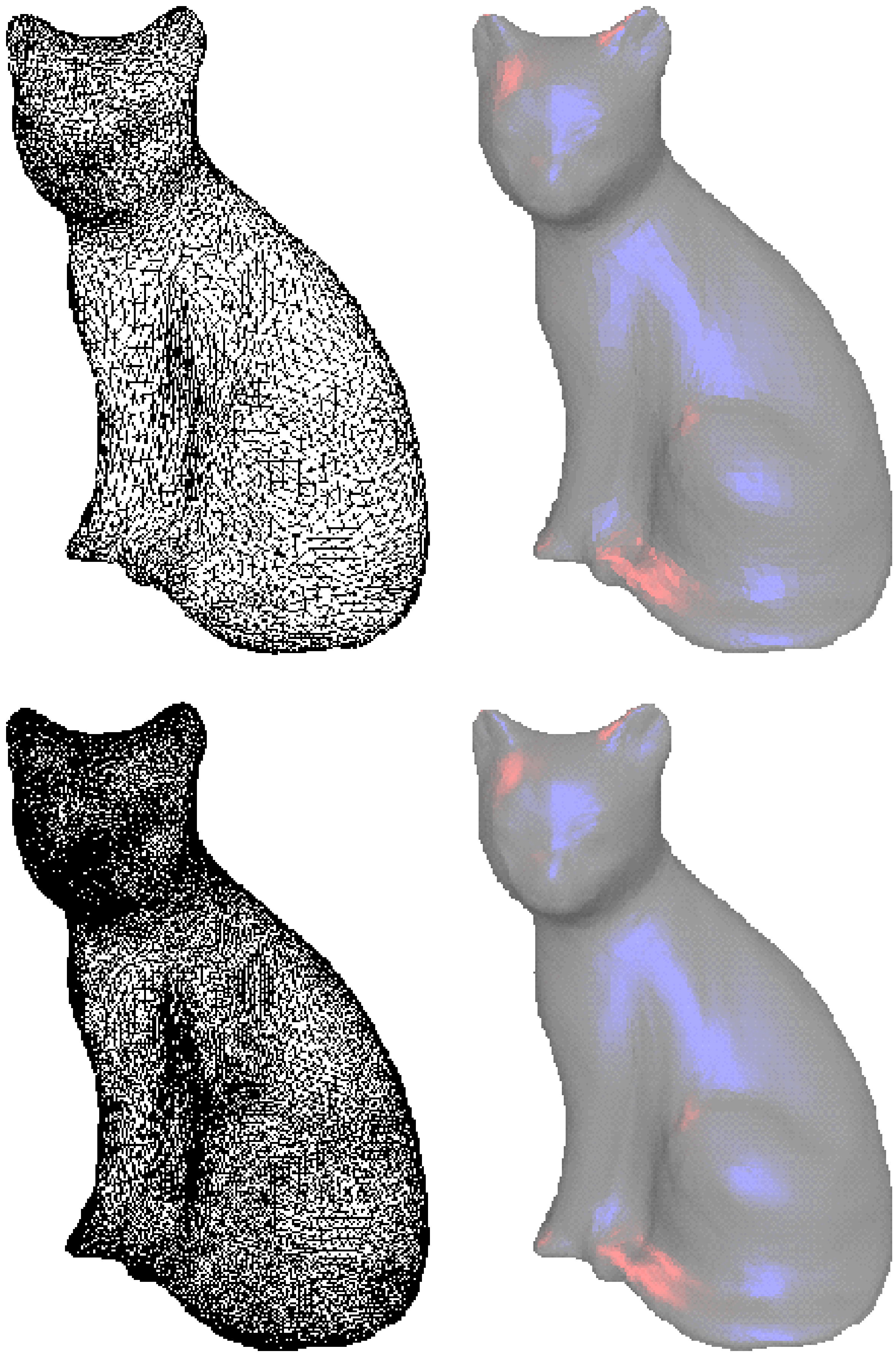


Figure 11.10: Butterfly-Subdivision: top: after 2 steps, bottom: after 3 steps

R: matrix consisting of eigenvectors
D: eigenvalues of the diagonal matrix

$$R = \begin{pmatrix} 1 & -1 & -2 \\ 1 & 0 & 1 \\ 1 & 1 & -2 \end{pmatrix} \quad D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{4} \end{pmatrix} \quad R^{-1} = \frac{1}{6}\begin{pmatrix} 1 & 4 & 1 \\ -3 & 0 & 3 \\ -1 & 2 & -1 \end{pmatrix}$$

$$\begin{aligned}
\Rightarrow\ S^k &= S \cdot \ldots \cdot S = (RD\underbrace{R^{-1}) \cdot (R}DR^{-1}) \cdot \ldots \cdot (RDR^{-1}) = RD^kR^{-1} \\
S^\infty &= RD^\infty R^{-1} = R \cdot (\lim_{k\to\infty} D^k) \cdot R^{-1} = \\
&= R \begin{pmatrix} \lim_{k\to\infty} 1^k & & \\ & \lim_{k\to\infty} \left(\frac{1}{2}\right)^k & \\ & & \lim_{k\to\infty} \left(\frac{1}{4}\right)^k \end{pmatrix} R^{-1} = \\
&= R \begin{pmatrix} 1 & & \\ & 0 & \\ & & 0 \end{pmatrix} R^{-1} = \frac{1}{6}\begin{pmatrix} 1 & 4 & 1 \\ 1 & 4 & 1 \\ 1 & 4 & 1 \end{pmatrix} \\
\Rightarrow v_0^\infty &= \frac{1}{6}v_{-1}^0 + \frac{2}{3}v_0^0 + \frac{1}{6}v_1^0
\end{aligned}$$

Remark: The subdivision matrix of good subdivision schemes for curves has the eigenvalues $1, \frac{1}{2}, \frac{1}{4}$.

11.3.2 Limit points for the subdivision of regular triangle meshes

Analogously to curves:

$$\begin{pmatrix} v_0^{k+1} \\ v_1^{k+1} \\ \vdots \\ \vdots \\ v_6^{k+1} \end{pmatrix} = \begin{pmatrix} \frac{10}{16} & \frac{1}{16} & \frac{1}{16} & \frac{1}{16} & \frac{1}{16} & \frac{1}{16} & \frac{1}{16} \\ \frac{3}{8} & \frac{3}{8} & \frac{1}{8} & & & & \frac{1}{8} \\ \frac{3}{8} & \frac{1}{8} & \frac{3}{8} & \frac{1}{8} & & & \\ \vdots & & \ddots & \ddots & \ddots & & \\ \frac{3}{8} & \frac{1}{8} & & & & \frac{1}{8} & \frac{3}{8} \end{pmatrix} \begin{pmatrix} v_0^k \\ v_1^k \\ \vdots \\ \vdots \\ v_6^k \end{pmatrix}$$

Subdivision matrix:

$$S = \frac{1}{16}\begin{pmatrix} 10 & 1 & 1 & 1 & 1 & 1 & 1 \\ 6 & 6 & 2 & 0 & 0 & 0 & 2 \\ 6 & 2 & 6 & 2 & 0 & 0 & 0 \\ 6 & 0 & 2 & 6 & 2 & 0 & 0 \\ 6 & 0 & 0 & 2 & 6 & 2 & 0 \\ 6 & 0 & 0 & 0 & 2 & 6 & 2 \\ 6 & 2 & 0 & 0 & 0 & 2 & 6 \end{pmatrix}$$

In the same way as for curves we diagonalize the subdivision matrix. The eigenvalues of the matrix are $1, \frac{1}{2}, \frac{1}{2}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{8}$.
In the limit we receive the following subdivision matrix and the limit point of v_0:

$$S^\infty = \frac{1}{12}\begin{pmatrix} 6 & 1 & 1 & 1 & 1 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 6 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$v_0^\infty = \frac{1}{2}v_0^0 + \sum_{j=1}^{6} \frac{1}{12} v_j^0$$

Remark: Eigenvalue criterias for the determination of the continuity of the limit surface for Eigenvalues $|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_n|$ of the subdivision matrix

- $\lambda_1 = 1 \quad \Rightarrow \quad C^0$-continuous
- $\lambda_2 = \lambda_3 = \frac{1}{2} \quad \Rightarrow \quad C^1$-continuous
- $\lambda_4 = \lambda_5 = \lambda_6 = \frac{1}{4} \quad \Rightarrow \quad C^2$-continuous

The analysis of the Loop scheme for irregular vertices and the Butterfly scheme for all vertices only yields C^1-continuity.