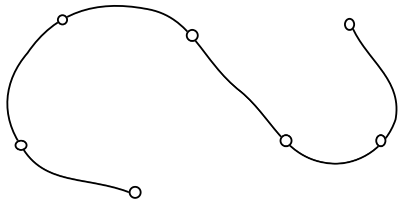
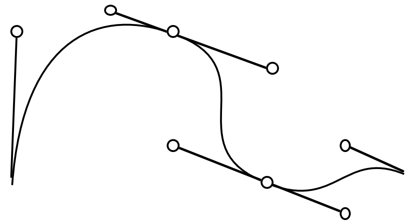


Splines



Originally a drafting tool to create a smooth curve

In compute graphics: a curve built from sections, each described by 3rd degree polynomial.

Very common in non-real-time graphics, both 2D and 3D!

Useful also for real-time.

Applications of splines

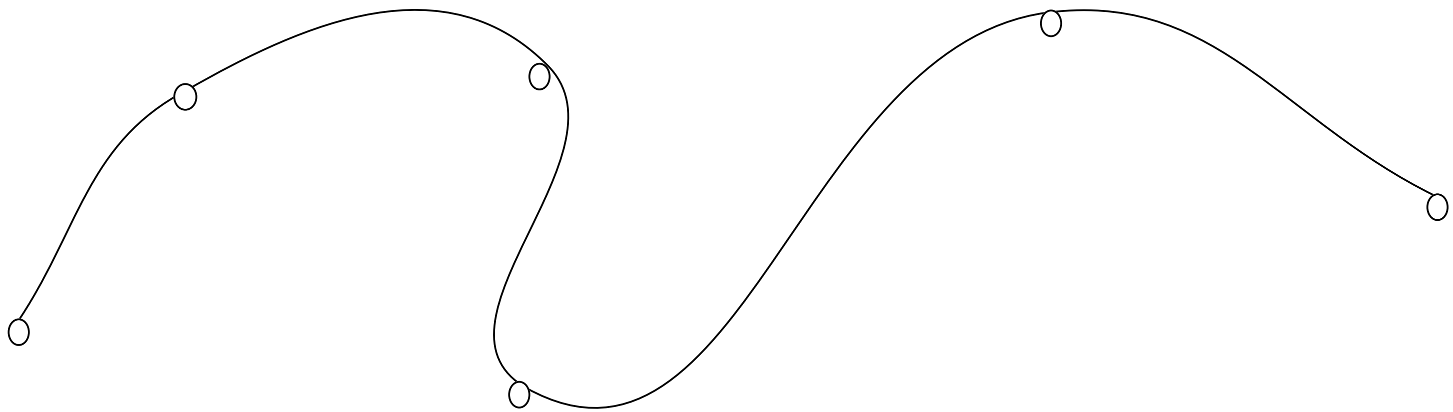
- Designing smooth curves (common in 2D illustrations)
- Modelling smooth surfaces
- Representating of smooth surfaces (converted to polygons in real-time)
- Animation paths

Important application of splines: Text rendering!

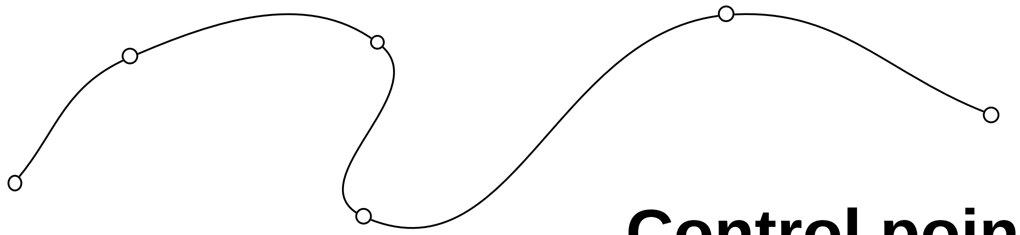
We will return to that subject on a later lecture.

Control points

A spline is specified by a set of control points.

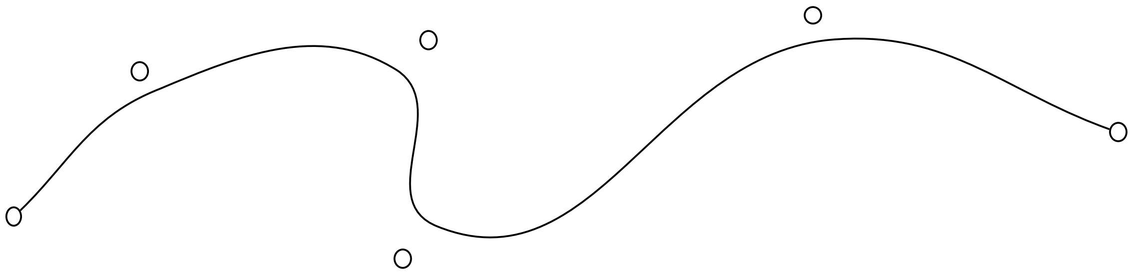


Interpolation spline



Control points on the curve.

Approximation spline



Control points not on the curve.

Parametric representation

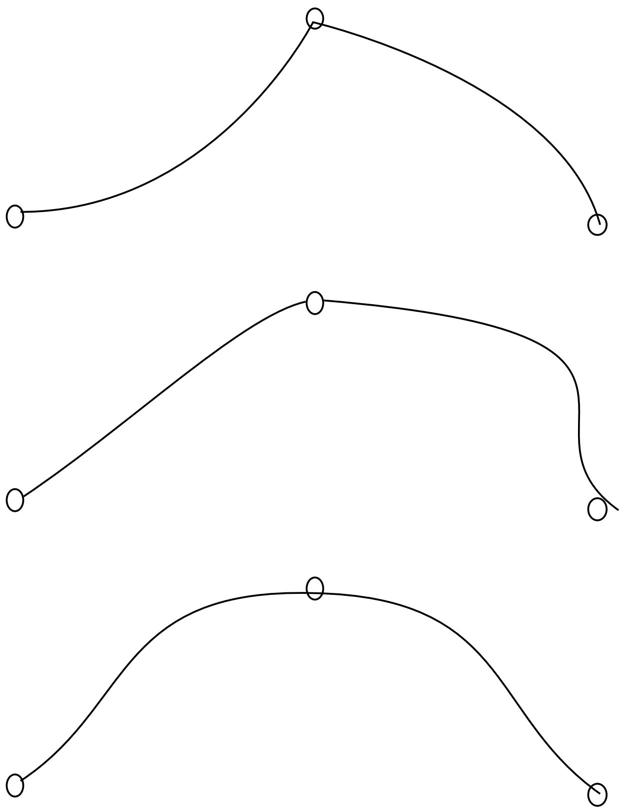
$$x = x(u)$$

$$y = y(u) \qquad u_1 \le u \le u_2$$

$$z = z(u)$$

A set of functions for each coordinate

Parametric continuity



C^0 = continuous position
= the curves meet

C^1 = continuous direction
= the curves meet at same angle

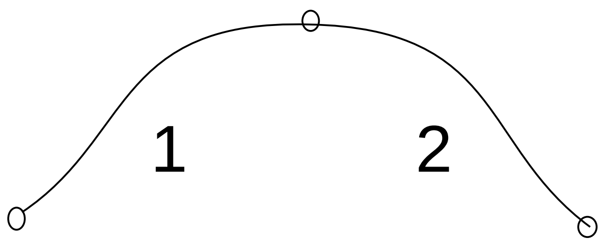
C^2 = continuous curvature
= the curves meet at same bend

Specification of splines by functions

$$x_1(u) = a_{x1}u^3 + b_{x1}u^2 + c_{x1}u + d_{x1}$$

$$y_1(u) = a_{y1}u^3 + b_{y1}u^2 + c_{y1}u + d_{y1}$$

$$z_1(u) = a_{z1}u^3 + b_{z1}u^2 + c_{z1}u + d_{z1}$$

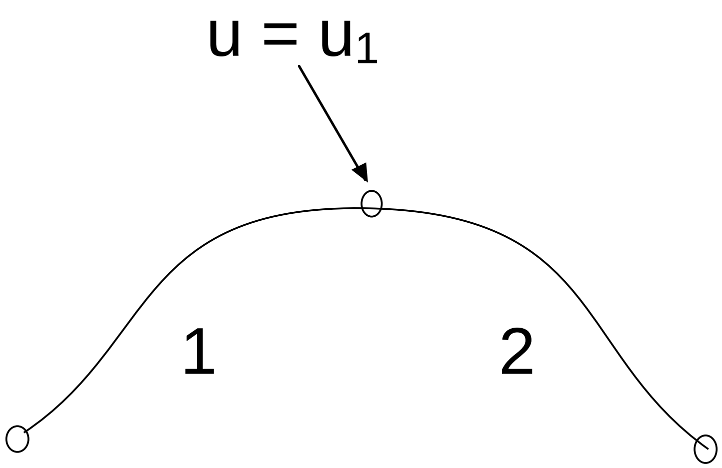


$$x_2(u) = a_{x2}u^3 + b_{x2}u^2 + c_{x2}u + d_{x2}$$

$$y_2(u) = a_{y2}u^3 + b_{y2}u^2 + c_{y2}u + d_{y2}$$

$$z_2(u) = a_{z2}u^3 + b_{z2}u^2 + c_{z2}u + d_{z2}$$

Parametric continuity



C^0:

$x_1(u_1) = x_2(u_1)$

$y_1(u_1) = y_2(u_1)$

$z_1(u_1) = z_2(u_1)$

C^1:

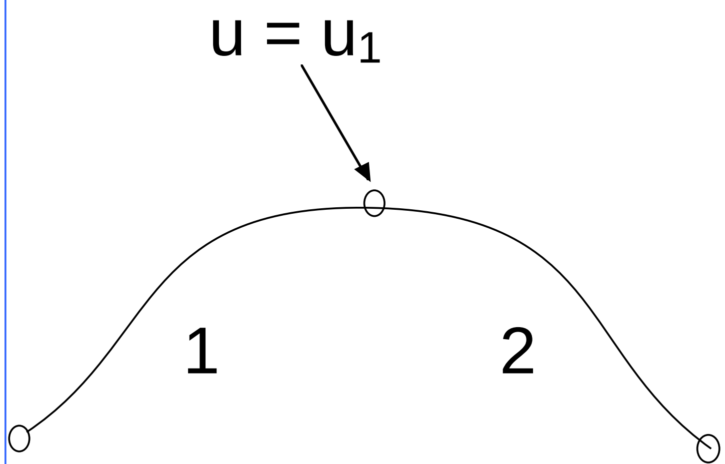
$x'_1(u_1) = x'_2(u_1)$

$y'_1(u_1) = y'_2(u_1)$

$z'_1(u_1) = z'_2(u_1)$

C1: 6 equations per vertex,
12 coefficients per section

Geometric continuity



G^0:

$$x_1(u_1) = x_2(u_1)$$
$$y_1(u_1) = y_2(u_1)$$
$$z_1(u_1) = z_2(u_1)$$

G^1:

$$x'_1(u_1) = k*x'_2(u_1)$$
$$y'_1(u_1) = k*y'_2(u_1)$$
$$z'_1(u_1) = k*z'_2(u_1)'$$

for some k

Essentially one less constraint

Blending functions

Rewrite parametric form to a set of polynomials, one polynomial for each control point

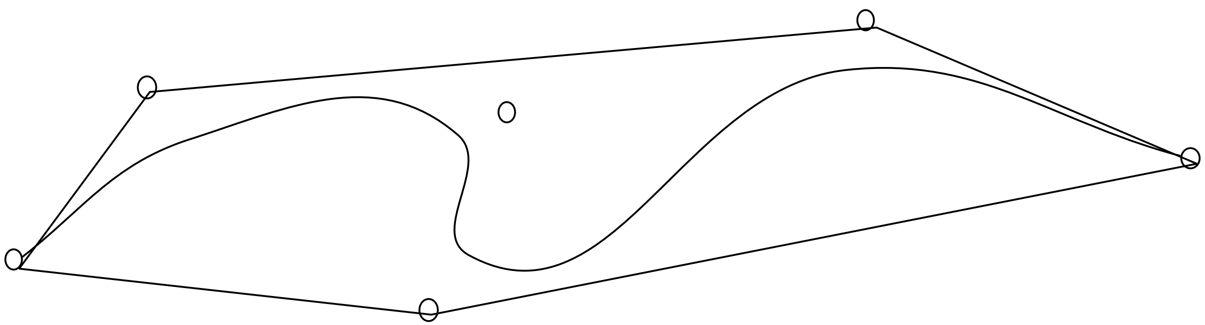
Approximation splines

Use a set of blending functions to blend together control points to points on the curve

Bézier curves
B-splines
NURBS

Common demand on approximations splines:

Stay within the convex hull of the control points!



Convex hull = minimal convex polygon enclosing a specified set of points