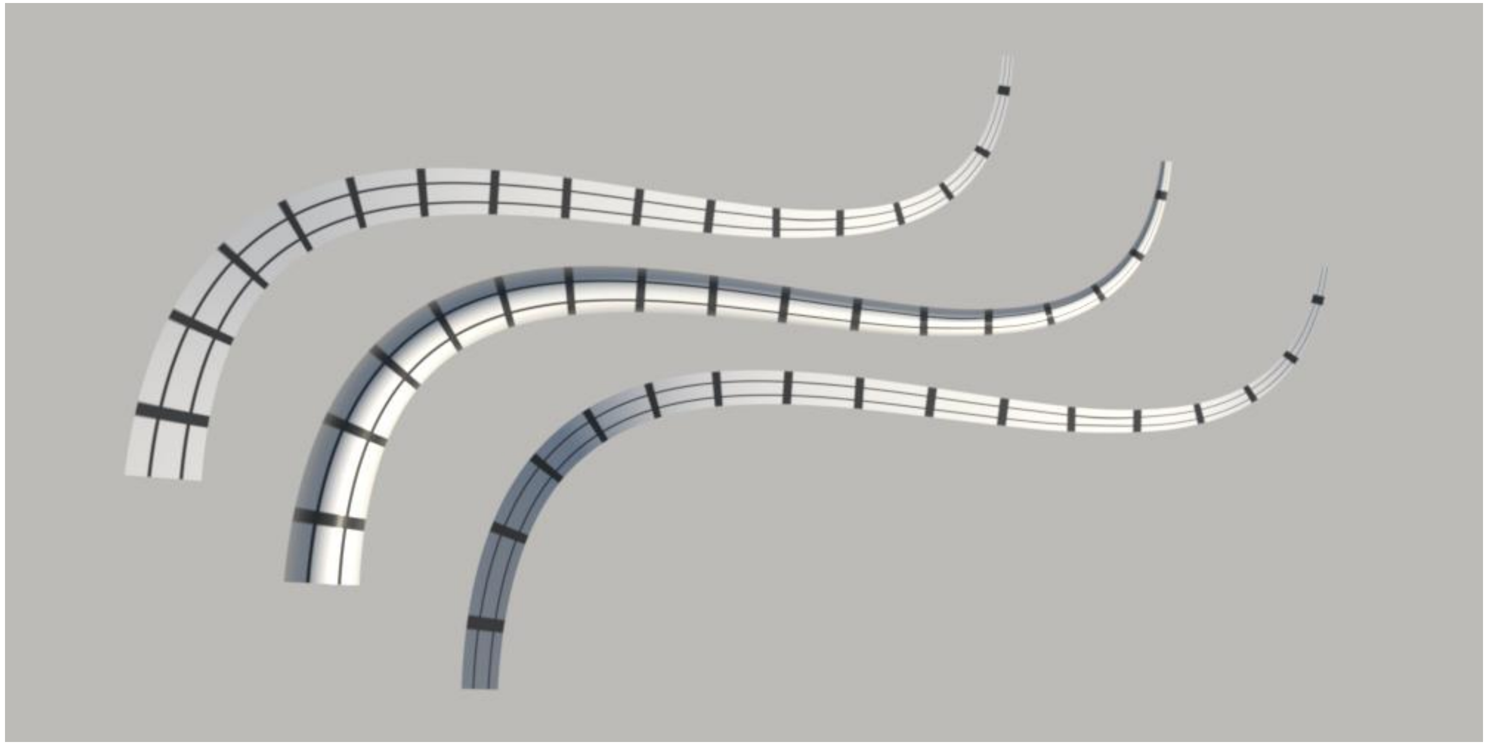


Curves and splines

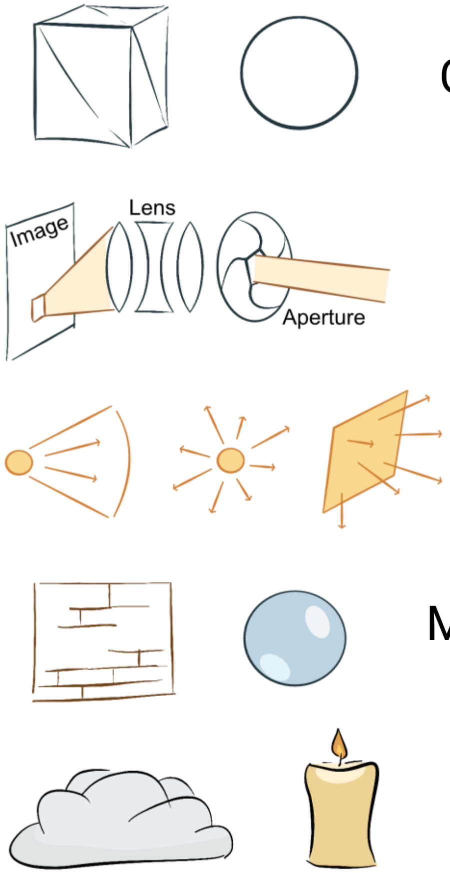
Saarland Informatics Campus

UNIVERSITÄT DES SAARLANDES

Upcoming lectures

- Some more advanced geometry (today & Thursday)
- GPU / Real-time rendering
  - Camera models and clipping
    - How to encode perspective cameras as a matrix and project your geometry on the screen
  - Rasterization
    - How to compute pixel occupancy from projected triangles
  - Graphics APIs
    - How to interface with your GPU and get it to do stuff
  - Shader programming
    - How to program the GPU
  - Shadow algorithms
    - How to get shadows if you cannot afford ray tracing

Today and Thursday: some more advanced geometry



Geometries

Camera model

Light sources

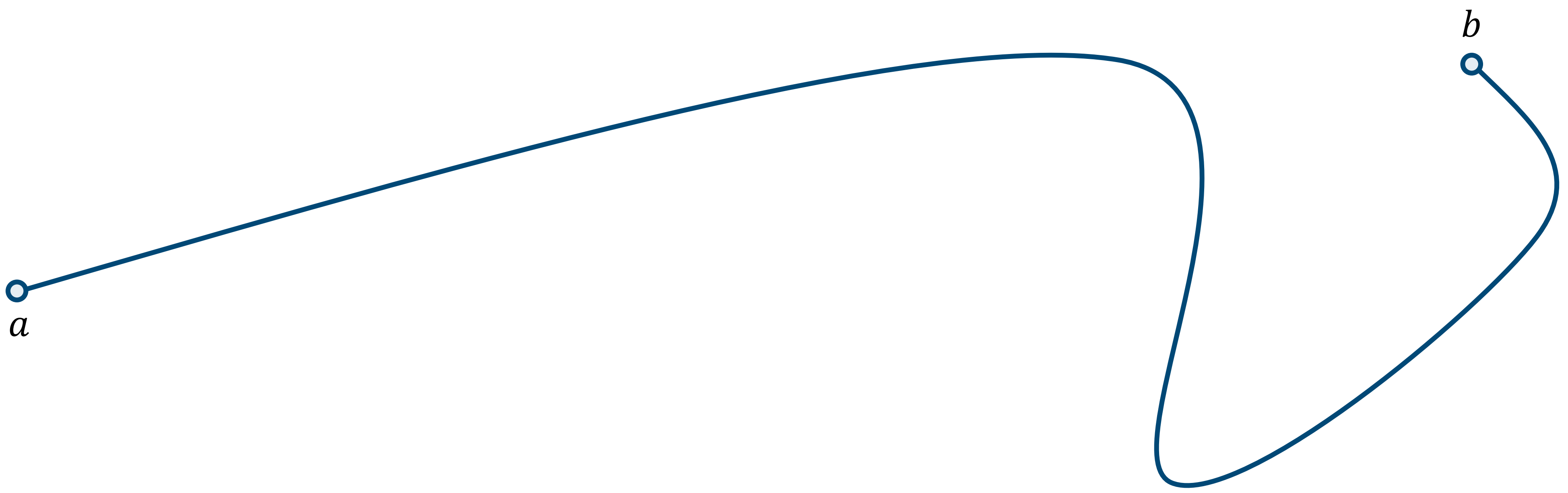
Materials & Textures

Volumes

Curves

What are curves?

- A line that is not straight
- Sequence of points that takes you from a to b



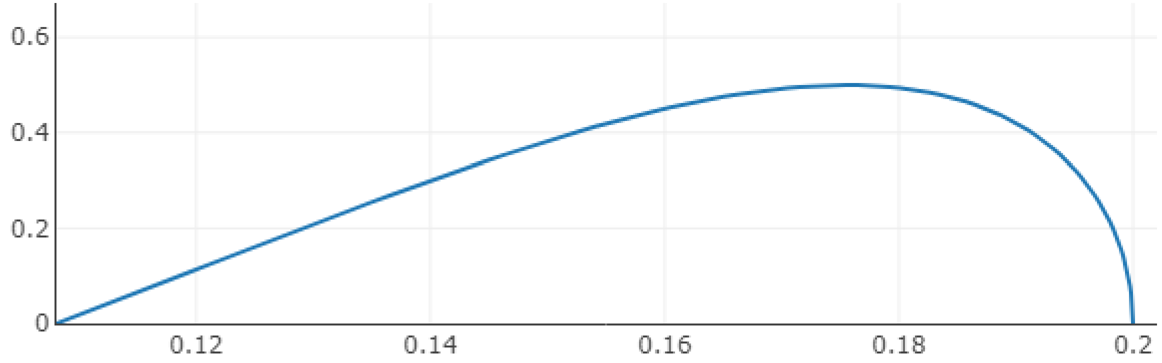
Fun fact

- If you look for info online, search engines are slightly more helpful if you search for the singular “curve” rather than its plural form

Parametric curves

- A useful way to describe a curve

$$c(t) = \begin{pmatrix} 0.2\cos t \\ 0.5\sin \pi t \end{pmatrix}$$
$$t \in [0,1]$$



- t is the “travel distance” or “time” along the curve

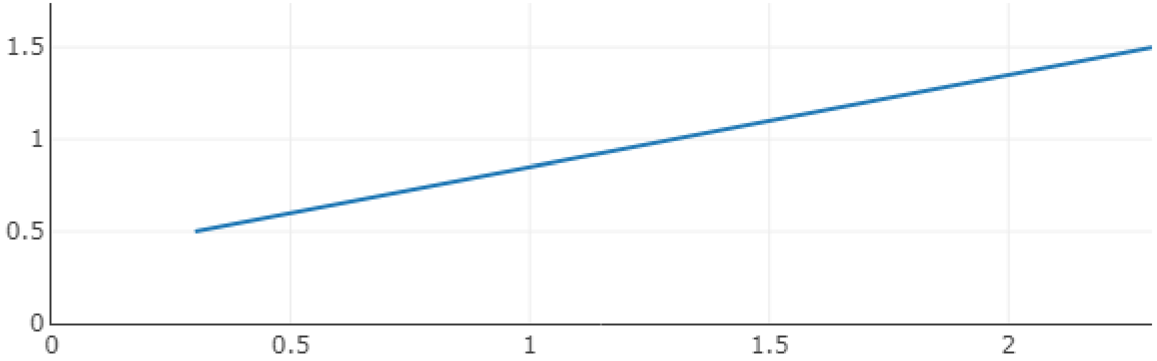
A familiar example

- The ray equation you have used this whole time is a parametric curve

$$r(t) = o + td$$

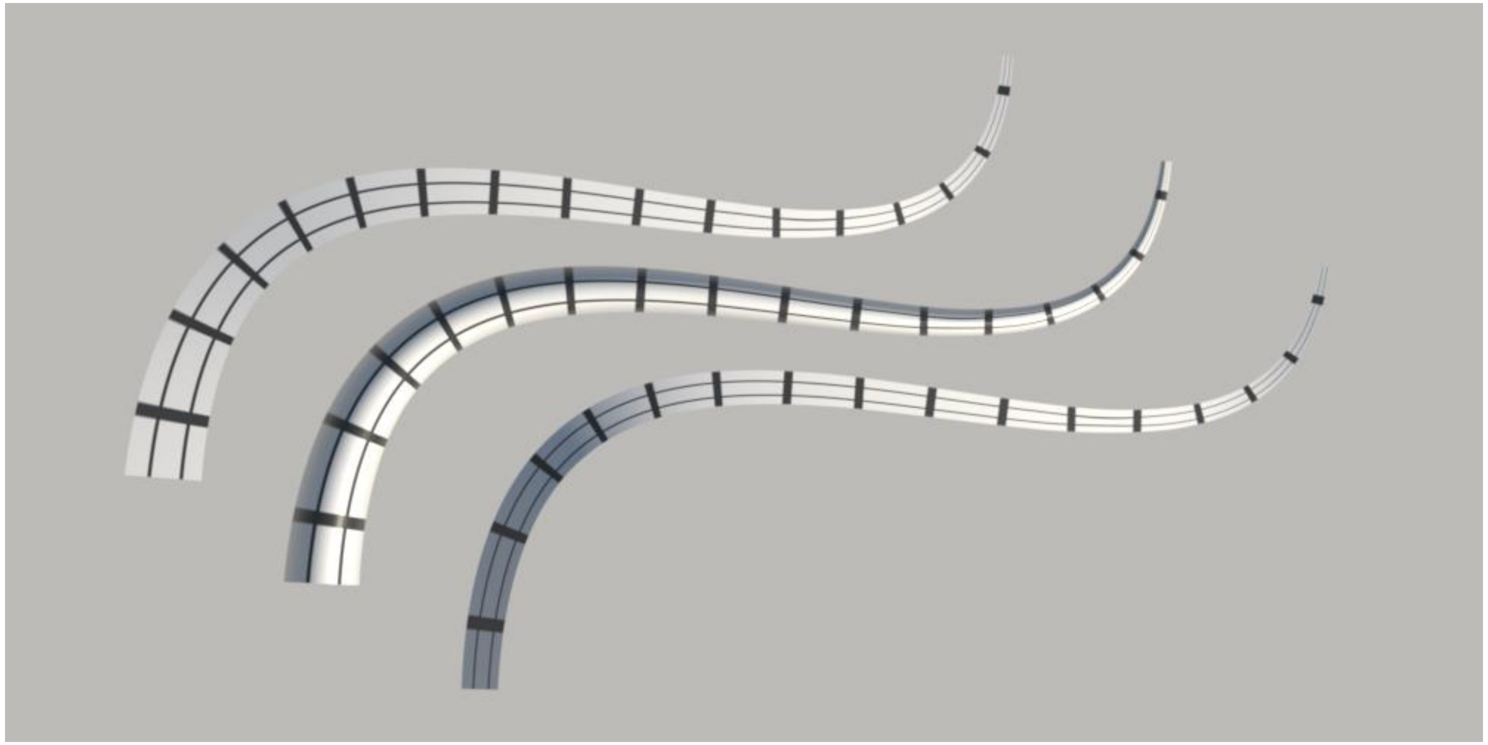
- e.g.,

$$r(t) = \begin{pmatrix} 0.3 \\ 0.5 \end{pmatrix} + t \begin{pmatrix} 2 \\ 1 \end{pmatrix}$$



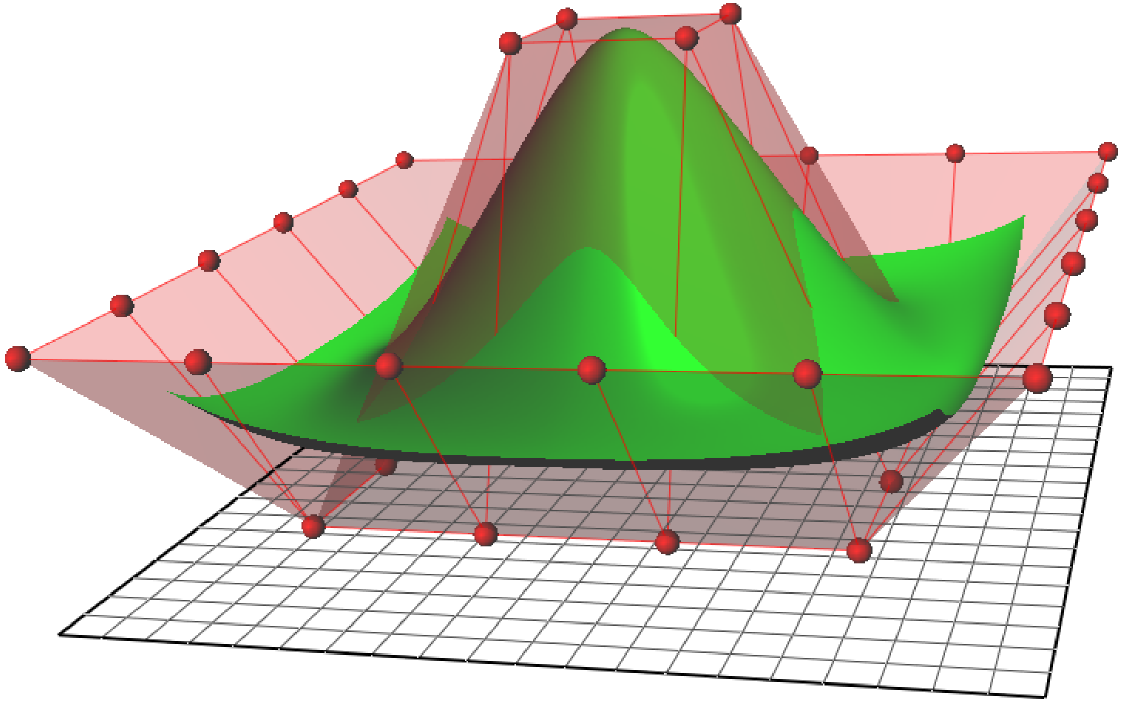
Wait, what do we even need those curves for?

Example: Modelling hair and fibers



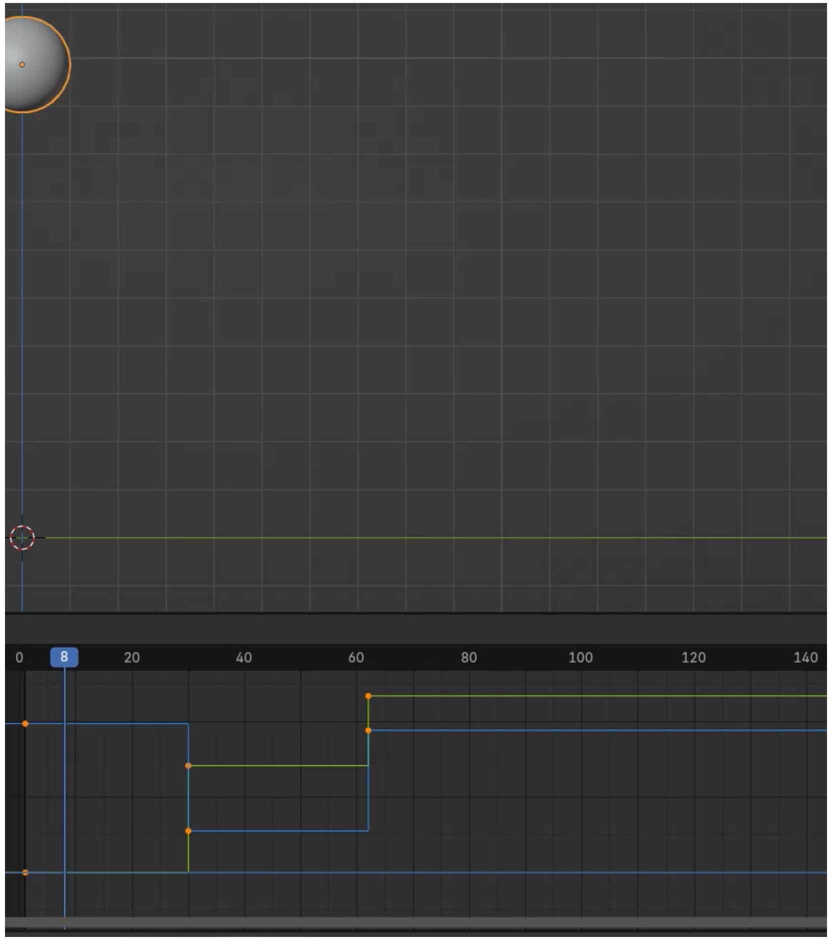
- With curves, we can describe intricate geometries like these with a few control points
- With triangle meshes, this would be millions of vertices

Example: Modelling smooth surfaces

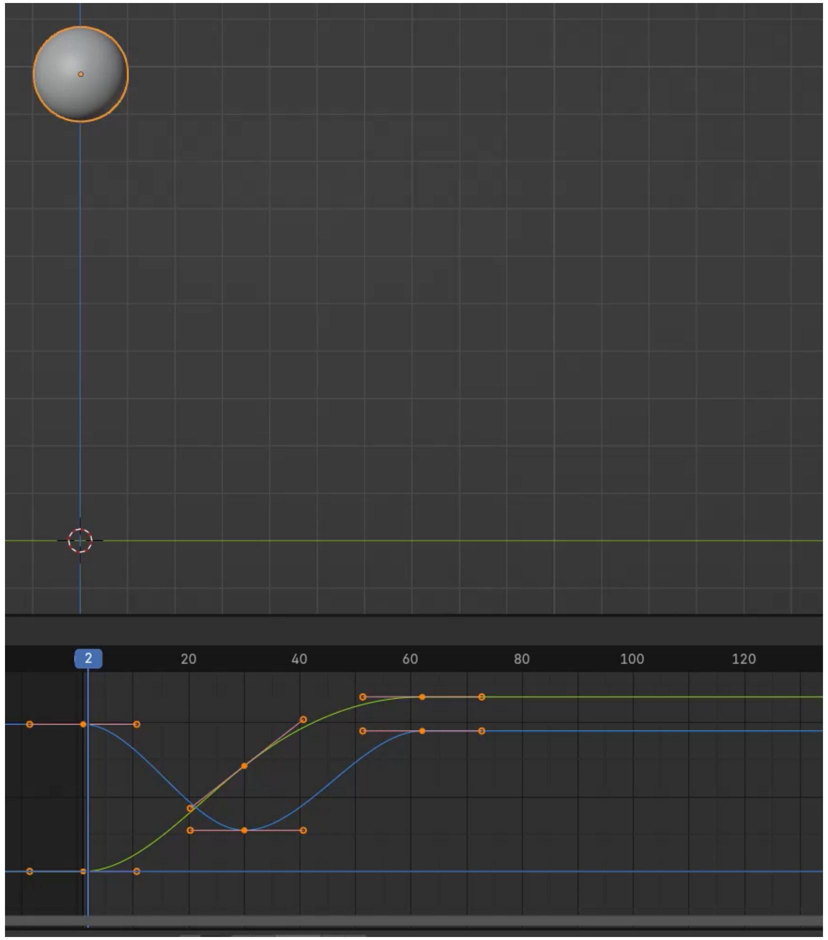


- Curve / spline: A few control points
- Mesh: requires millions of triangles for similar result

Example: Interpolating keyframes in animations

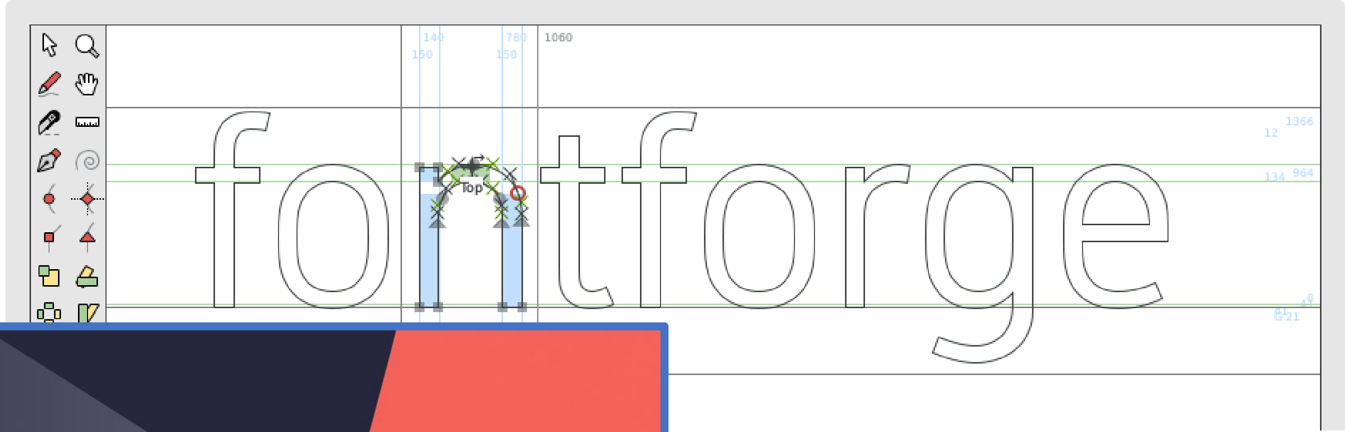


Keyframes only



Bézier interpolation

Example: 2D vector graphics and fonts

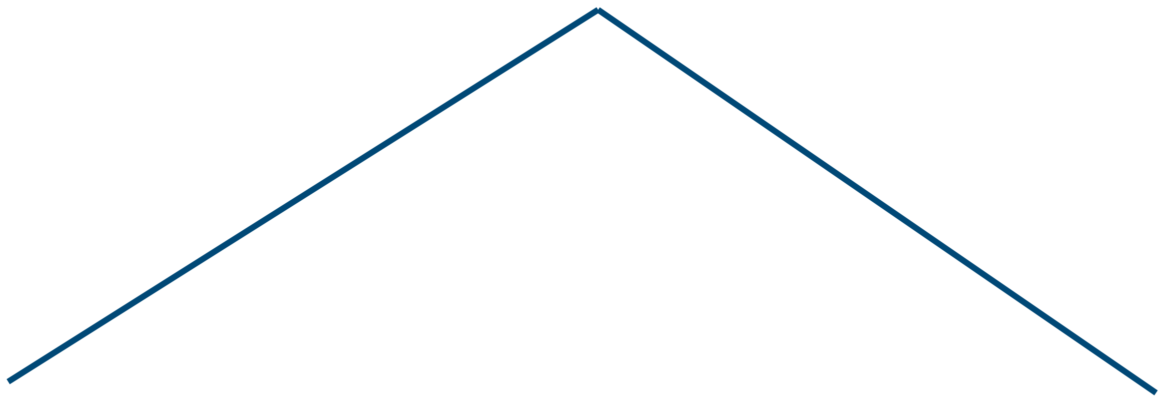


Images that work at any resolution

Smoothness

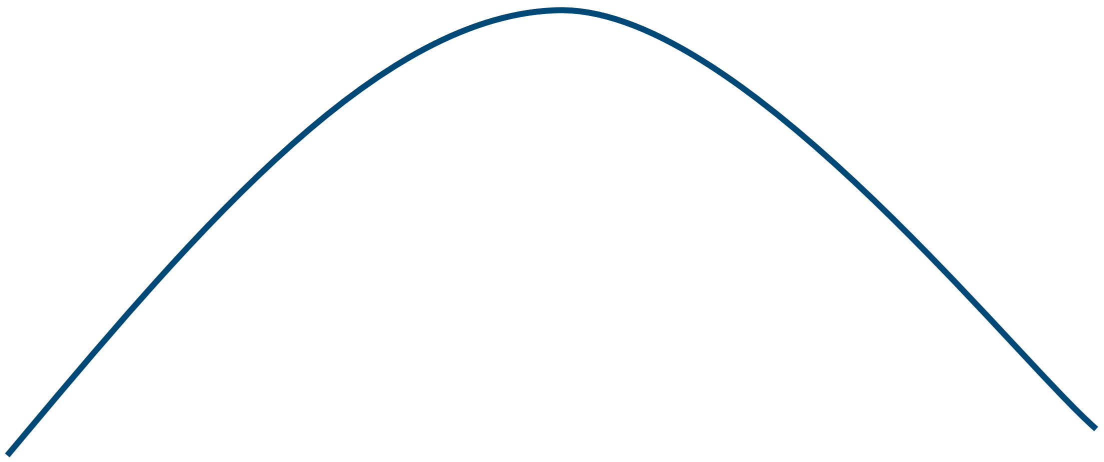
C^0 continuity

- $c(t)$ is continuous, but not differentiable



C^1 continuity

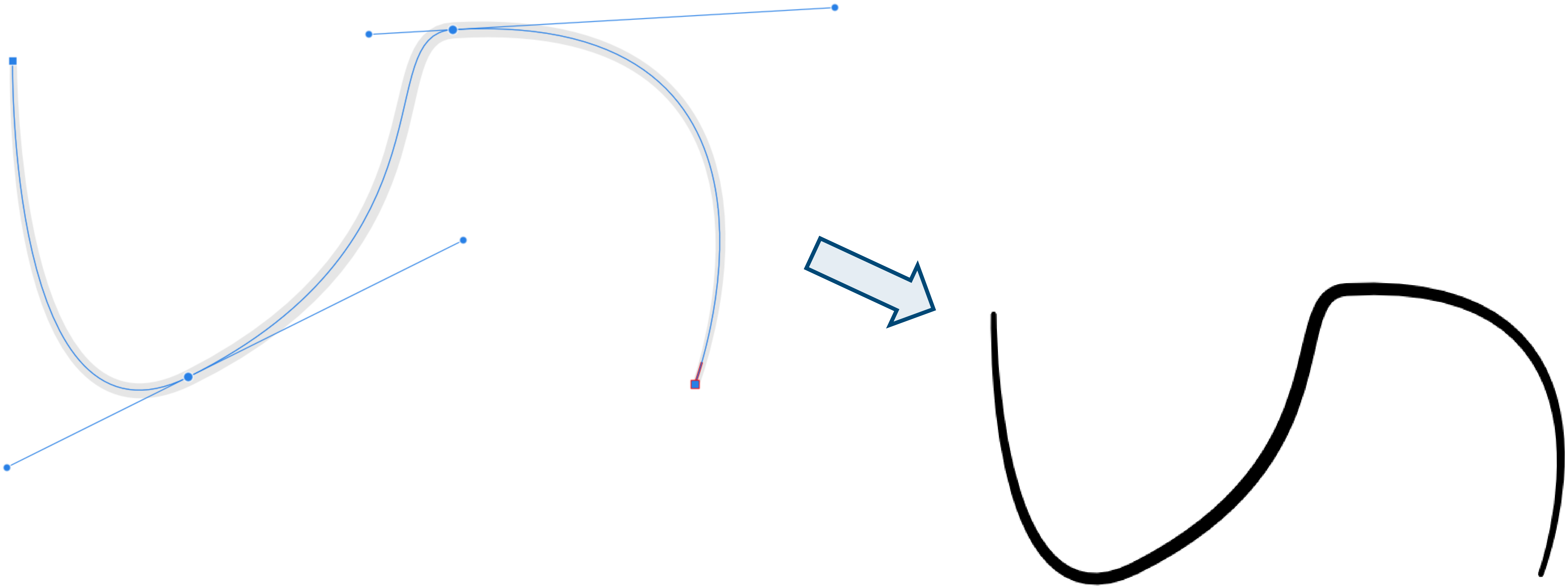
- first derivative $\frac{d}{dt}c(t)$ is C^0 continuous



- And so on: For C^n, the nth derivative has to be C^{n-1} continuous
- Typically, C^2 is desired for a perceptually smooth result

Defining and modelling curves

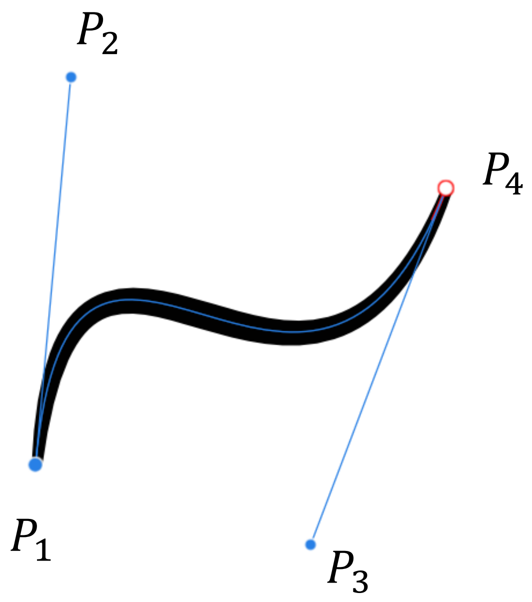
Control point interpolation



Cubic Bézier curves

- Polynomial interpolation
- 4 control points $P_1 \dots P_4$, $t \in [0,1]$

$$c(t) = (1-t)^3 P_1 + 3(1-t)^2 t P_2 + 3(1-t) t^2 P_3 + t^3 P_4$$

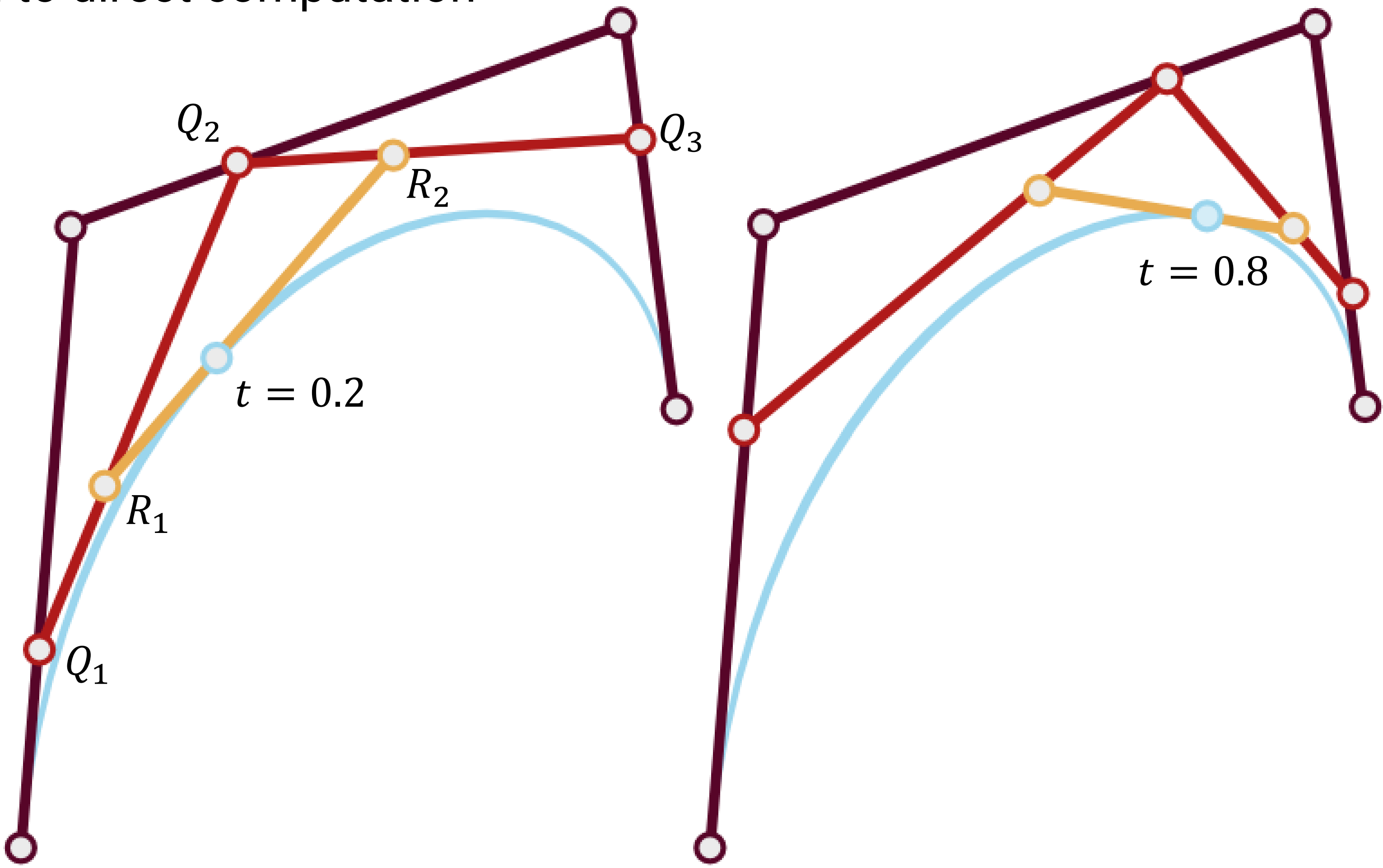


- Interpolate smoothly from P_1 to P_4
- P_2 and P_3 "act like magnets" to steer the curve in-between
- Explicit tangents:
  - $P_2 - P_1$ is the tangent at P_1
  - Allows smoothly connecting curves
- Convex hull property:
  - All points $c(t)$ inside convex hull of $\{P_1, P_2, P_3, P_4\}$

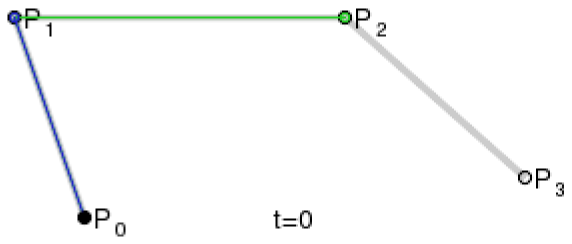
de Casteljau‘s algorithm (here for cubic Bézier curves)

- Faster & numerically stable compared to direct computation
- Recursive linear interpolation

1. Compute $Q_1 = tP_1 + (1-t)P_2$
   - Same for Q_2 and Q_3
2. Compute $R_1 = tQ_1 + (1-t)Q_2$
   - Same for R_2
3. Compute result as $tR_1 + (1-t)R_2$



de Casteljau's algorithm



Splines

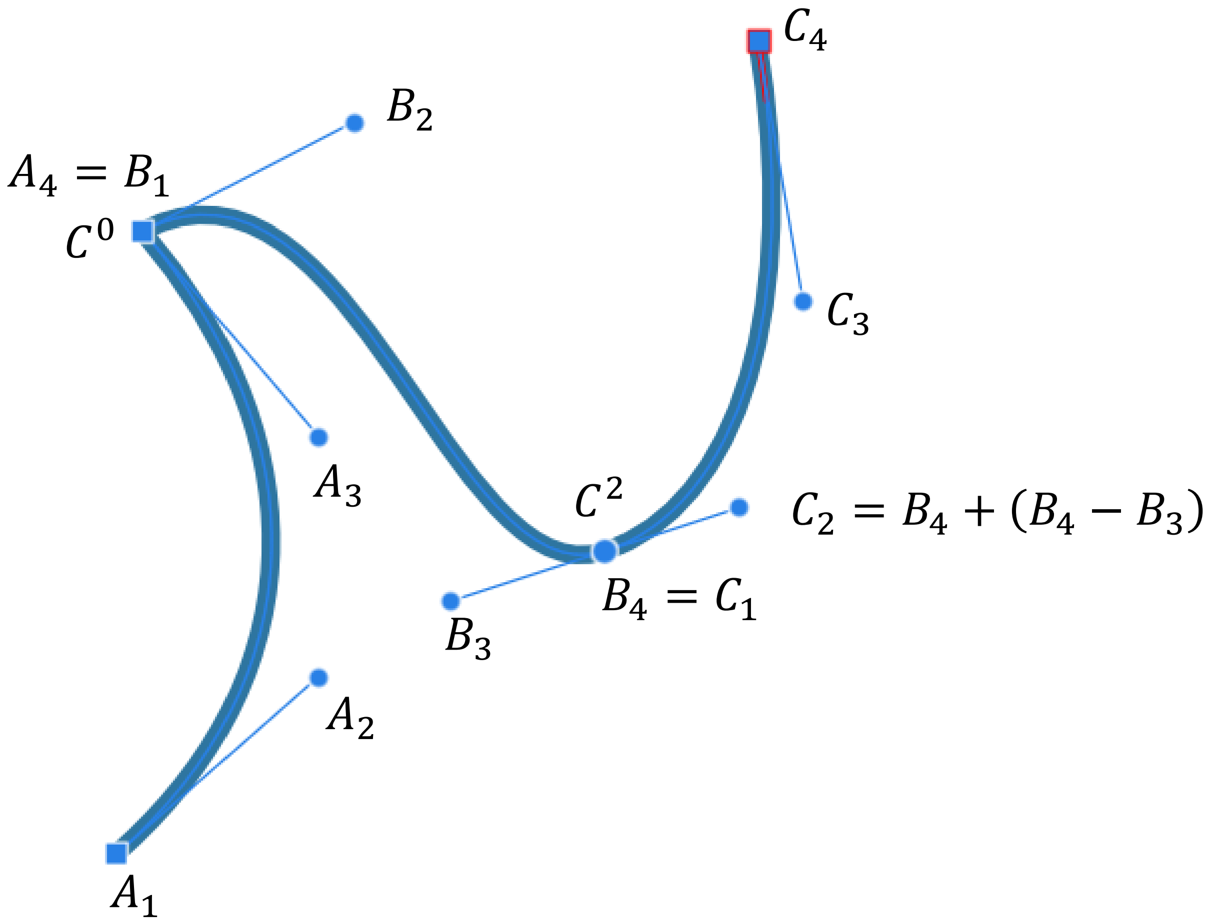
Stitching curves together

A “spline” is a piecewise polynomial function

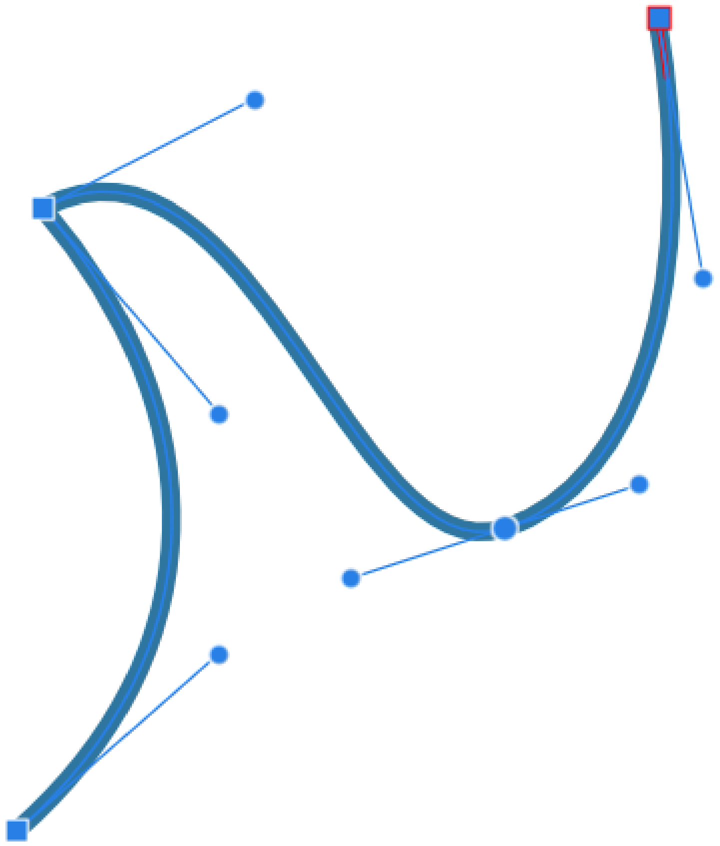
- We *could* use a single high-degree polynomial curve
- But:
  - Difficult to fit / model
  - Numerical issues

- Splines stitch low-order polynomial curves instead
  - Simple
  - Numerically stable (if done well)

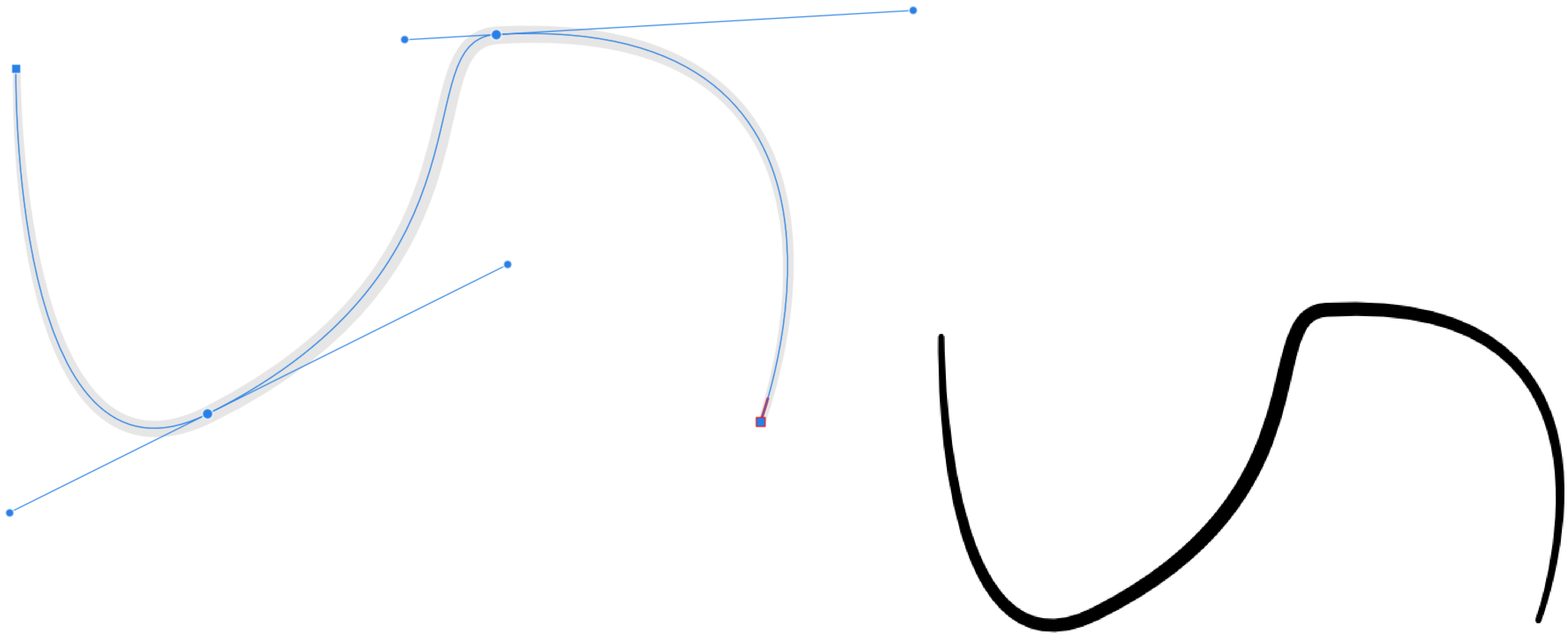
Piecewise Bézier spline



Quiz time! How smooth is this Bézier spline?



And what about this one?



Smoothness of piecewise cubic Bézier splines

- The “handles” (control polygon edges) have to
  - Align (C^1)
  - Be of same length (C^2)

Bézier curves and splines are popular, but not the only option

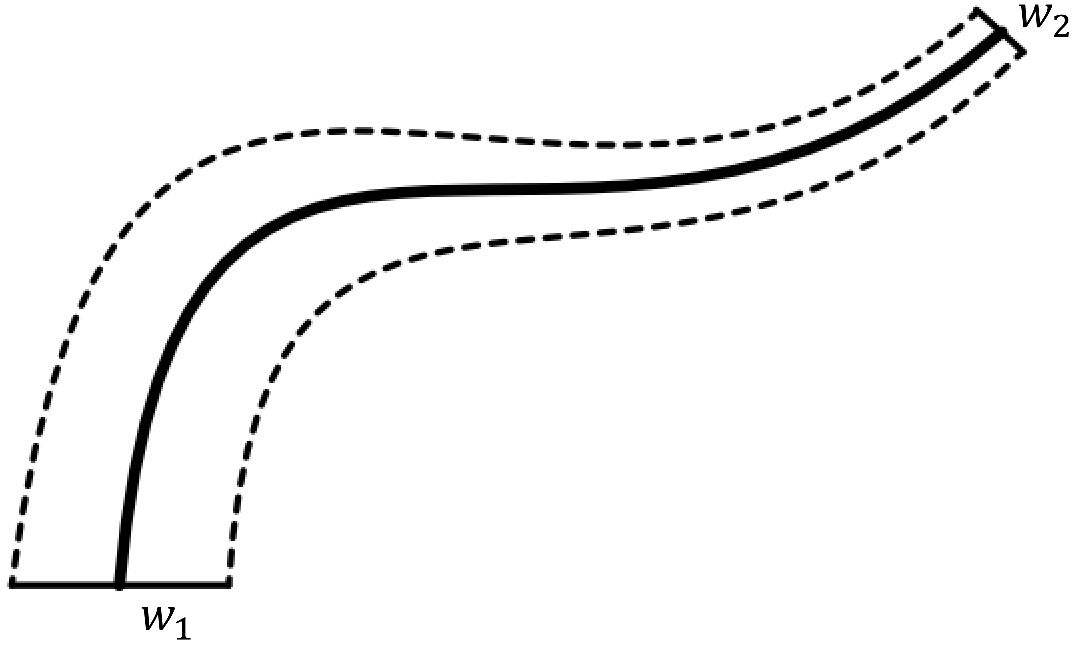
- B-Splines
- Catmull-Rom splines
- Hermite splines

Rendering splines: Fibers

Hair, fur, grass, ...

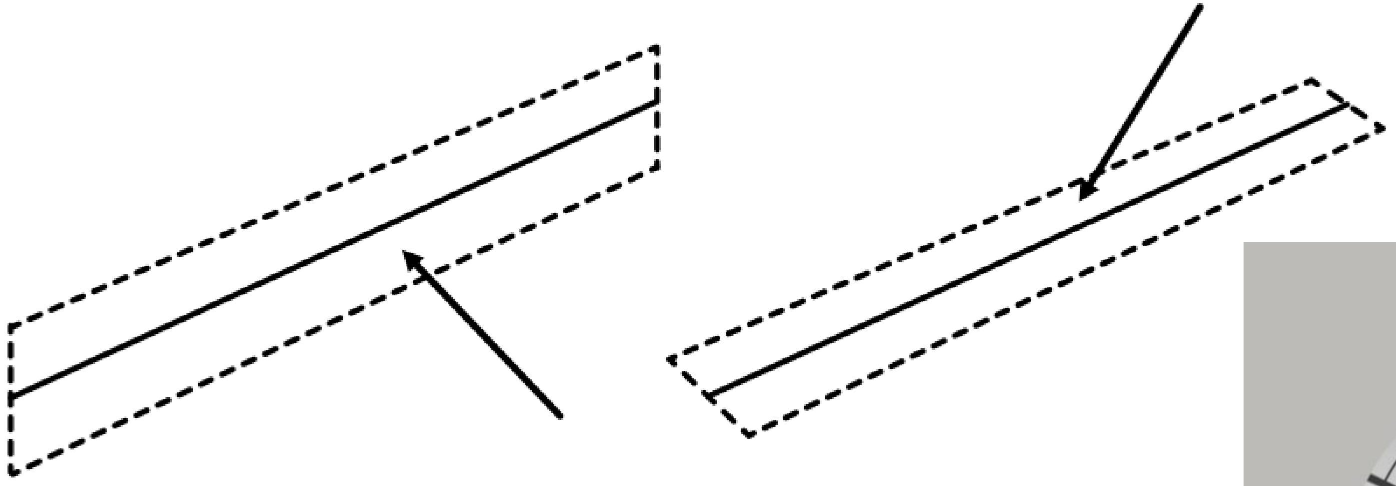
Defining a fiber with a curve and a width

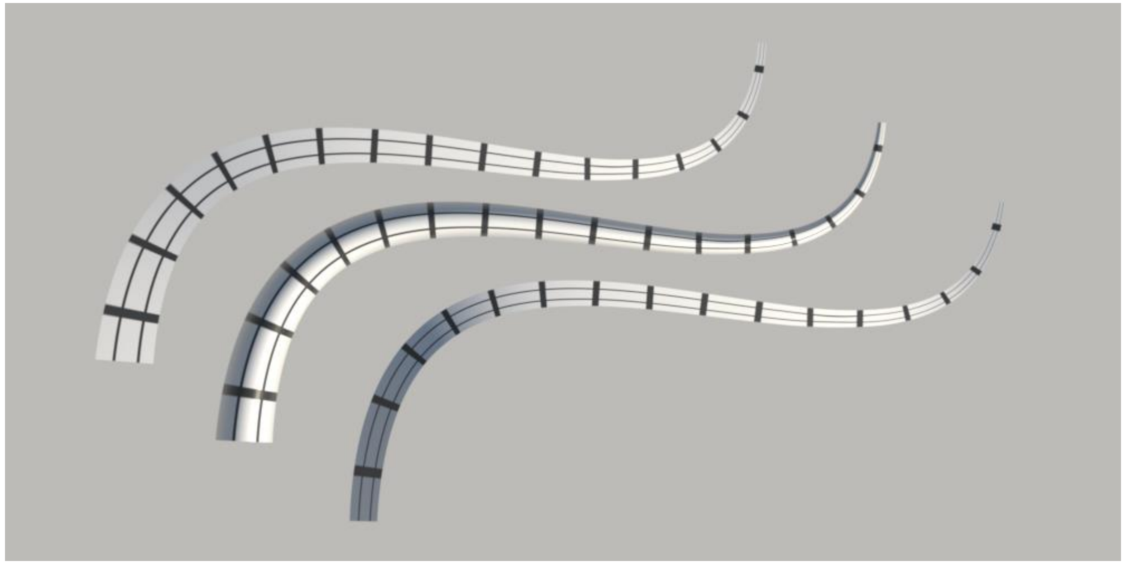
- e.g., “thicken” a Bézier curve



Flat fibers

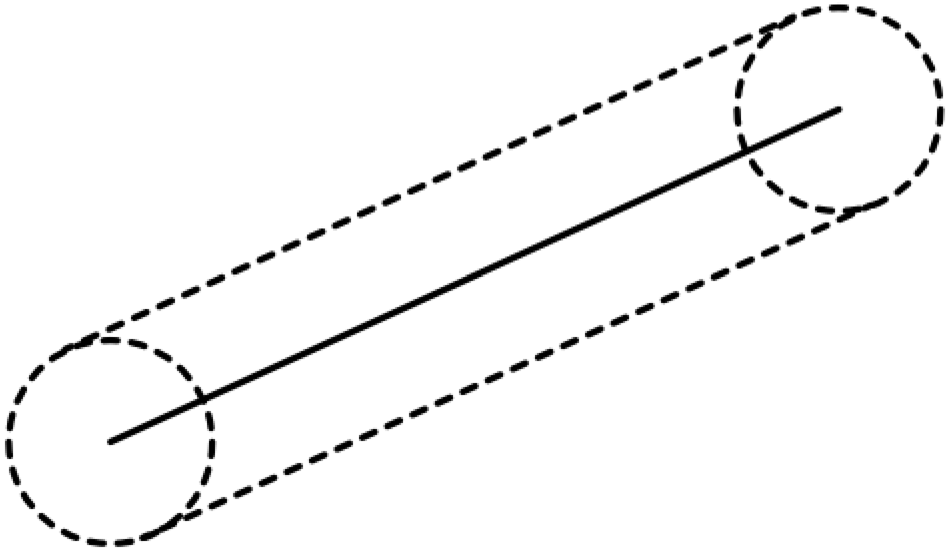
- Always face the ray

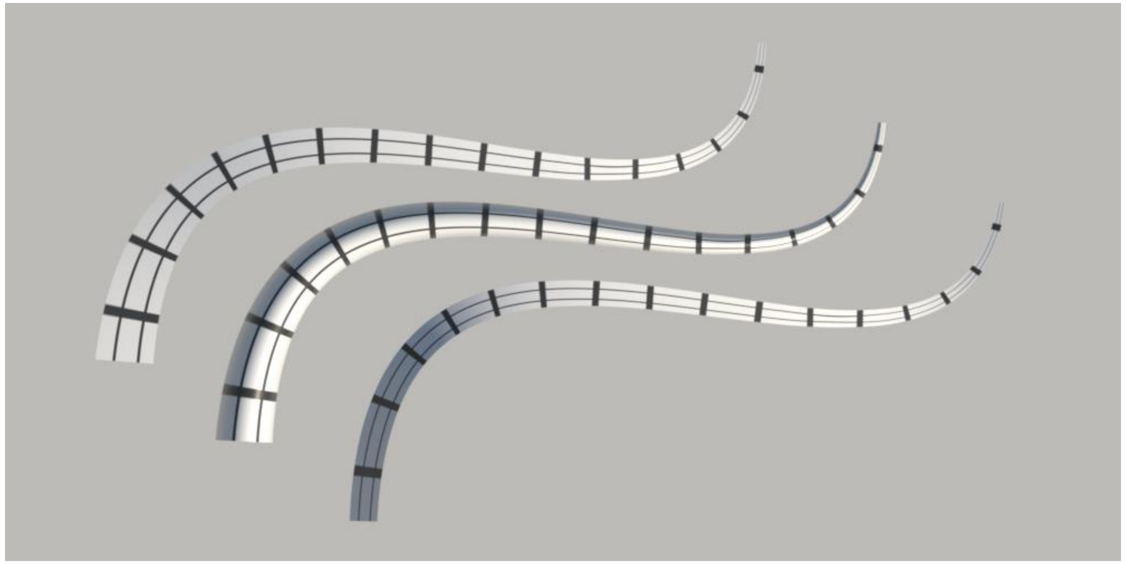




Cylinder fibers

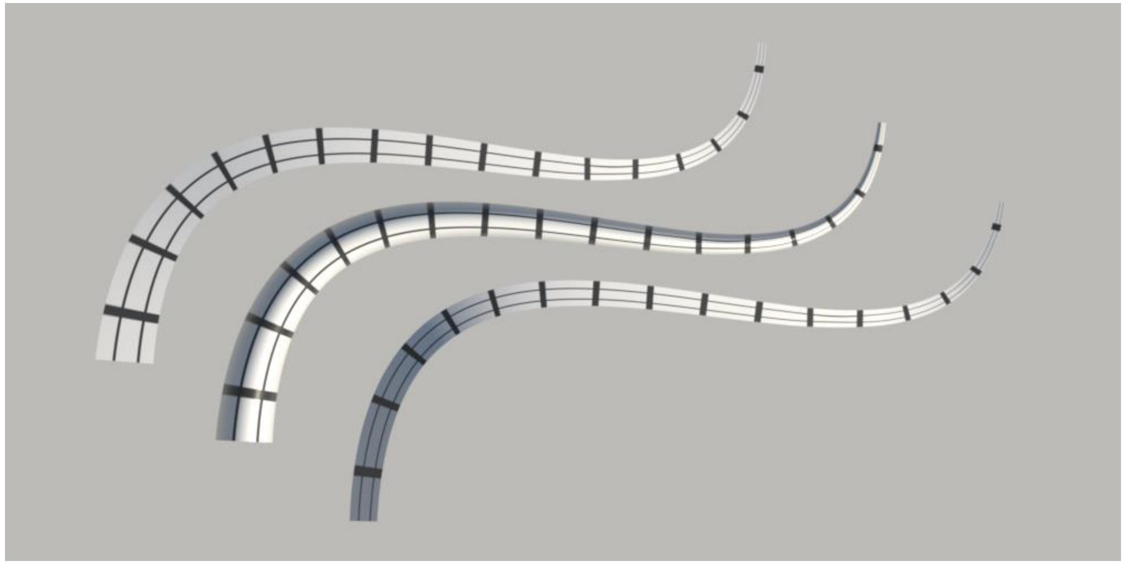
- Sweep a circle along the curve





Ribbon fibers

- Flat, orientation is fixed and interpolated between control points

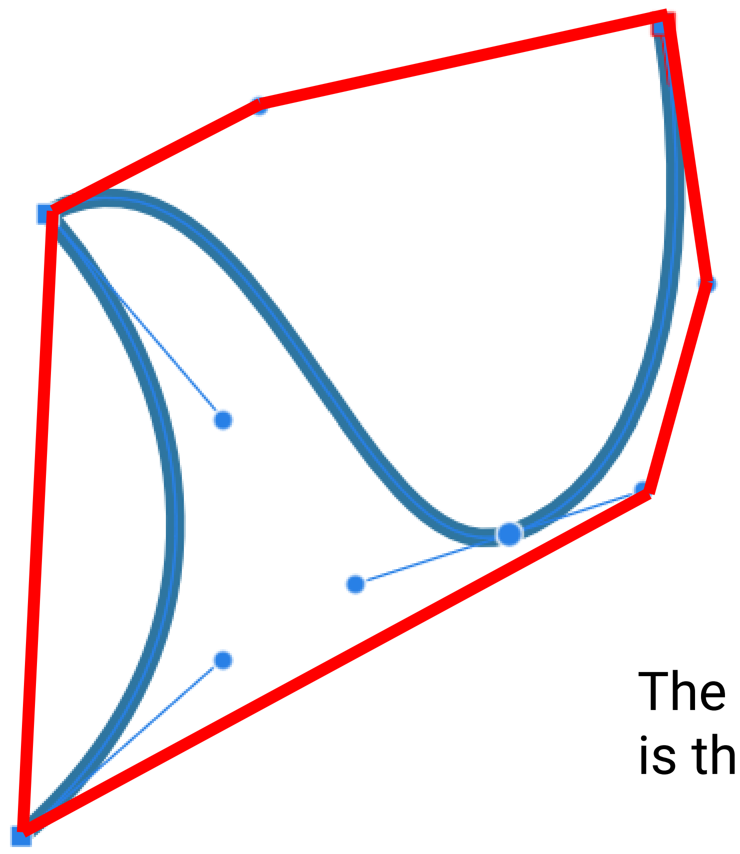


Ray-tracing a fiber

- Same root-finding problem as always:
  - take the ray equation
  - substitute into the spline equation
  - solve
- Expensive, so some clever culling (using the convex hull) is beneficial

- https://pbr-book.org/4ed/Shapes/Curves

The curve lies inside the convex hull of the control points



The convex hull of a point-set / mesh / polygon
is the smallest polygon / polyhedron that contains all points

Rendering splines: Smooth surfaces

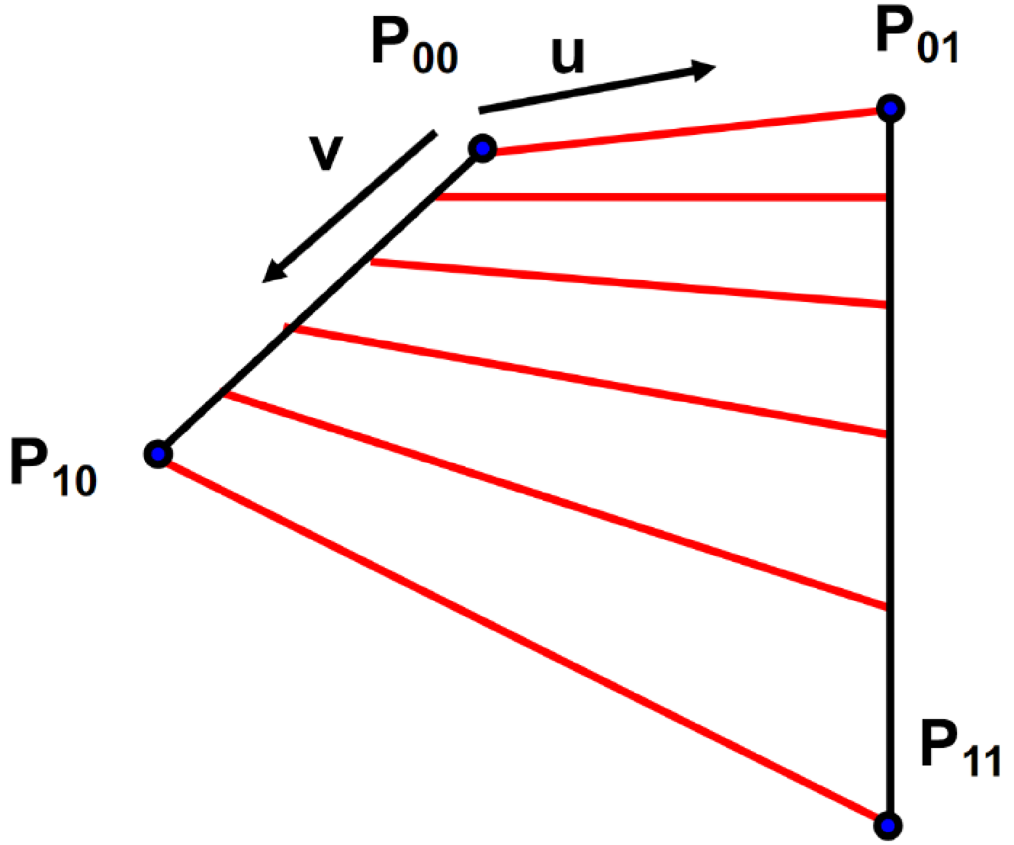
Tensor product surface

- "Extrudes" a curve into a surface, by following another curve

- The hair curves we discussed just now are special cases of this idea

Example: bilinear patch

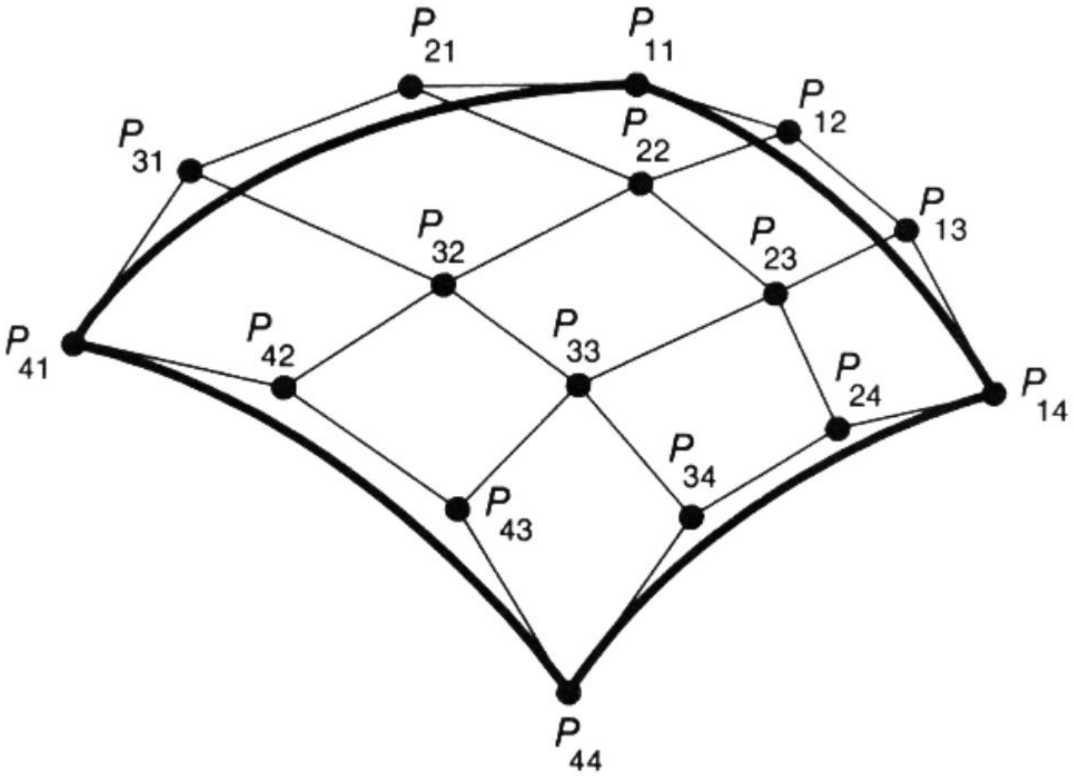
- Tensor product of two linear curves
- Useful shape for modelling:
  - Connects 4 vertices
  - But does not force them to be coplanar



Example: Bézier surface

- Same idea as a bilinear patch
- But with Bézier curves

- Aligning Bézier patches nicely is harder than curves
  - ➔ NURBS (non-uniform rational B-spline surface)
  - ➔ Dedicated modelling tools (used by CAD software)



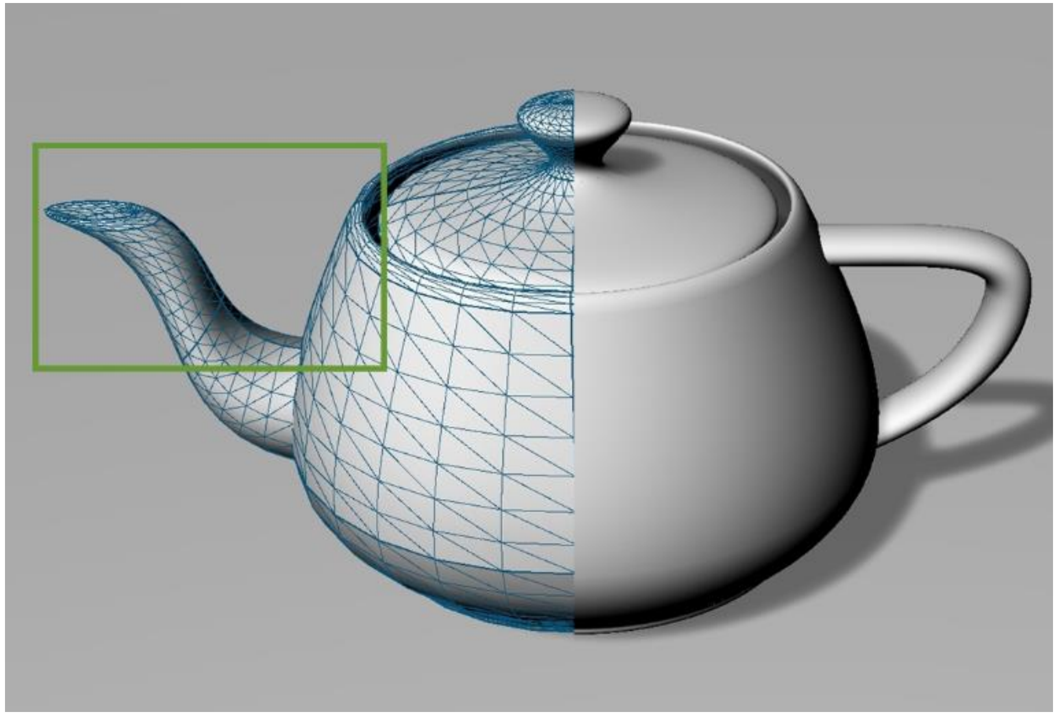
Ray-tracing a spline surface

- We *can* directly intersect a ray and a spline patch
- This is a root-finding problem
  - Substitute ray equation into spline equation and solve
  - Same as any other ray-object intersection test
- But: For higher-degree polynomials, this is *expensive*

Tessellation

- Approximate the spline surface with a triangle mesh
- For a 2D tensor product surface:
  - Subdivide the $u \times v$ domain with a regular grid
  - Evaluate spline position for each grid corner
  - Create triangles (or bilinear patches) with those vertices

Xiong et al. 2023 “ETER”



Summary

Today: Curves and splines

- Smooth interpolation of control points
- Many uses (2D vector graphics, fonts, key-frame interpolation in animation, ...)
- In our focus area (static-scene rendering):
  - Modelling hair, grass, and similar fine structures
  - Modelling smooth surfaces

Next up: Subdivision surfaces

- Uses ideas from curves and splines
- Subdivide coarse geometry so it becomes more smooth