Computer Aided Geometric Design Compendium WS2023

Ida Hönigmann

December 3, 2023

Organization

Lecture each Thursday 12:00 to 14:00 (full 2 hours).
Oral exam. Write email to fix date and time.
Problem session each Thursday 14:00 to 16:00. Mandatory attendance!
Kreuzerlübung.

1 Bezier curves

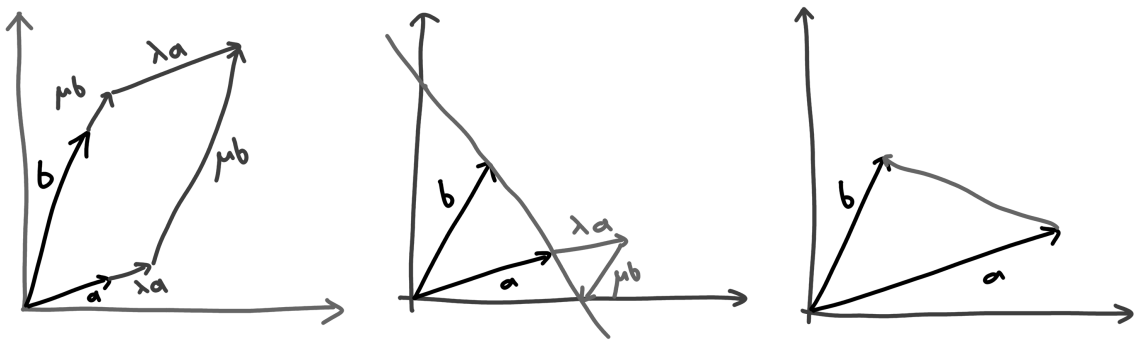

Figure 1: Linear combination, affine combination and convex combination

Example 1. ***Linear combination*** $\lambda a + \mu b$
Affine combination $\lambda a + \mu b$ *and* $\lambda + \mu = 1$
What is μ *so that* $\lambda a + \mu B$ *is on the line?*

$$\lambda a + \mu b = a + t(b - a) \implies a(\underbrace{\lambda - 1 + t}_{=0}) + b(\underbrace{\mu - t}_{=0}) = 0$$

If a, b *are linearly independent* $\implies \mu = t \wedge \lambda + \mu = 1$
Convex combination $\lambda a + \mu b$ *and* $\lambda + \mu = 1$ *and* $\lambda, \mu \geq 0$
Line is $a + t(b - a)$ *with* $t \in [0, 1] \implies \mu, \lambda \in [0, 1]$

Definition 1 (combinations)**.** *linear combination* $\sum_{i=1}^{n} \lambda_i v_i$ *with* $v_1, ..., v_n \in \mathbb{R}^d, \lambda_1, ..., \lambda_n \in \mathbb{R}$
affine combination $\sum_{i=1}^{n} \lambda_i v_i$ *with* $\sum_{i=1}^{n} \lambda_i = 1$
convex combination $\sum_{i=1}^{n} \lambda_i v_i$ *with* $\sum_{i=1}^{n} \lambda_i = 1$ *and* $\forall i : \lambda_i \geq 0$

Algorithm 1 (of de Casteljou, Bezier curve)**.** *Given:* $b_0, ..., b_n \in \mathbb{R}^d$ *(called control points / Kontrollpunkte),* $t \in \mathbb{R}$
Recursion: $b_i^0(t) := b_i$
$b_i^j(t) := (1 - t)b_i^{j-1}(t) + tb_{i+1}^{j-1}(t)$ *for* $j = 1, ..., n$ *and* $i = 0, ..., n - j$
Result: $b(t) := b_0^n(t)$ *(called Bezier curve)*

Remark 1. *In the algorithm above often we choose* $t \in [0, 1]$.

Example 2.

Remark 2. *In this course* $\mathbb{N} = \{1, 2, 3, ...\}$ *and* $\mathbb{N}_0 = \{0, 1, 2, 3, ...\}$

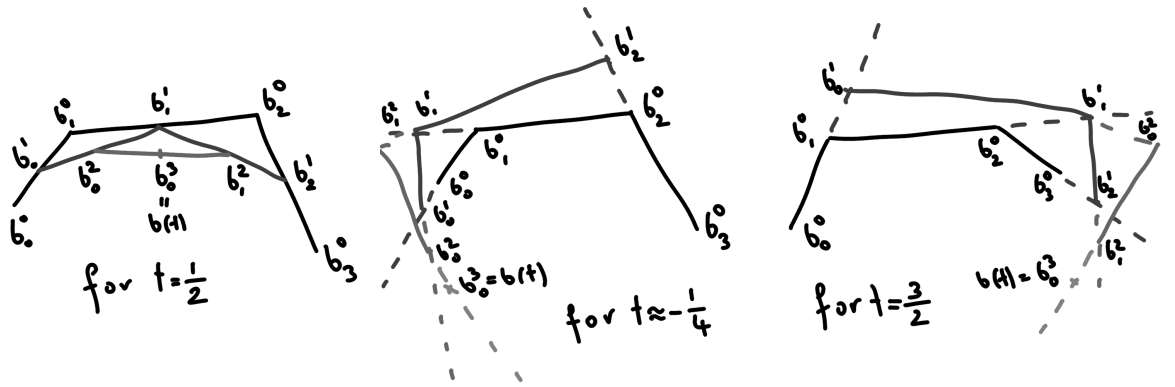

Figure 2: Examples of the de casteljou algorithm

Recap 1. $0! := 1, n! := n(n-1)(n-2)\cdots 1$ *for* $n \geq 1$.

$$\binom{n}{k} := \begin{cases} \frac{n!}{k!(n-k)!} & , n \geq k \geq 0 \\ 0 & , k > n \end{cases} \quad \text{for } n, k \in \mathbb{N}_0$$

Definition 2 (Bernstein polynomials)**.** *For* $n, i \in \mathbb{N}_0$ *we define* $B_i^n(t) := \binom{n}{i} t^i (1-t)^{n-i} \in \mathbb{R}[t]$

Remark 3. *Special cases of Bernstein polynomials*

$$i > n \implies B_i^n(t) = 0 \qquad B_i^n(0) = \begin{cases} 0, i \neq 0 \\ 1, i = 0 \end{cases}$$

$$B_i^n(1) = \begin{cases} 0, i \neq n \\ 1, i = n \end{cases} \qquad B_0^0(t) = 1$$

Theorem 1. $b_i^j(t) = \sum_{l=0}^{j} B_l^j(t) b_{i+l}$

Proof. Induction over j: $j = 0$:

$$j = 0: \qquad b_i^0(t) := b_i = 1 \cdot b_i = B_0^0(t) \cdot b_i \qquad \checkmark$$

$$j-1 \to j: \qquad b_i^j(t) := (1-t) b_i^{j-1}(t) + t b_{i+1}^{j-1}(t) \overset{\text{IA}}{=} (1-t) \sum_{l=0}^{j-1} B_l^{j-1}(t) b_{i+l} + t \sum_{l=0}^{j-1} B_l^{j-1}(t) b_{i+1+l} =$$

$$(1-t) \sum_{l=0}^{\cancel{j-1}\, j} B_l^{j-1}(t) b_{i+l} + t \sum_{\substack{l=1 \\ 0}}^{j} B_{l-1}^{j-1}(t) b_{i+l} = \sum_{l=0}^{j} (\underbrace{(1-t) B_l^{j-1}(t) + t B_{l-1}^{j-1}(t)}_{= B_l^j(t) \text{ using the following lemma}}) b_{i+l} =$$

$$\sum_{l=0}^{j} B_l^j(t) b_{i+l} \qquad \checkmark$$

□

Corollary 1. *The Bezier curve equals* $b(t) = b_0^n(t) = \sum_{l=0}^{n} B_l^j(t) b_{i+l}$*, which is called the Bernstein representation of the Bezier curve.*

Remark 4. *As* $b(t) = \sum_{l=0}^{n} B_l^n(t) b_l \in C^\infty$ *it is a polynomial curve of degree* n*, which is in* C^∞ *and therefore "very smooth".*

Lemma 1. $B_l^j(t) = (1-t) B_l^{j-1}(t) + t B_{l-1}^{j-1}(t)$

Proof.

$$(1-t) B_l^{j-1}(t) + t B_{l-1}^{j-1}(t) = (1-t) \binom{j-1}{l} t^l (1-t)^{j-1-l} + t \binom{j-1}{l-1} t^{l-1} (1-t)^{j-1-l+1} =$$

$$\binom{j-1}{l} t^l (1-t)^{j-l} + \binom{j-1}{l-1} t^l (1-t)^{j-l} = \left(\binom{j-1}{l} + \binom{j-1}{l-1} \right) t^l (1-t)^{j-l} = \binom{j}{l} t^l (1-t)^{j-l} = B_l^j(t)$$

□

Remark 5. *What is $b(0)$?* $b(0) = \sum_{i=0}^{n} B_i^n(0) b_i = b_0 + 0 + 0 + \cdots + 0 = b_0$
What is $b(1)$? $b(1) = \sum_{i=0}^{n} B_i^n(1) b_i = 0 + \cdots + 0 + b_n = b_n$

Definition 3 (end-point-interpolating)**.** *Curves which pass through the first and last point are called end-point-interpolating (Endpunktinterpolierend).*

Remark 6. *Bezier curves are end-point-interpolating.*

Remark 7. *How many intersection points are there between a planar (i.e. in $\mathbb{R}^2$) Bezier curve and a straight line?*

Straight line: $p + t(q - p)$ *Bezier curve:* $b(t) = \sum_{i=0}^{n} B_i^n(t) \underbrace{b_i}_{\in \mathbb{R}^2}$

Solving $p + t(q - p) = \sum_{i=0}^{n} B_i^n(t) b_i$ results in at most n solutions.

Lemma 2. $\frac{d}{dt} B_i^n(t) = n(B_{i-1}^{n-1}(t) - B_i^{n-1}(t))$

Proof.

$$\frac{d}{dt} B_i^n(t) = \frac{d}{dt} \binom{n}{i} t^i (1-t)^{n-i} = \binom{n}{i} i t^{i-1} (1-t)^{n-i} - \binom{n}{i} t^i (n-i)(1-t)^{n-i-1} =$$
$$\frac{n!}{\cancel{i!}_{(i-1)!}(n-i)!} \cancel{i} t^{i-1}(1-t)^{n-i} - \frac{n!}{i!\cancel{(n-i)!}_{(n-i-1)!}} \cancel{(n-i)} t^i (1-t)^{n-i-1} =$$
$$n\left(\frac{(n-1)!}{(i-1)!(n-i)!} t^{i-1}(1-t)^{n-i} - \frac{(n-1)!}{i!(n-i-1)!} t^i (1-t)^{n-i-1} \right) =$$
$$n\left(\binom{n-1}{i-1} t^{i-1}(1-t)^{n-i} - \binom{n-1}{i} t^i (1-t)^{n-i-1} \right) = n(B_{i-1}^{n-1}(t) - B_i^{n-1}(t))$$

□

Theorem 2. $\dot{b}(t) := \frac{d}{dt} b(t) = n \sum_{i=0}^{n-1} B_i^{n-1}(t)(b_{i+1} - b_i) = n(b_1^{n-1}(t) - b_0^{n-1}(t))$

Proof.

$$\dot{b}(t) = \frac{d}{dt}\left(\sum_{i=0}^{n} B_i^n(t) b_i \right) = \sum_{i=0}^{n} \frac{d}{dt} B_i^n(t) b_i = \sum_{i=0}^{n} n(B_{i-1}^{n-1}(t) - B_i^{n-1}(t)) b_i = n\left(\sum_{i=0}^{n} B_{i-1}^{n-1}(t) b_i - \sum_{i=0}^{n} B_i^{n-1}(t) b_i \right) =$$
$$n\left(\sum_{i=1}^{n} B_{i-1}^{n-1}(t) b_i - \sum_{i=0}^{n} B_i^{n-1}(t) b_i \right) = n\left(\underbrace{\sum_{i=0}^{n-1} B_i^{n-1}(t) b_{i+1}}_{=b_1^{n-1}(t)} - \underbrace{\sum_{i=0}^{n-1} B_i^{n-1}(t) b_i}_{=b_0^{n-1}(t)} \right) = n\left(\sum_{i=0}^{n-1} B_i^{n-1}(t)(b_{i+1} - b_i) \right)$$

□

Corollary 2.
- $\dot{b}(0) = n(b_1 - b_0)$
- $\dot{b}(1) = n(b_n - b_{n-1})$
- *The last segment in the algorithm of de Casteljou is the tangent of the Bezier curve in $b(t)$.*
- *The derivative of a bezier curve of degree n is a bezier curve of degree $n-1$ with control points $(b_1, b_0), (b_2 - b_1), \cdots, (b_n - b_{n-1})$.*

Corollary 3. $\ddot{b}(t) = n(n-1) \sum_{i=0}^{n-2} B_i^{n-2}(t)(b_{i+2} - 2b_{i+1} + b_i)$
$\ddot{b}(0) = n(n-1)(b_2 - 2b_1 + b_0)$, $\ddot{b}(1) = n(n-1)(b_n - 2b_{n-1} + b_{n-2})$

Corollary 4. *The curvature of a bezier curve in the point $b(0)$ depends only on b_0, b_1, b_2.*
The curvature of a bezier curve in the point $b(1)$ depends only on b_{n-2}, b_{n-1}, b_n.

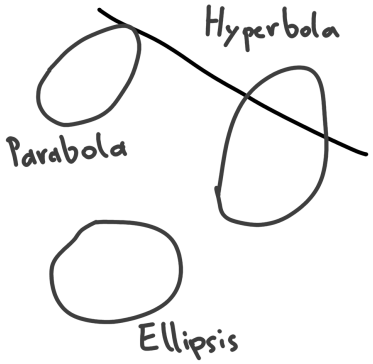

Figure 3: Categorization of Parabolas, Hyperbolas and Ellipsis as intersection points with the line at infinity.

Example 3. *Quadratic Bezier curve*

$$b(t) = \sum_{i=0}^{2} B_i^2(t) b_i = \binom{2}{0} t^0 (1-t)^2 b_0 + \binom{2}{1} t^1 (1-t)^1 b_1 + \binom{2}{2} t^2 (1-t)^0 b_2 = t^2(b_2 - 2b_1 + b_0) + t(2b_1 - 2b_0) + b_0$$

which is an affine transformation of a parabola and therefore a parabola.
Quadratic bezier curves are parabolas.

Remark 8. *Line at infinity (Ferngerade) is the collection of points where parallel lines intersect.*

Remark 9. *Different applications using these curves are Rhino, OpenSCAD, Autocad, Geogebra, ...*

2 Parameterized curves

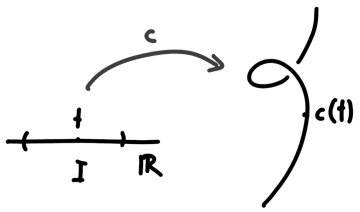

Figure 4: parameterized curve $c(t)$

Definition 4. *$c : I \subseteq \mathbb{R} \to \mathbb{R}^3$ is called a parameterized curve.*
$\dot{c}(t) := \frac{d}{dt} c(t)$ is called the tangential vector. For $\mathbb{R}^3$ we have $\dot{c}(t) = (\dot{c_1}(t), \dot{c_2}(t), \dot{c_3}(t))$.
The velocity is defined as $||\dot{c}(t)||$.
A point $c(t)$ is called regular, if $\dot{c}(t) \neq 0$ and is called singular, if $\dot{c}(t) = 0$.

Example 4. *A helix (Schraublinie) is defined by $c(t) = (\cos(t), \sin(t), t)^T$.*

$$\dot{c}(t) = (-\sin(t), \cos(t), 1)^T \qquad ||\dot{c}(t)|| = \sqrt{\sin^2(t) + \cos^2(t) + 1} = \sqrt{2}$$

We see that the helix is passed through with constant velocity. Furthermore all points are regular.

Example 5. *$c : \mathbb{R} \to \mathbb{R}^3, t \mapsto (t^2, t^3, t^4)$, $\dot{c}(t) = (2t, 3t^2, 4t^3)$. We see that 0 is singular as $\dot{c}(0) = (0, 0, 0)$. Everywhere else the curve is regular.*

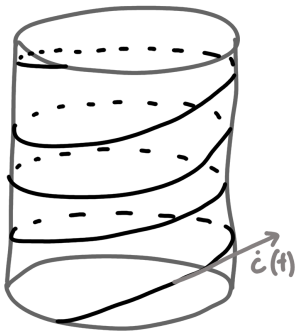

Figure 5: Helix with tangential vector

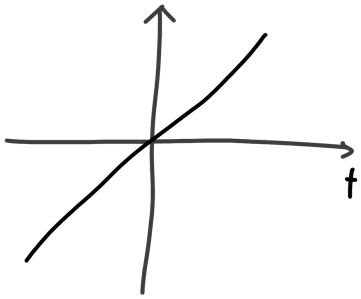

Figure 6: identity line can be parameterized such that $(0,0)$ is singular.

Remark 10. *A point being regular or singular depends on the parameterisation of the curve.*
For example $c(t) = (t,t)$ produces a regular curve, while $c(t) = (t^3, t^3)$ produces a curve where 0 is singular.
There are curves and points where no parameterisation exists such that the point is regular.

Definition 5. $c : I \to \mathbb{R}^2 \in C^2(I, \mathbb{R}^2)$
The curvature of the curve in the point $c(t)$ is defined as $\kappa(t) = \frac{\det(\dot{c}(t), \ddot{c}(t))}{||\dot{c}(t)||^3}$

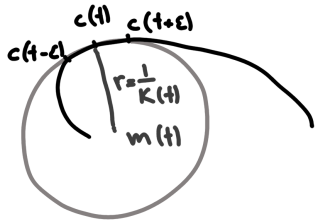

Figure 7: Circle of curvature

Example 6. *The circle of curvature has a radius of $\frac{1}{\kappa(t)}$. $m(t)$ is called the center of curvature.*
$m(t) = c(t) + \frac{1}{\kappa(t)} n(t)$ where $n(t) = \frac{(-\dot{c_2}(t), \dot{c_1}(t))}{||\dot{c}(t)||}$.

Remark 11. *Exercise: compare this definition of curvature with the school version concerning graphs.*

Example 7. *For a circle we have $c(t) = (r\cos(t), r\sin(t))^T$, $\dot{c}(t) = (-r\sin(t), r\cos(t))^T$, $\ddot{c}(t) = (-r\cos(t), -r\sin(t))^T$*

$$\kappa(t) = \frac{\det \begin{pmatrix} -r\sin(t) & -r\cos(t) \\ r\cos(t) & -r\sin(t) \end{pmatrix}}{r^3} = \frac{r^2\sin^2(t) + r^2\cos^2(t)}{r^3} = \frac{r^2}{r^3} = \frac{1}{r}$$
$$n(t) = \frac{(-r\cos(t), -r\sin(t))}{r} = (-\cos(t), -\sin(t))$$

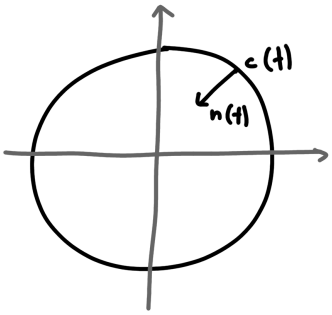

Figure 8: Circle with normal vector

Definition 6. *A point $c(t)$ with $\dot{\kappa}(t) = 0$ is called a vertex.*

Example 8. *An ellipse has four vertices.*

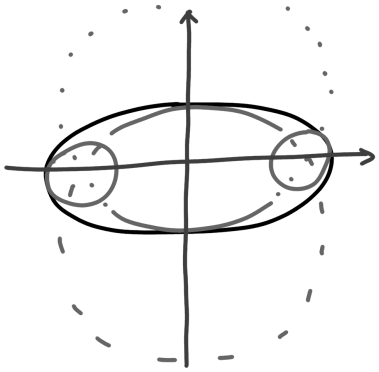

Figure 9: Ellipse and the four vertices.

$(t, \exp t)$ has no vertex.
Klothoids are curves with $\kappa(t) = t$. They are used in road construction and have no vertex.

Definition 7. $c : I \to \mathbb{R}^3$
$\kappa(t) = \frac{||\dot{c}(t) \times \ddot{c}(t)||}{||\dot{c}(t)||^3}$ *is called the curvature of a space curve.*
$\tau(t) = \frac{\det(\dot{c}(t), \ddot{c}(t), \dddot{c}(t))}{||\dot{c}(t) \times \ddot{c}(t)||^2}$ *is called torsion of a space curve.*

Example 9. *For the helix $t \mapsto (\cos(t), \sin(t), pt)$ the torsion depends on p.*

3 Properties of Bezier curves

Definition 8. *$\alpha : \mathbb{R}^n \to \mathbb{R}^m$ is called **affine** if $\exists l : \mathbb{R}^n \to \mathbb{R}^m$ ...linear $\exists v \in \mathbb{R}^m : \alpha(x) = l(x) + v$.*
*α is called **affinity** if α is affine and bijective.*

Example 10. *An example of an linear function is shear (Scherung).*

$$l \begin{pmatrix} x \\ y \end{pmatrix} := \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Area is preserved.

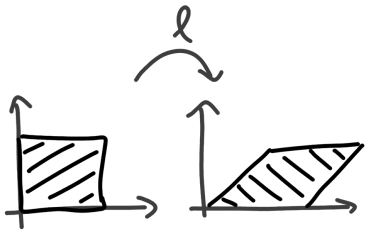

Figure 10: Shear preserves area

Theorem 3. *Bezier curves are invariant under affine transformations.*

Proof. Let $b(t) = \sum_{i=0}^{n} B_i^n(t) b_i$ be a bezier curve and $\alpha(x) = l(x) + v$ where $b_i \in \mathbb{R}^d, l : \mathbb{R}^d \to \mathbb{R}^m, v \in \mathbb{R}^m$

$$\alpha(b(t)) = l(b(t)) + v = l\left(\sum_{i=0}^{n} B_i^n(t) b_i\right) + v = \sum_{i=0}^{n} B_i^n(t) l(b_i) + v =$$
$$\sum_{i=0}^{n} B_i^n(t) l(b_i) + \sum_{i=0}^{n} B_i^n(t) v = \sum_{i=0}^{n} B_i^n(t)(l(b_i) + v) = \sum_{i=0}^{n} B_i^n(t) \alpha(b_i)$$

□

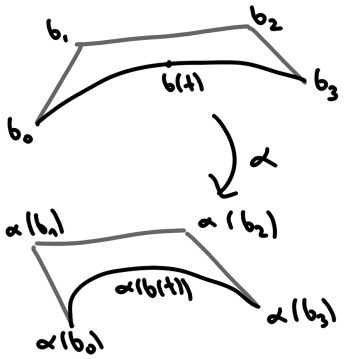

Figure 11: Bezier curve is invariant under affine transformation

Lemma 3. $\sum_{i=0}^{n} B_i^n(t) = 1$

Proof. Binomial theorem: $(x+y)^n = \sum_{i=0}^{n} \binom{n}{i} x^i y^{n-i}$ for $x = t, y = 1 - t$ we get

$$1 = (t + (1-t))^n = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} = \sum_{i=0}^{n} B_i^n(t)$$

□

Lemma 4. *For $t \in [0,1]$ it holds that $B_i^n(t) \geq 0$*

Proof.

$$B_i^n(t) = \underbrace{\binom{n}{i}}_{\geq 0} \underbrace{t^i}_{\geq 0} \underbrace{(1-t)^{n-i}}_{\geq 0} \geq 0$$

□

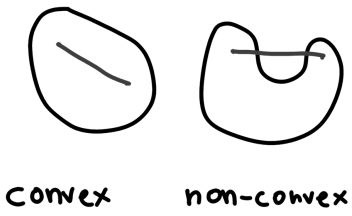

Figure 12: example of convex and non-convex set

Definition 9. *$M \subseteq \mathbb{R}^d$ then $conv(M) := \{\sum_{i=1}^k \lambda_i v_i : v_1, ..., v_k \in M, \lambda_1, ..., \lambda_k \geq 0, \sum_{i=1}^k \lambda_i = 1, k \in \mathbb{N}\}$ is called the convex hull of M.*
M is called convex set, iff $\forall x, y \in M \forall t \in [0,1] : tx + (1-t)y \in M$.

Example 11.

Remark 12 (Pick theorem)**.** *For any polygon formed of points in $\mathbb{Z} \times \mathbb{Z}$ it holds that the area can be calculated by $I + R/2 - 1$ where I is the number of points inside the polygon and R is the number of points on the edges of the polygon.*

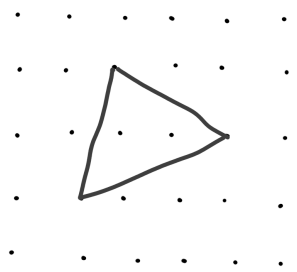

Figure 13: example picks theorem where $I = 2, R = 3$ and therefore $A = \frac{5}{2}$

Theorem 4. $\forall t \in [0,1] : b(t) \in conv(\{b_0, b_1, ..., b_n\})$

Proof. $b(t) = \sum_{i=0}^n \underbrace{B_i^n(t)}_{\geq 0} b_i \forall t \in [0,1]$ and $\sum_{i=0}^n B_i^n(t) = 1 \forall t \in [0,1]$. Therefore $b(t) \in conv(\{b_0, ..., b_n\})$ □

Remark 13 (Montecarlo method for calculating area)**.** *random points, count how many fall within the area from the total amount of points, estimate volume from that.*

Theorem 5. *Bezier curves are symmetric, meaning that if $b(t)$ is a bezier curves of $b_0, ..., b_n$ and $\tilde{b}(t)$ is a bezier curve of $b_n, ..., b_0$ then $b(t) = \tilde{b}(1-t)$*

Proof.

$$\tilde{b}(t) = \sum_{i=0}^n B_i^n(t) b_{n-i} = \sum_{i=0}^n \underbrace{B_i^n(1-t)(1-t)}_{B_i^n(t)} b_{n-i} = \sum_{i=0}^n B_{n-i}^n(1-t) b_i = \sum_{i=0}^n B_i^n(1-t) b_i = b(1-t)$$

□

Lemma 5. $\alpha \in \mathbb{R} \implies B_i^n(\alpha t) = \sum_{j=0}^n B_i^j(\alpha) B_j^n(t)$

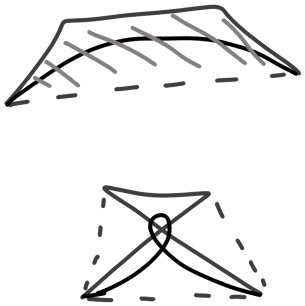

Figure 14: example of a bezier curve and its convex hull

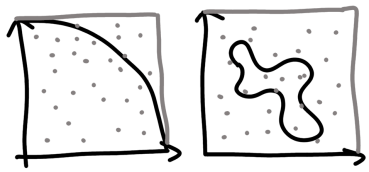

Figure 15: montecarlo method for a quarter circle and some random shape

Proof. can be done by induction, but is tedious □

Theorem 6 (Sub-division-property (Unterteilungseigenschaft))**.** *Let $b(t)$ be a bezier curve to $b_0, ..., b_n$ and $\alpha \in \mathbb{R}$.*

$$\tilde{b}(t) := b(\alpha t) \hat{b}(t) := b((1-\alpha)t + \alpha)$$

Then $\tilde{b}$ is a bezier curve to $b_0^0(\alpha), b_0^1(\alpha), ..., b_0^n(\alpha)$; $\hat{b}$ is a bezier curve to $b_0^n(\alpha), b_1^{n-i}(\alpha), ..., b_n^0(\alpha)$ and "glued together" they result in the original bezier curve.

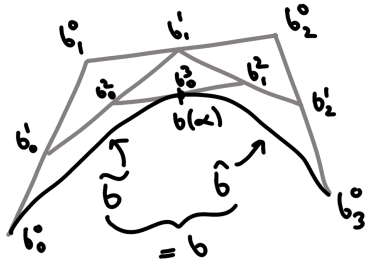

Figure 16: bezier curve b is divided at point $b(\alpha)$ into two bezier curves $\tilde{b}$ and $\hat{b}$

Proof.

$$\tilde{b}(t) = b(\alpha t) = \sum_{i=0}^{n} B_i^n(\alpha t) b_i = \sum_{i=0}^{n} \left(\sum_{j=0}^{n} B_i^j(\alpha) B_j^n(t) \right) b_i = \sum_{j=0}^{n} \sum_{i=0}^{n} B_i^j(\alpha) B_j^n(t) b_i =$$

$$\sum_{j=0}^{n} B_j^n(t) \underbrace{\sum_{i=0}^{\not{n}^j} B_i^j(\alpha) b_i}_{b_0^j(\alpha)} = \sum_{j=0}^{n} B_j^n(t) b_0^j(\alpha)$$

Where we used $b_i^j(t) = \sum_{l=0}^{j} B_l^j(t) b_{i+l}$, which we already showed.

$\hat{b}$ is analogous. □

Definition 10. *The polynomial ring is defined as* $\Pi_n := \{f \in \mathbb{R}[t] : deg(f) \leq n\}$*, which is a vector space.*

Theorem 7. $\{B_0^n(t), ..., B_n^n(t)\}$ *is a basis of* Π_n*.*

Proof. $\forall i : B_i^n(t) \in \Pi_n$ therefore $span(B_0^n(t), ..., B_n^n(t)) \subseteq \Pi_n$.

We know that $dim(\Pi_n) = n + 1$ as $\{1, t, t^2, ..., t^n\}$ is a basis.

Let $k \leq n$.

$$1 = \sum_{i=0}^{n-k} B_i^{n-k}(t) = \sum_{i=k}^{n} B_{i-k}^{n-k}(t) = \sum_{i=k}^{n} \binom{n-k}{i-k} t^{i-k} (1-t)^{n-i}$$

$$\implies \binom{n}{k} t^k = \sum_{i=k}^{n} \binom{n}{k} \binom{n-k}{i-k} t^i (1-t)^{n-i} = \sum_{i=k}^{n} \binom{n}{i} \binom{i}{k} t^i (1-t)^{n-i} = \sum_{i=k}^{n} \binom{i}{k} B_i^n(t)$$

$$\implies t^k \in span(B_0^n(t), ..., B_n^n(t)) \implies \Pi_n \subseteq span(B_0^n(t), ..., B_n^n(t))$$

□

Corollary 5. *Every polynomial curve is a bezier curve.*

Theorem 8 (corner-cutting (Eckenabschneiden))**.** *Define the sequence* P_n *by this (recursive) definition* $P_0 := (b_0^0, b_0^1, ..., b_0^n)$*,* $P_1 := (b_0^0, b_0^1, ..., b_0^n, b_1^{n-1}, b_2^{n-2}, ..., b_n^0)$*, ...,* P_n *consists of the points we get by sub-dividing the bezier curve (with* $t = \frac{1}{2}$*).*

Then this sequence "converges" to the bezier curve

Proof. $d := \max_{0 \leq i \leq n} ||b_i - b_{i+1}$ ... maximum length of edges of control polygon

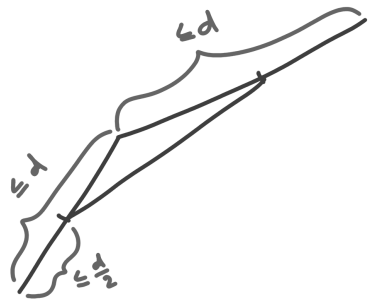

Figure 17: the new edges are at most $\frac{d}{2}$ long

From the image we can deduce that the maximum length of the edges of the new control polygon is at most $\frac{d}{2}$. With induction we get that the control polygon of P_n has the maximum edge length of $\frac{d}{2^n}$.

How far are the points of the control polygon from the bezier curve b? $||P_k(i) - b|| \leq (n-1)\frac{d}{2^k} \to 0$ for $k \to \infty$. □

Remark 14. *Not only does the sequence from the theorem converge pointwise, but it holds that* $P_n \to b$*.*

Remark 15. *Disadvantages of bezier curves are: unintuitive, local changes are not possible as every change in a control point changes the entire curve, points can be far away from the curve.*

4 B-Spline-Curves

Remark 16. *B-Spline curves are compositions of bezier curves.*

Figure 18: continuous, tangential continuous and curvature continuous composition

Example 12.

Definition 11 (B-Spline-Curves)**.** *$m \in \mathbb{N}$ is the number of control points, $n \in \mathbb{N}$ is the degree of the b spline curve, $T = (t_0, t_1, ..., t_{m+n+1})$ where $t_i \in \mathbb{R}$, $t_i \leq t_{i+1}$ and $t_i < t_{i+n+1}$ is called knot vector (Knotenvektor).*

$$\alpha_i^r(t) := \begin{cases} \frac{t-t_i}{t_{i+r}-t_i} & t_{i+r} - t_i \neq 0 \\ 0 & else \end{cases}$$

$$N_i^0(t) := \begin{cases} 1 & t \in [t_i, t_{i+1}) \\ 0 & else \end{cases} \qquad N_i^r(t) := \alpha_i^r(t) N_i^{r-1}(t) + (1 - \alpha_{i+1}^r(t)) N_{i+1}^{r-1}(t)$$

$N_i^r : \mathbb{R} \to \mathbb{R}$ are called b-spline basis functions (B-Spline-Basisfunktionen) and are piece-wise polynomial curves.

Theorem 9. *Every b-spline curve $s(t)$ can be expressed in the form $s(t) = \sum_{i=0}^{m} N_i^n(t) c_i$ where $c_0, ..., c_m \in \mathbb{R}^d$ are the control points and n is the degree of the b-spline curve.*

Proof. without proof □

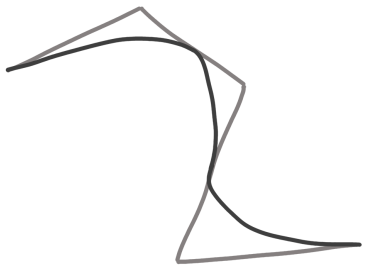

Figure 19: from this set of control points there are different possible quadratic b-spline curves (depending on the chosen knot vector).

Example 13.

Remark 17. *Typically a knot vector of a similar form as $T = (0, 0, 0, 0, 1, 2, 3, 4, 5, 5, 5, 5)$ is chosen. 0 often repeated, then counting up, then as many numbers of the same value as 0 in the start.*

Example 14. *Every line segment in the control polygon is cut into thirds.*

Example 15. *closed b-spline curves*

Remark 18. *B-spline curves of degree n with $n+1$ control points ($\implies m = n$) are bezier curves.*

Figure 20: cubic b-spline curve

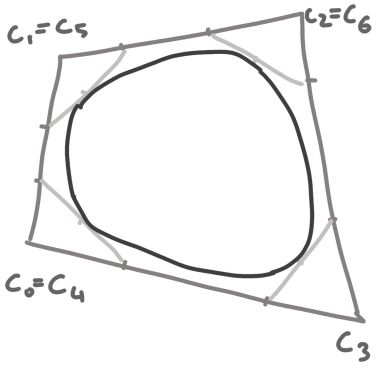

Figure 21: curvature continuous, closed b-spline curve.

5 NURBS-Curve

Remark 19. *Non-Uniform-Rational-B-Spline-Curve*

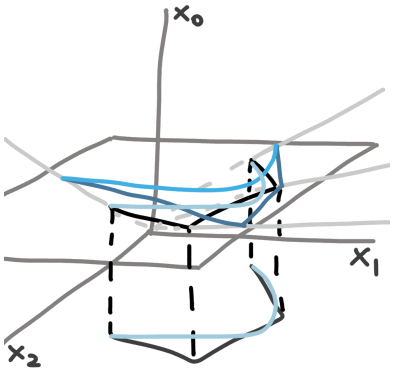

Figure 22: construction of a nurbs curve

Lemma 6 (Construction of a NURBS curve). *1. given are control points $c_0, ..., c_m \in \mathbb{R}^d$*

2. $d_i := (1, c_i) \in \mathbb{R}^{d+1}$ (this is called transforming into homogeneous coordinates (homogenisieren))

3. choose weights $w_i > 0$. $e_i := w_i d_i \in \mathbb{R}^{d+1}$

4. calculate b-spline curve with control points e_i

$$s(t) := \sum_{i=0}^{m} N_i^n(t) e_i = \sum_{i=0}^{m} N_i^n(t) w_i (1, c_i) = \left(\sum_{i=0}^{m} N_i^n(t) w_i, \sum_{i=0}^{m} N_i^n(t) w_i c_i \right)$$

5. project $s(t)$ back into the $(1 \times \mathbb{R}^d)$ plane

$$\tilde{x}(t) = pr(s(t)) = \left(1, \frac{\sum_{i=0}^{m} N_i^n(t) w_i c_i}{\sum_{i=0}^{m} N_i^n(t) w_i} \right)$$

6. dehomogenate

$$x(t) := pr_{\mathbb{R}^d}(\tilde{x}) = \frac{\sum_{i=0}^{m} N_i^n(t) w_i c_i}{\sum_{i=0}^{m} N_i^n(t) w_i}$$

$x(t)$ is called the NURBS curve with weights $w_0, .., w_m$

Remark 20. *In the special case $w_i = w \in \mathbb{R}^+$ the NURBS curve is a B-spline curve.*

Remark 21.

$$\sum_{i=0}^{m} N_i^n(t) = 1 \qquad\qquad N_i^n(t) \geq 0$$

Remark 22. *The bigger w_i is the more the curve is attracted towards c_i.*

Remark 23. *NURBS and B-splines are invariant under affine transformations.*

6 Subdivision algorithms

6.1 Subdivision for curves

Algorithm 2 (Chaikin). *given some polygon $p_i, p_{i+1}, ... \in \mathbb{R}^d$*

1. copy every vertex $p_i \to p_i^1$

2. calculate average of two neighboring $p_i^1 s$ $m_i^1 = \frac{1}{2}(p_i^1 + p_{i+1}^1)$

3. average the $m_i^1 s$ $p_i^2 = \frac{1}{2}(m_i^1 + m_{i+1}^1)$

The resulting polygon looks similar to the original one, but the corners are rounded. Continue by repeating the procedure with the new polygon as input.

Theorem 10. *The algorithm of Chaikin gives polygons that converge to a quadratic B-Spline curve.*

Remark 24. *Remember Chaikin as: copy once, average twice*

Algorithm 3 (Lane-Riesenfeld). *same as Chaikin, but copy once, average n times*

Theorem 11. *The algorithm of Lane-Riesenfeld gives polygons that converge to a B-Spline curve of degree n. Lane-Riesenfeld is:*

- *a approximating scheme*
- *is affine invariant.*

Remark 25. *Chaikin is the $n = 2$ special case of Lane-Riesenfeld.*

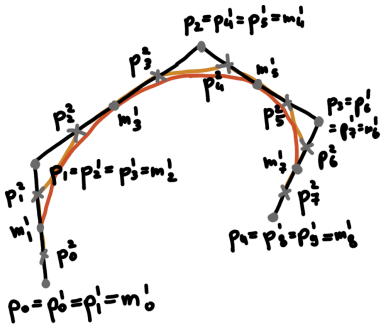

Figure 23: Chaikin algorithm for five given points.

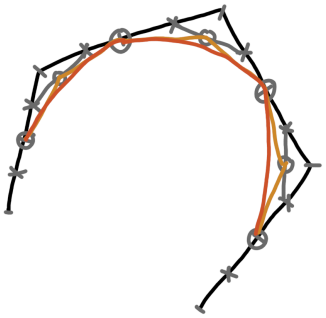

Figure 24: Lane Riesenfeld algorithm with $n = 3$ for five given points.

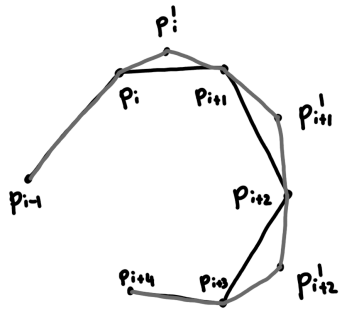

Figure 25: Four Point Scheme for six given points.

Algorithm 4 (Four-Point-Scheme)**.**

$$p_i^1 := -\frac{1}{16}p_{i-1} + \frac{9}{16}p_i + \frac{9}{16}p_{i+1} - \frac{1}{16}p_{i+2}$$

this is an affine combination. Inductive $p_i^2, ...$

Theorem 12. *The curve resulting from the four point scheme converge to a* C^1 *curve. Four Point Scheme is:*

- *a interpolating*
- *is affine invariant.*

Remark 26. *The weights of the four point scheme can be chosen differently as well:*

$$p_i^1 := -wp_{i-1} + (\frac{1}{2} + w)p_i + (\frac{1}{2}w)p_{i+1} - wp_{i+2}$$

Above we choose $w = \frac{1}{16}$. This is always interpolating, but only for $w \in (0, \frac{\sqrt{5}-1}{8})$ the resulting curve will be C^1.

Remark 27. *All of the above schemes preserve sub spaces and therefore have the property of linear precision (lineare Präzesion).*

They do not have "circular precision", meaning that if all points lie on a sphere then the curve can leave the sphere.

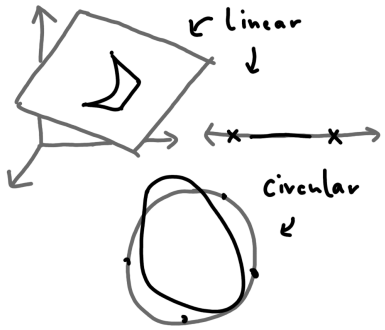

Figure 26: Example for linear precision, but not circular precision.

6.2 Subdivision for meshes

Definition 12. *The degree (Valenz) of a vertex is the number of edges connected to the vertex.*

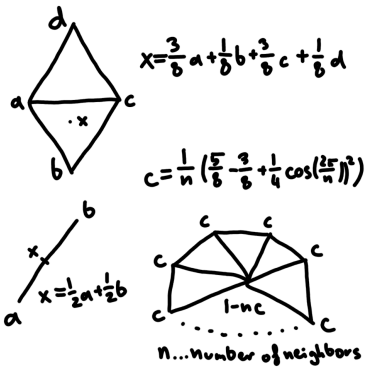

Figure 27: Loop subdivision.

Algorithm 5 (Loop (triangle meshes))**.**

Theorem 13. *Loop subdivision*

- *is approximating*
- *converges to a C^2 surface for generic $n = 6$ vertices and to a C^1 surface otherwise*
- *is face splitting (faces are subdivided)*

Algorithm 6 (Modified-Butterfly-Scheme (triangle meshes))**.** *As in the image we generate new vertices from edges. For non-generic $k \neq 6$ we specify*

$$s_j = \frac{1}{k}\left(\frac{1}{4} + \cos\left(\frac{2j\pi}{k}\right) + \frac{1}{2}\cos\left(\frac{4j\pi}{k}\right)\right)$$

Theorem 14. *The Modified Butterfly Scheme:*

- *is interpolating*

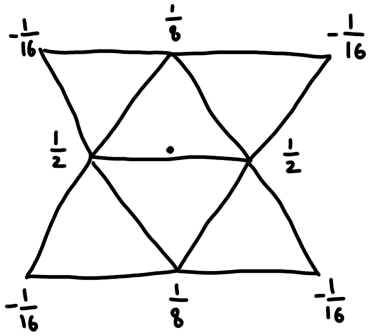

Figure 28: Modified Butterfly subdivision.

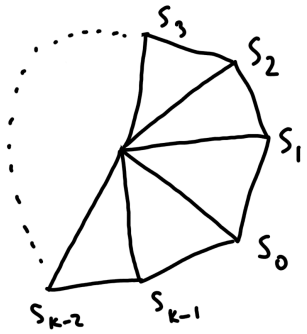

Figure 29: Special case in Modified Butterfly subdivision.

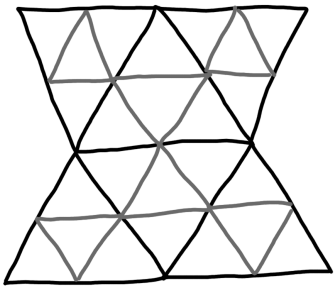

Figure 30: Result of Modified Butterfly subdivision.

- *converges to a C^1 surface*
- *is face-splitting*

Algorithm 7 (Catmull-Clark (square meshes))**.**

Theorem 15. *Catmull Clark is:*

- *approximating*
- *converges to a C^2 surface for generic vertices*
- *is face-splitting*

Algorithm 8 (Doo-Sabin)**.**

$$c_j = \frac{\left(3 + 2\cos\left(\frac{2\pi j}{k}\right)\right)}{4k}, j > 0 \qquad\qquad c_0 = \frac{1}{4} + \frac{5}{4k}$$

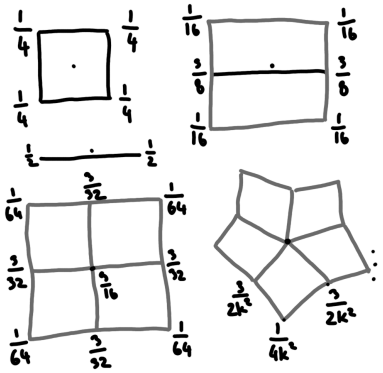

Figure 31: Catmul Clark subdivision.

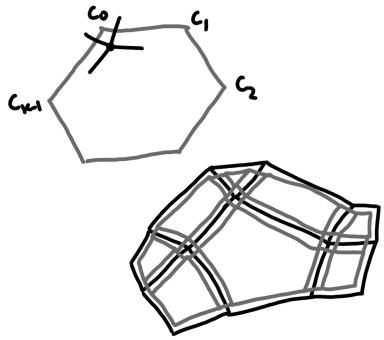

Figure 32: Doo Sabin subdivision.

Theorem 16. *Doo-Sabin is:*

- *approximating*
- *converges to C^1 surface*
- *is vertex-splitting.*

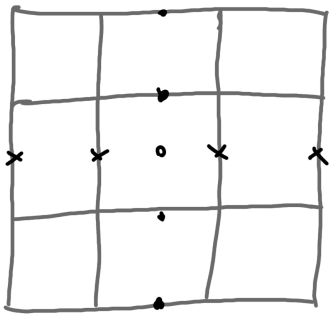

Figure 33: Kobbelt subdivision.

Algorithm 9 (Kobbelt-Scheme (square meshes))**.** *Apply four-point-scheme for vertical and horizontal vertexes. Then apply four point scheme on resulting points. The "horizontal" result and the "vertical" result are the same.*

Theorem 17. *The Kobbelt scheme is:*

- *interpolating*
- *converges to a C^1 surface*

- *is face-splitting.*

Algorithm 10 (Half-Edge-Data-Structure)**.** *given: mesh, that represents a orientable surface.*

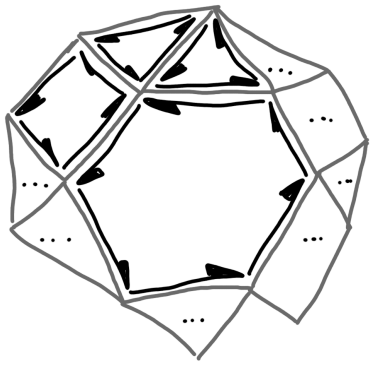

Figure 34: Halfedge datastructure.

A mesh is a collection of lists. These lists include vertices v_i, edges e_i, faces f_i and half-edges h_i.

Every vertex v is assigned a half-edge `v->h`*. Every face, every edge is assigned a half-edge. Every half-edge is assigned the opposing (*`flip`*), next (*`next`*), previous (*`prev`*) half-edge, the corresponding face (*`f`*), vertex (*`v`*) and edge (*`e`*).*

Remark 28. *In python use the package* `geopy`*. Find it via tiss/Lehrunterlagen. The documentation is at* `www.geometrie.tuwien.ac.at/kilian/docs/geopy`*.*