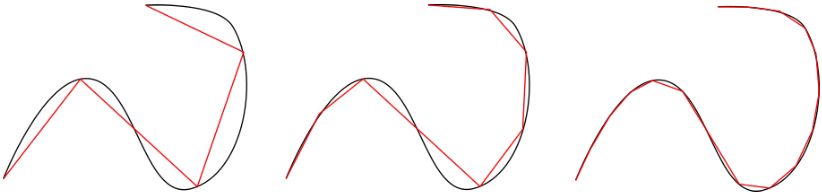
CS 461 - Computer Graphics

Splines

Amal Dev Parakkat

Polylines

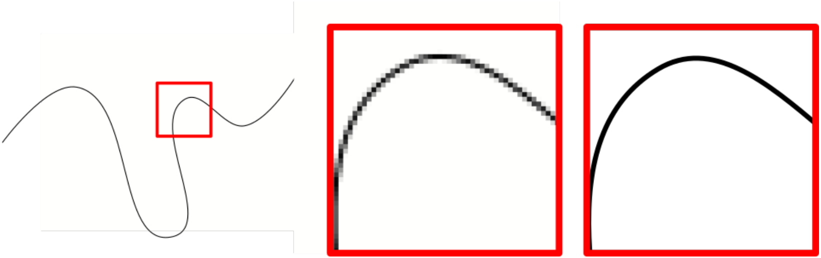
- Path - sequence of points connected by straight lines
- Looks good



Smooth curves

Looks nicer!!!

Concept is not that easy



2D curves

- A continuous set of points
- A mapping from an interval S onto the plane

$$P(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}$$

Spline:

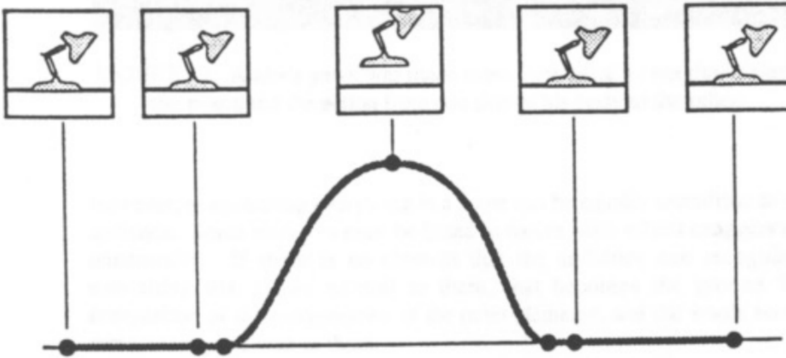
- A type of smooth curve
- Have a wide variety of applications
- Interpolation and Approximation

Physical splines

- Overall idea
- Video: `https://www.youtube.com/watch?v=PwGnyJJCPIg`

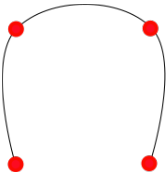
Splines

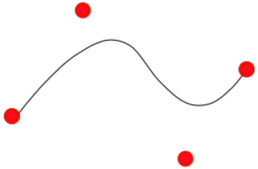
- Easy to store and manipulate
- Results in a smooth parametric curve P(t)



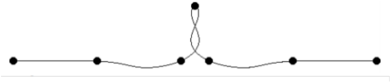
Interpolation vs Approximation

- Interpolation: Goes through every point
- Approximation: Does not go through all points



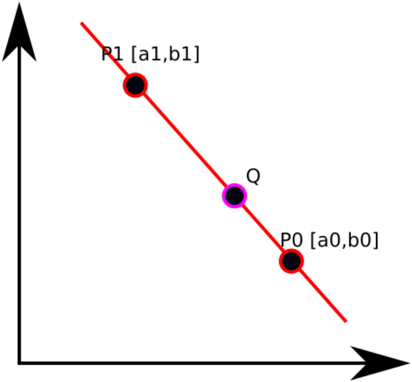


- Difficult to control
- Difficult to predict



- Bezier curves, B-spline curves, Hermite curves...

Bezier Curve



Bezier Curve

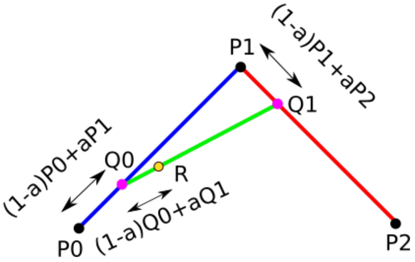
- Affine combination of points
- Line P_0P_1 can be described parametrically

$$\vec{v} \equiv \vec{P_0P_1} = (a_1 - a_0, b_1 - b_0)$$

$$\begin{aligned} Q &= P_0 + \frac{1}{3}\vec{v} \\ &= [a_0, b_0] + \frac{1}{3}(a_1 - a_0, b_1 - b_0) \\ &= \frac{2}{3}[a_0, b_0] + \frac{1}{3}[a_1, b_1] \end{aligned}$$

- A linear combination
- Sum of coefficients = 1

Bezier Curve



- Suppose we have 3 points: P_0, P_1, P_2
- The points Q_0, Q_1 and R can be written as:

$$Q_0 = (1 - a) * P_0 + a * P_1$$
$$Q_1 = (1 - a) * P_1 + a * P_2$$
$$R = (1 - a) * Q_0 + a * Q_1$$

- $a \equiv$ parameter
- Trace R

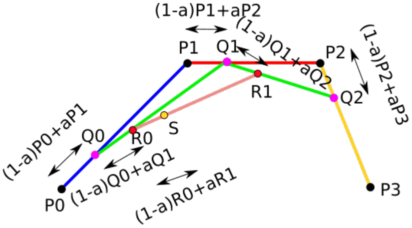
Bezier Curve

- Rewriting the equations:

$$\begin{aligned} R &= (1-a)*[(1-a)P_0 + a*P_1] + a*[(1-a)P_1 + a*P_2] \\ &= (1-a)^2 * P_0 + 2(1-a)a * P_1 + a^2 * P_2 \end{aligned}$$

- Take out the polynumbers:
  $A = (1-a)^2, B = 2(1-a)a, C = a^2$
- Called as Berstein polynomials of degree 2
- Represented using $B_{0,2}, B_{1,2}, B_{2,2}$
- Degree 2 - Quadratic Bezier curve

Bezier Curve - Increasing the degree

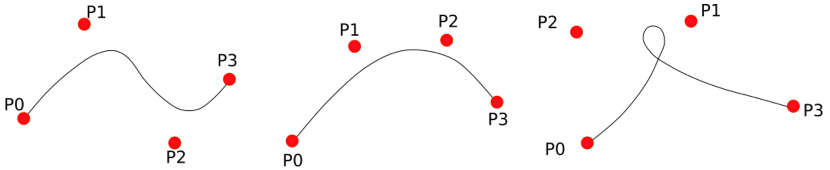


- Suppose we have 4 points:

$$
\begin{aligned}
S &= (1-a) * R_0 + a * R_1 \\
&= (1-a)^3 * P_0 + 3(1-a)^2 a * P_1 + 3(1-a)a^2 * P_2 + a^3 * P_3
\end{aligned}
$$

- $a \equiv$ parameter
- Trace S $\longrightarrow$ Polynomial of degree 3 - Cubic Bezier curve

Cubic Bezier curve



- User specifies 4 control points P_0, P_1, P_2, P_3
- Curve passes through P_0 and P_3
- Approximated by P_1, P_2
- Control points
- Cubic polynomial $\longrightarrow$ Cubic Bezier curve

Cubic Bezier curve

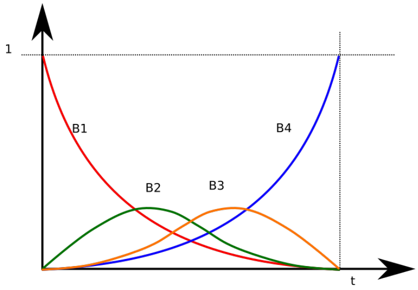
- Rewriting in terms of t

$$P(t) = (1-t)^3 * P_0 + 3t(1-t)^2 * P_1 + 3t^2(1-t) * P_2 + t^3 * P_3$$
$$x(t) = (1-t)^3 * x_0 + 3t(1-t)^2 * x_1 + 3t^2(1-t) * x_2 + t^3 * x_3$$
$$y(t) = (1-t)^3 * y_0 + 3t(1-t)^2 * y_1 + 3t^2(1-t) * y_2 + t^3 * y_3$$

- What happens at t=0 and t=1?
- Curve is tangent at P_0 to $(P_0 - P_1)$ and at P_3 to $(P_2 - P_3)$

Cubic Bezier curve

- P(t) is a weighted combination of the four control points
- P_1 and P_2 never have full control



Cubic Bezier curve

- Can be written in matrix form:

$$\begin{pmatrix} B_{0,3} \\ B_{1,3} \\ B_{2,3} \\ B_{3,3} \end{pmatrix} = \begin{pmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} 1 \\ t \\ t^2 \\ t^3 \end{pmatrix}$$

- In general, Bernstein Polynomials of degree n is:

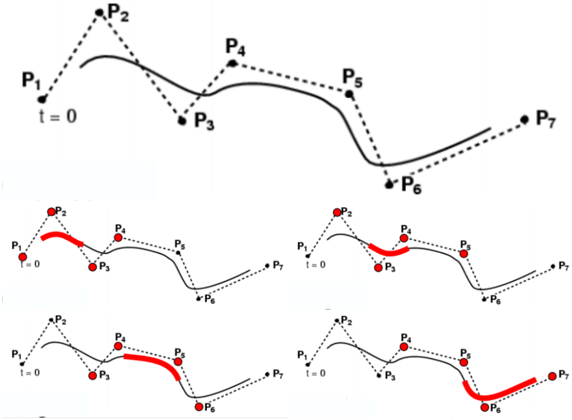
$$B_{i,n} = \frac{n!}{i! * (n-i)!} t^i (1-t)^{n-i}$$

- Bounded by the convex hull
- Drawing algorithm - idea!!! - Video

Connecting Bezier curves

- We want long curves
- Different continuity conditions
- Video: `https://www.youtube.com/watch?v=A3lDRn6jafs`
- Written assignment 1: How can we guarantee this continuity?
- 24^{th} midnight?

Introduction to Cubic B-Splines



- ≥ 4 control points
- Locally cubic
- It does not pass through any control points

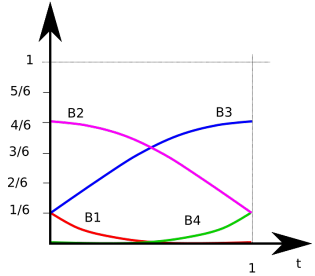
Introduction to Cubic B-Splines

- Basis function:

$$Q(t) = \frac{(1-t)^3}{6} * P_0 + \frac{3t^3 - 6t^2 + 4}{6} * P_1 + \frac{-3t^3 + 3t^2 + 3t + 1}{6} * P_2 + \frac{t^3}{6} * P_3$$

$$\mathrm{B}_{B-Spline} = \frac{1}{6}\begin{pmatrix} 1 & -3 & 3 & -1 \\ 4 & 0 & -6 & 3 \\ 1 & 3 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- By default, Curvature continuous
- Bezier $\Longleftrightarrow$ B-spline



A few important things

- Reordering of seminar dates!!!
- Programming Assignments are going to be:
  - Transforming (scaling and rotating) 3D models - using mouse
  - Ray-tracing a 3D scene
  - Animating a 3D model
  - Texturing a 3D model???
- Project topic and abstract deadline - ASAP - Tips - Springer The Visual Computer - `https://www.springer.com/journal/371`

Project

- Abstract should include:
  - What you are going to do?
  - Technical challenge that you are going to address
  - Why current systems can’t do it?
  - What is the preliminary idea?

Next class

- Next class 24^{th} Sep
- Topic: Transformations