UiO : Department of Mathematics
University of Oslo

Adaptive plotting of splines

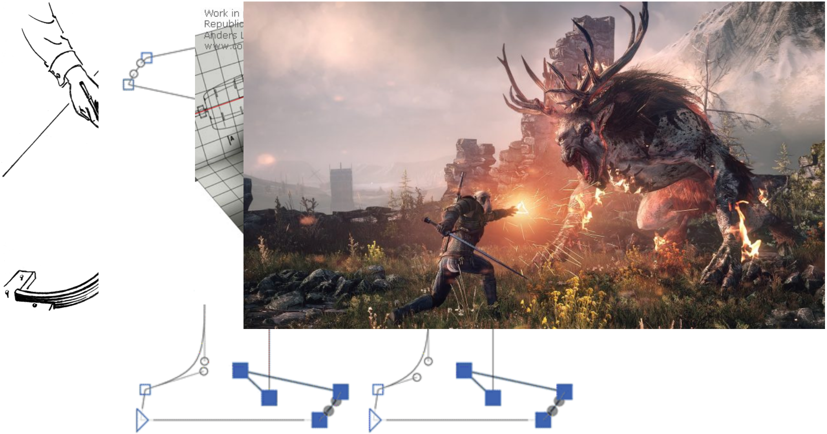
Master's thesis, 30 credits
Supervisor: Knut Mørken

Stein Dahl

February 19, 2020

Splines? Computer graphics?

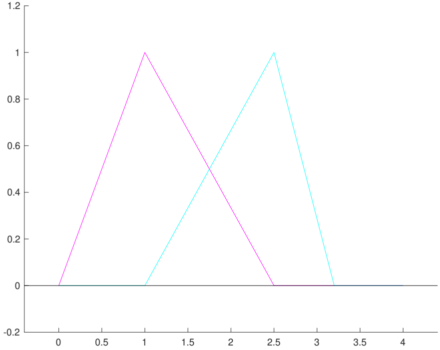
1
2
3
4



[1]https://en.wikipedia.org/wiki/Flat_spline.png
[2]https://glyphsapp.com/blog/new-features-in-glyphs-2-5
[3]https://www.rcuniverse.com/forum/.../gid=1&pid=4
[4]https://www.pcgamesn.com/15-best-rpgs-pc

Contents

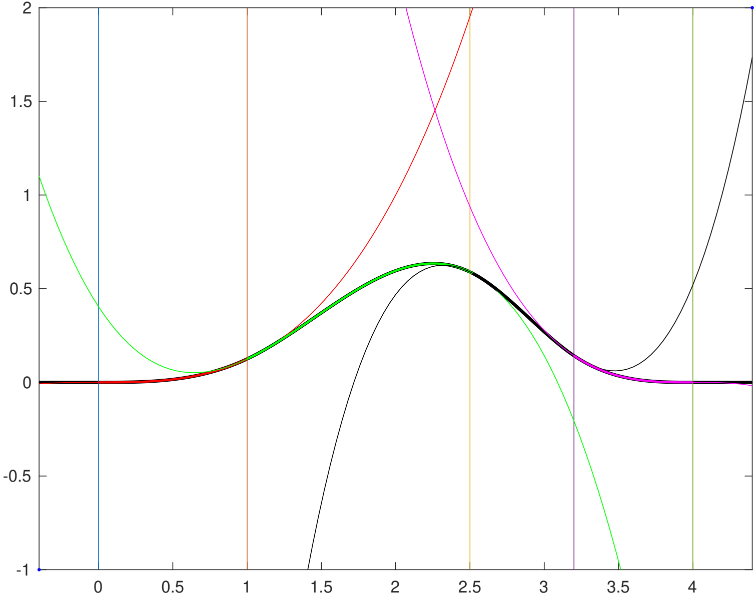
B-splines



$$B_{j,0,\mathbf{t}}(x) = \begin{cases} 1 & \text{if} \quad t_j \leq x < t_{j+1} \\ 0 & \text{otherwise.} \end{cases}$$

Knot vector: $\mathbf{t} = \{t_j\}$

$$B_{j,p,\mathbf{t}}(x) = \frac{x - t_j}{t_{j+p} - t_j} B_{j,p-1,\mathbf{t}} + \frac{t_{j+p+1} - x}{t_{j+p+1} - t_{j+1}} B_{j+1,p-1,\mathbf{t}}$$



2
1.5
1
0.5
0
-0.5
-1
0
0.5
1
1.5
2
2.5
3
3.5
4

Properties

1. Local knots. *The jth B-spline only depends on the knots* $t_j, t_{j+1}, \dots, t_{j+p+1}$.
2. Local support. *If x is outside of the interval* $\left[t_j, t_{j+p+1}\right)$*, then* $B_j(x) = 0$.
3. Active B-splines. *If x lies in the interval* $\left[t_\mu, t_{\mu+1}\right)$*, then* $B_j(x) = 0$ *for* $j < \mu - p$ *and* $j > \mu$.
4. Positivity. *If x is in* (t_j, t_{j+p+1})*, then* $B_j(x) > 0$.
5. Partition of unity. *If x is in* $[t_\mu, t_{\mu+1})$*, then* $\sum_{j=\mu-p}^{\mu} B_j(x) = 1$.
6. Special values. *If* $x = t_{\mu+1} = \cdots = t_{\mu+p} < t_{\mu+p+1}$*, then* $B_\mu(x) = 1$ *and* $B_j(x) = 0$ *for* $j \neq \mu$.

Spline space

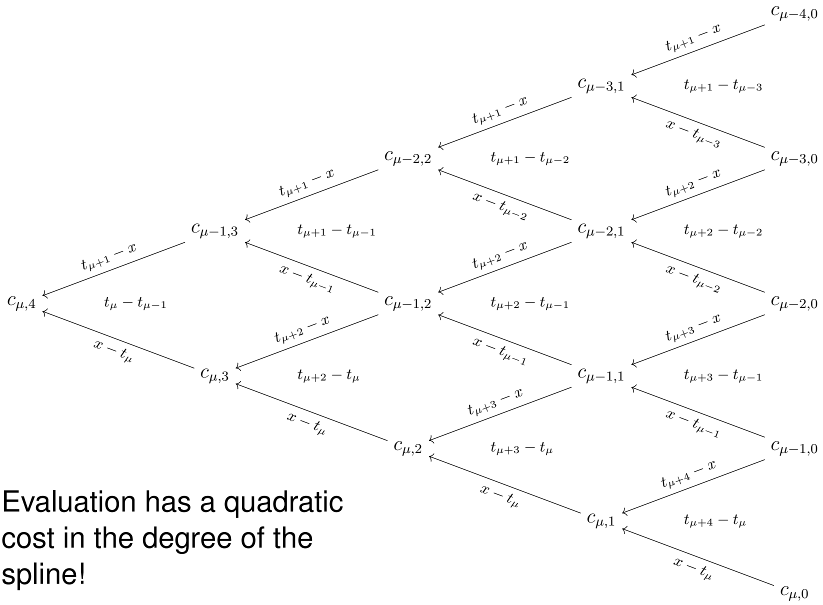
Definition (Spline space and B-spline coefficients)

Let $\mathbf{t} = \{t_i\}_{i=1}^{n+p+1}$ be a knot vector for n B-splines. Let $s \geq 1$ be an integer. The space of all linear combinations of these B-splines of degree p is the spline space $\mathbb{S}^s_{p,\mathbf{t}}$ defined by

$$\mathbb{S}^s_{p,\mathbf{t}} = \left\{ \sum_{j=1}^{n} \mathbf{c}_j B_{j,p} \mid \mathbf{c}_j \in \mathbb{R}^s \textit{ for } \quad 1 \leq j \leq n \right\}. \tag{1}$$

The coefficients $(\mathbf{c}_j)_{j=1}^n$ are called the *B-spline coefficients* of f.

Evaluation



Evaluation has a quadratic cost in the degree of the spline!

Control Polygon

Definition (Knot averages)

Let $\mathbf{t} = \{t_i\}_{i=1}^{n+p+1}$ be a knot vector. The vector $\mathbf{t}^* = \{t_j^*\}_{j=1}^{n}$ holds the *knot averages*, where

$$t_j^* = \frac{1}{p}(t_{j+1} + \ldots + t_{j+p}).$$

Definition (Control points, control polygon)

The *control points* $\{\mathbf{P}_j\}_{j=1}^{n}$ of the spline f are given by

$$\mathbf{P}_j = \begin{cases} (t_j^*, c_j) & \text{if} \quad s = 1 \\ \mathbf{c}_j & \text{if} \quad s = 2. \end{cases}$$

Knot insertion

Definition (Knot refinement, change of basis)

Refine $\mathbf{t} = \{t_i\}_{i=1}^{n_0+p+1}$ into $\bar{\mathbf{t}} = \{\bar{t}_j\}_{j=1}^{n_1+p+1}$.

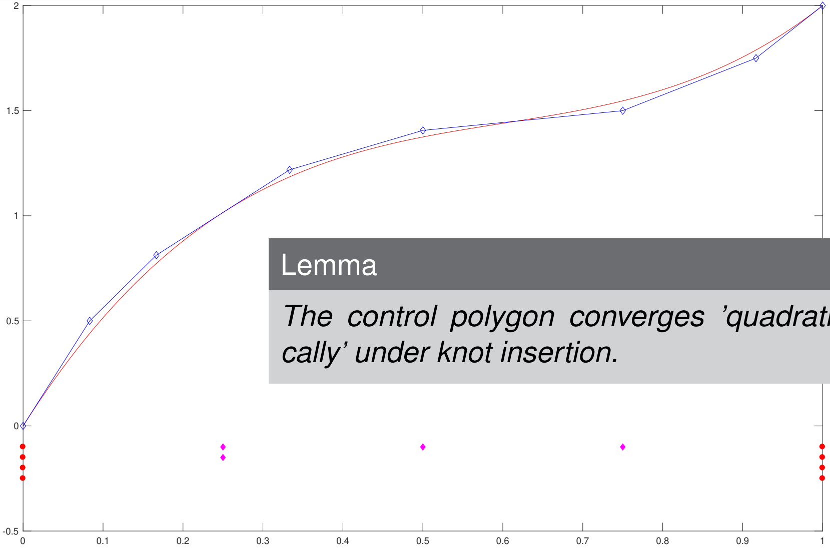
We can now write $f = \sum_{i=1}^{n_0} c_i B_{i,p,\mathbf{t}} = \sum_{j=1}^{n_1} \bar{c}_j B_{j,p,\bar{\mathbf{t}}}$.

Lemma (Böhm's method)

Insert a knot z in $\mathbf{t}$ in the interval $[t_\mu, t_{\mu+1})$. Then

$$\bar{c}_j = \begin{cases} c_j & \text{if } 1 \le j \le \mu - p, \\ \frac{z-t_j}{t_{j+p}-t_j} c_j + \frac{t_{j+p}-z}{t_{j+p}-t_j} c_{j-1} & \text{if } \mu - p + 1 \le j \le \mu, \\ c_{j-1} & \text{if } \mu + 1 \le j \le n+1. \end{cases}$$

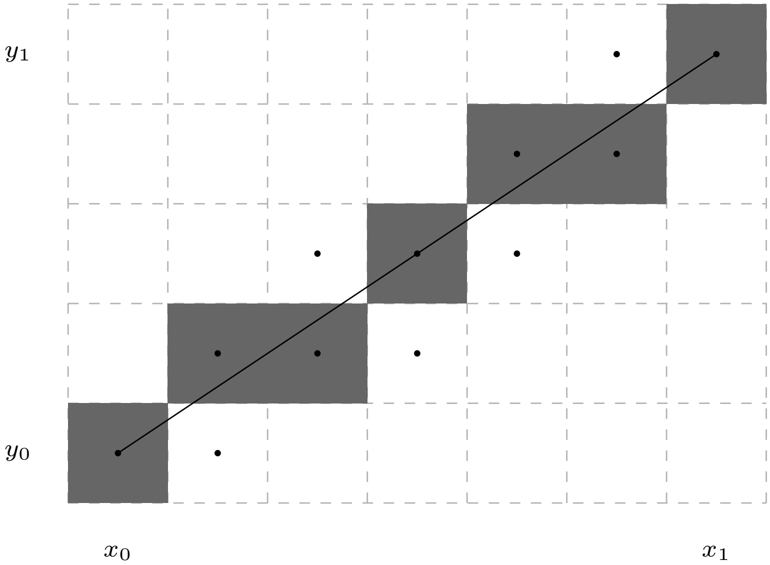
CP convergence under knot insertion



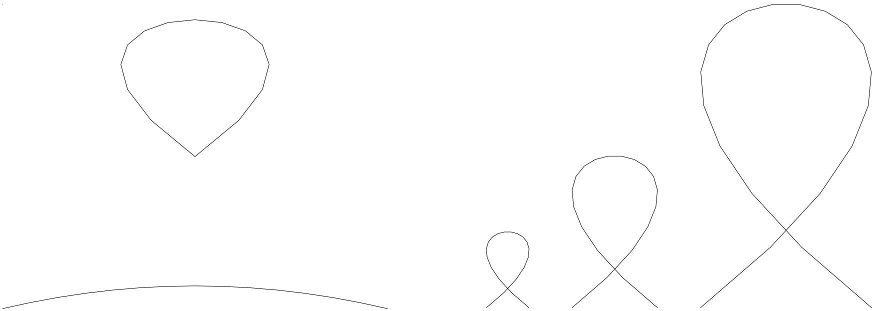
Lemma

The control polygon converges 'quadratically' under knot insertion.

Bresenham's line algorithm



Visual quality: Arc length and curvature



Uniform sampling

1. Sample in N points of the parameter interval $[a, b]$.
   Global uniform sampling: sample values x_i are given by

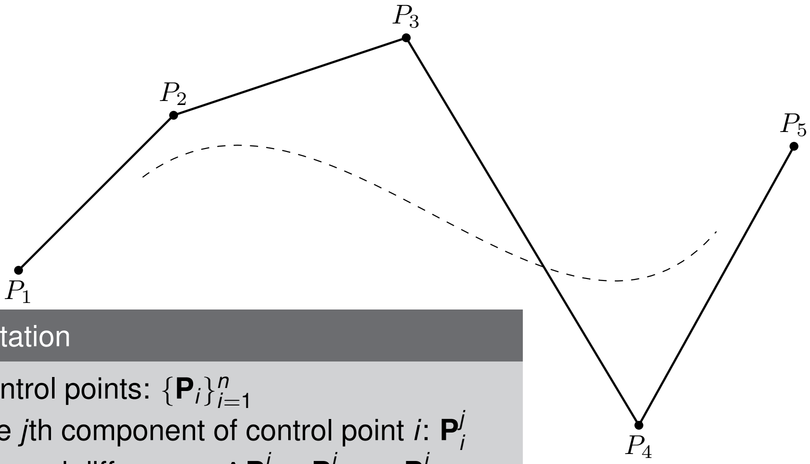
$$x_i = a + (i-1)h, \qquad h = \frac{b-a}{N-1}.$$

2. Draw lines between the sample points

Adaptive sampling: Wolfram Mathematica

1. The function is evaluated at N_0 uniformly spaced values.
2. The angle between successive line segments is examined.
   - If above threshold α_τ: Recursive subdivision of that part.
3. The recursion continues until the angle criterion or the recursion depth limit is reached.

Control polygon obtained - what now?



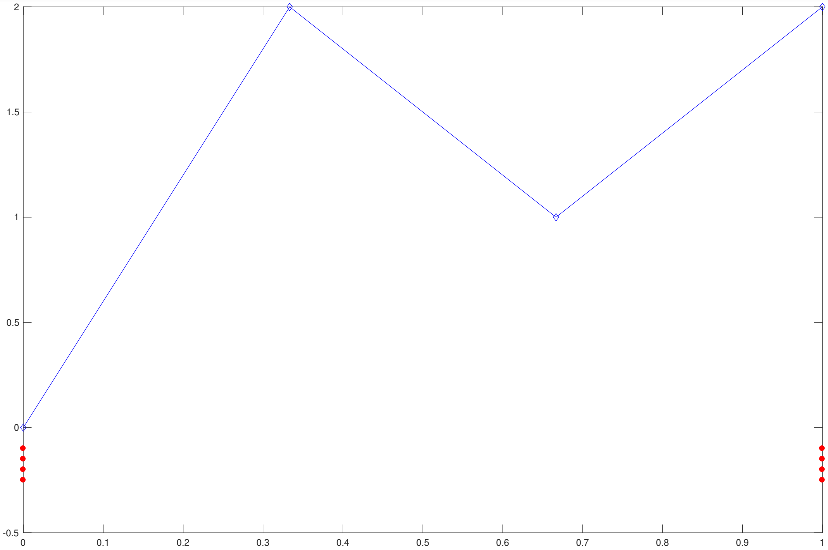
Notation

Control points: $\{\mathbf{P}_i\}_{i=1}^n$

The jth component of control point i: $\mathbf{P}_i^j$

Forward difference: $\Delta\mathbf{P}_i^j = \mathbf{P}_{i+1}^j - \mathbf{P}_i^j$.

Example reviewed

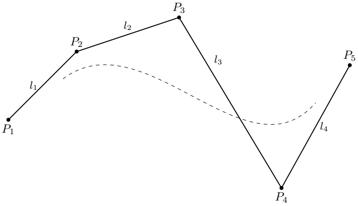


I: Length methods

Definition

$$l_i = \sqrt{(\Delta \mathbf{P}_i^1)^2 + (\Delta \mathbf{P}_i^2)^2}$$

$$\tilde{l}_i = \max\left\{\Delta \mathbf{P}_i^1, \Delta \mathbf{P}_i^2\right\} + \frac{1}{2}\min\left\{\Delta \mathbf{P}_i^1, \Delta \mathbf{P}_i^2\right\}$$



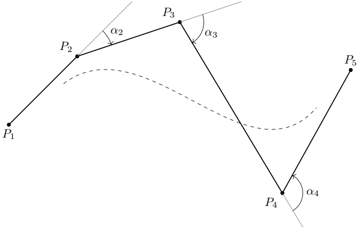
General observations

- Simplicity trumps accuracy
- It is generally possible to find clever simplifications

II: Angle methods

Recall

$$\begin{aligned}\Delta \mathbf{P}_{i-1} \cdot \Delta \mathbf{P}_i =& \Delta \mathbf{P}^1_{i-1} \Delta \mathbf{P}^1_i + \Delta \mathbf{P}^2_{i-1} \Delta \mathbf{P}^2_i \\ =& \|\Delta \mathbf{P}_{i-1}\| \|\Delta \mathbf{P}_i\| \cos \alpha_i.\end{aligned}$$



Definition

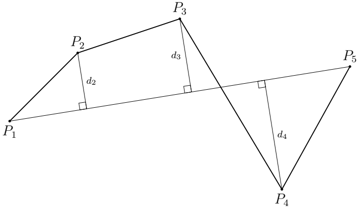
$$x_i = \frac{\Delta \mathbf{P}^1_{i-1} \Delta \mathbf{P}^1_i + \Delta \mathbf{P}^2_{i-1} \Delta \mathbf{P}^2_i}{\sqrt{(\Delta \mathbf{P}^1_{i-1})^2 + (\Delta \mathbf{P}^2_{i-1})^2}\sqrt{(\Delta \mathbf{P}^1_i)^2 + (\Delta \mathbf{P}^2_i)^2}}$$

$$\alpha_i = \arccos(x_i), \qquad \tilde{\alpha}_i = -\frac{\pi}{2} x_i |x_i| + \frac{\pi}{2}.$$

III: Base line methods

Definition (Base line)

Given control points $\mathbf{P}_\mu$ and $\mathbf{P}_\nu$ $(\mu < \nu)$, the *base line* is the line segment $\mathbf{P}_\mu\mathbf{P}_\nu$.



Definition

$$d_i = \begin{cases} d(\mathbf{P}_i, \mathbf{P}_\mu\mathbf{P}_\nu) & \text{if the projection of } \mathbf{P}_i \text{ lies on } \mathbf{P}_\mu\mathbf{P}_\nu \\ \min\{d(\mathbf{P}_i, \mathbf{P}_\mu), d(\mathbf{P}_i, \mathbf{P}_\nu)\} & \text{if not.} \end{cases}$$

$$r_i = \frac{L_{\sigma(i),\sigma(i+1)}}{d(\mathbf{P}_{\sigma(i)}, \mathbf{P}_{\sigma(i+1)})} \qquad \text{for } i = 0, \ldots, S-1.$$

Placement of knots

Recall: from Böhms equation

$$\bar{t}_j^* = \omega_j t_j^* + (1 - \omega_j) t_{j-1}^* \quad \text{with} \quad \omega_j = \frac{z - t_j}{t_{j+p} - t_j}.$$

Assume we want a ratio $\omega_\mu/(1 - \omega_\mu)$ between c_μ and $c_{\mu-1}$. Then

$$z = \omega_\mu(t_{\mu+p} - t_\mu) + t_\mu.$$

Observation

- The new coefficients lie on each of the p line segments of the relevant old local control polygon.
- More precisely, $p - 1$ old vertices vanish, and these are replaced by p new vertices.

Stopping criteria

1. Limiting the maximum number of knots to insert.
2. Control smoothness using the geometric criteria directly.
3. Introduce threshold for difference of some measure of old and new control polygon.

Distance-to-pixel computation

- Length of diagonal of rectangle containing the spline: $\Delta\mathbf{c}$
- Chosen limit for pixel distance: δ
- The diagonal length of the screen measured in number of pixels: ρ
- Control point movement threshold distance ϵ given by:

$$\epsilon = \frac{\delta}{\rho}\Delta\mathbf{c}. \tag{2}$$

The adaptive algorithm

1. Preprocessing: reduce spline to having (p+1)-regular knot vector and no discontinuities.
2. Initialize threshold values and allocate arrays
3. While: (#inserted knots < max-to-insert) AND (time < max-time):
    - Apply geometric criterion $\mathcal{C}$ to find candidate control points
    - Compute exact value of new knot using rule $\mathcal{R}$
    - Insert knot using knot insertion algorithm $\mathcal{I}$
    - Check the threshold stopping criterion $\mathcal{S}$ in this area of the control polygon and flag accordingly
4. Return refined spline control polygon: array of points

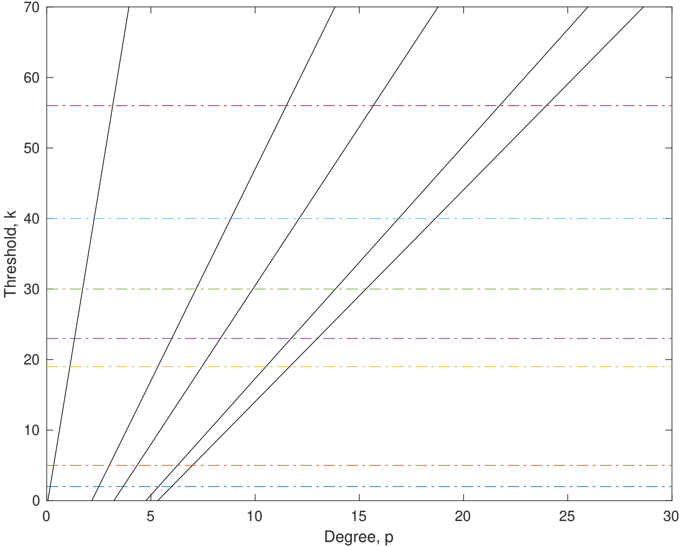
Computational cost

Criterion $\mathcal{C}$	Operation count k
Angle method	40
Base line method	30
Simplified angle method	23
Length-ratio method	19
Length method	5
Simplified length method	2

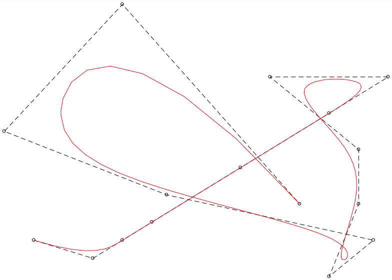
- In total, $Q = kp + o + lp + 4p$ ops. per iteration of main loop
- Assume we started with N_0 points and inserted $K = yN_0$ knots
- Average number of operations per final point: $R = QK/(N_0 + K)$
- Evaluation cost: $R_{eval} = \frac{3}{2}p(p+1)s$, should have $R/R_{eval} < 1$
- The *operation threshold value* in the criterion $\mathcal{C}$ is

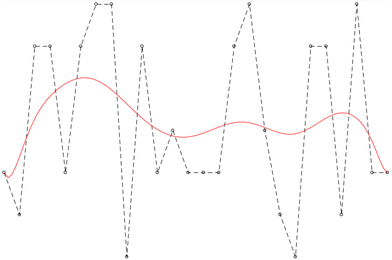
$$k_l = \frac{3}{2}(p+1)(1+\frac{1}{y})s - 3s - 13.$$

Comparing theoretical performance



Test curves

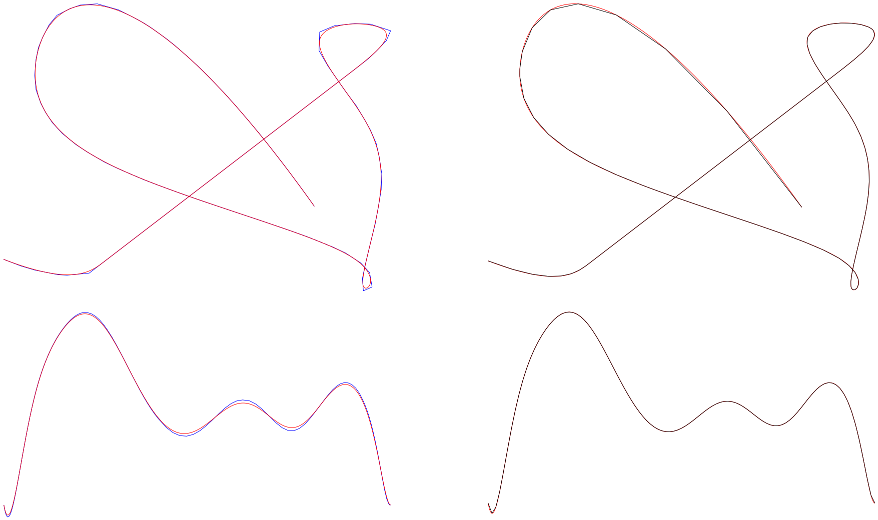




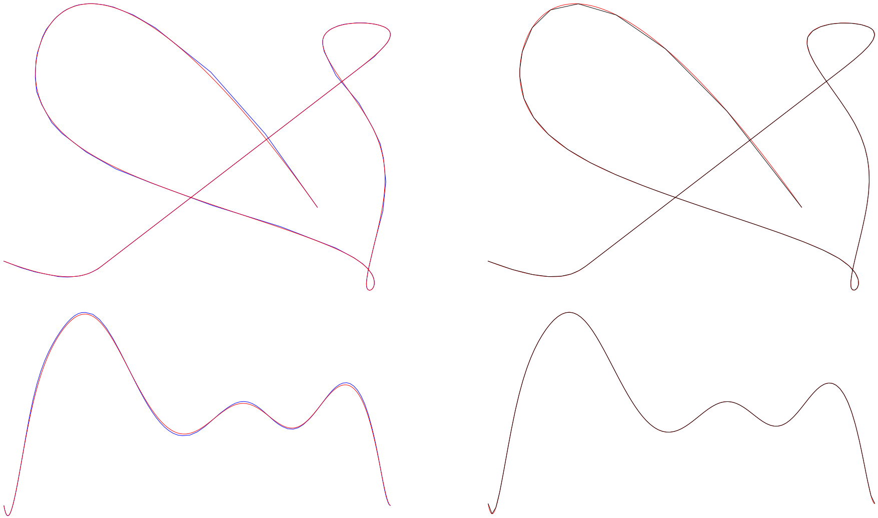
Generation of random curves

1. $n \in [3, 500)$
2. $p \in [2, 30)$
3. $\mathbf{t} \in [0, 10)^{n+p+1}$ (sorted, regular)
4. $\mathbf{c} \in [0, 10)^{s \times n}$

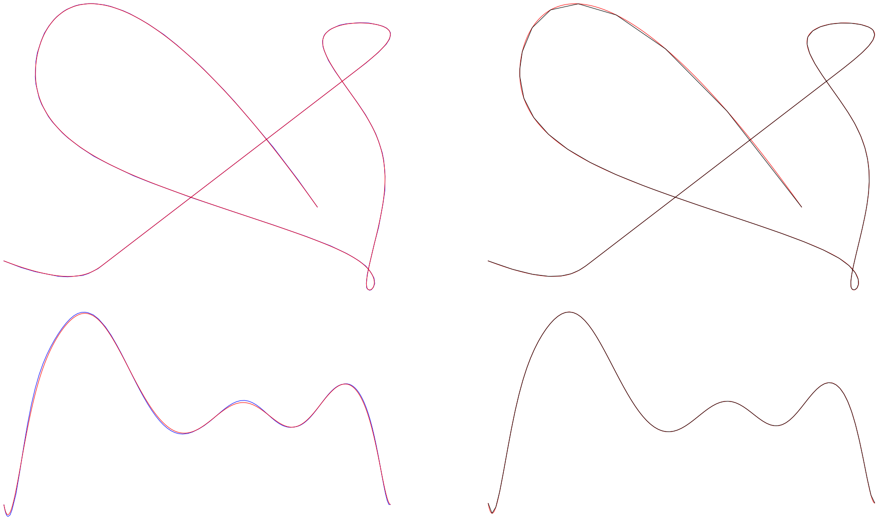
Results: length methods



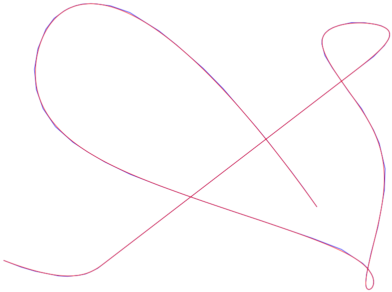
Results: angle methods

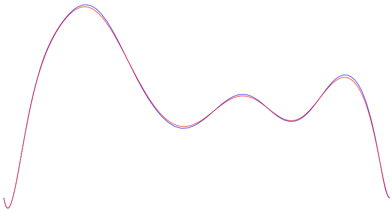


Results: base line methods



Combining methods: 2-stage process





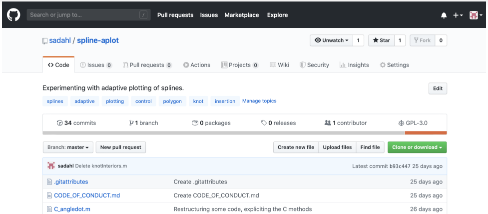
1. Reduce segment lengths
2. Proceed with angle/ratio criterion

Theorem (Quad. conv. of CP)

$$||\Gamma(f) - f||_{[t_1^*, t_n^*]} \leq K_p h^2 ||D^2 f||_{[t_1, t_{n+p+1}]},$$

$$K_p = \frac{2^{p-1}}{(p-2)!} p^3 (p-1) + \frac{1}{8}.$$

Project status - alive & online



Next steps

1. Polish 2-stage algorithm and implement in C
2. Complete implementation for surfaces
3. Implement parallelized version of the algorithm
4. Measure time consumption on different platforms
5. Create a package and advocate for splines to become default geometric primitives

de Boor. C
A practical guide to splines.
New York, USA: Springer, 2001.

Foley, J. D. et al.
Computer Graphics: Principles and Practice (2nd ed.)
Boston, USA: Addison-Wesley Longman Publishing Co., Inc., 1990

Dahl. S
Adaptive plotting of splines.
Oslo, Norway: Department of Mathematics, University of Oslo, 2019.

Dahl. S
Github public repository.
https://github.com/sadahl/spline-aplot.git