Motivation

- How do we model complex shapes?
  - In this course, only 2D curves, but can be used to create interesting 3D shapes by surface of revolution, lofting etc

- Techniques known as spline curves

- This unit is about mathematics required to draw these spline curves, as in HW 3

- History: From using computer modeling to define car bodies in auto-manufacturing. Pioneers are Pierre Bezier (Renault), de Casteljau (Citroen)
**Bezier Curve (with HW2 demo)**

- **Motivation:** Draw a smooth intuitive curve (or surface) given few key user-specified control points
- Control points (all that user specifies, edits)
- Demo HW 3

**Issues for Bezier Curves**

Main question: Given control points and constraints (interpolation, tangent), how to construct curve?

- Algorithmic: deCasteljau algorithm
- Explicit: Bernstein-Bezier polynomial basis
- 4x4 matrix for cubics
- Properties: Advantages and Disadvantages

**deCasteljau: Linear Bezier Curve**

- Just a simple linear combination or interpolation (easy to code up, very numerically stable)
- Linear (Degree 1, Order 2)
- \( F(0) = P_0, F(1) = P_1 \)
- \( F(u) = (1-u)P_0 + uP_1 \)

**deCasteljau: Quadratic Bezier Curve**

- Quadratic
- Degree 2, Order 3
- \( F(0) = P_0, F(1) = P_2 \)
- \( F(u) = (1-u)^2P_0 + 2u(1-u)P_1 + u^2P_2 \)

**Geometric interpretation: Quadratic**

- Geometric interpretation: Quadratic
Geometric Interpretation: Cubic

DeCasteljau: Cubic Bezier Curve

Summary: deCasteljau Algorithm

DeCasteljau Implementation

Summary of HW2 Implementation

Issues for Bezier Curves
Recap formulae

- Linear combination of basis functions
  - Linear: \[ F(u) = P_0(1-u) + P_u \]
  - Quadratic: \[ F(u) = P_0(1-u)^2 + P_12u(1-u) + P_u^2 \]
  - Cubic: \[ F(u) = P_0(1-u)^3 + P_13u(1-u)^2 + P_23u^2(1-u) + P_u^3 \]

- Explicit form for basis functions? Guess it?

Summary of Explicit Form

- Linear: \[ F(u) = P_0(1-u) + P_u \]
  - Quadratic: \[ F(u) = P_0(1-u)^2 + P_12u(1-u) + P_u^2 \]
  - Cubic: \[ F(u) = P_0(1-u)^3 + P_13u(1-u)^2 + P_23u^2(1-u) + P_u^3 \]

- Degree n: \[ F(u) = \sum B^k(u) \]

Cubic 4x4 Matrix (derive)

\[
F(u) = P_0(1-u)^3 + P_13u(1-u)^2 + P_23u^2(1-u) + P_u^3
\]

\[
= \begin{pmatrix} u^3 & u^2 & u & 1 \end{pmatrix} \begin{pmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{pmatrix}
\]

\[
M = \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}
\]

Issues for Bezier Curves

Main question: Given control points and constraints (interpolation, tangent), how to construct curve?

- Algorithmic: deCasteljau algorithm
- Explicit: Bernstein-Bezier polynomial basis
- 4x4 matrix for cubics
- Properties: Advantages and Disadvantages

Recap formulae

- Linear combination of basis functions
  - Linear: \[ F(u) = P_0(1-u) + P_u \]
  - Quadratic: \[ F(u) = P_0(1-u)^2 + P_12u(1-u) + P_u^2 \]
  - Cubic: \[ F(u) = P_0(1-u)^3 + P_13u(1-u)^2 + P_23u^2(1-u) + P_u^3 \]

- Degree n: \[ F(u) = \sum B^k(u) \]

- Explicit form for basis functions? Guess it?

Binomial coefficients in \([(1-u)u]^n\)

Recap formulae

- Linear combination of basis functions
  - Linear: \[ F(u) = P_0(1-u) + P_u \]
  - Quadratic: \[ F(u) = P_0(1-u)^2 + P_12u(1-u) + P_u^2 \]
  - Cubic: \[ F(u) = P_0(1-u)^3 + P_13u(1-u)^2 + P_23u^2(1-u) + P_u^3 \]

- Degree n: \[ F(u) = \sum B^k(u) \]

- Explicit form for basis functions? Guess it?

Binomial coefficients in \([(1-u)+u]^n\)

Recap formulae

- Linear combination of basis functions
  - Linear: \[ F(u) = P_0(1-u) + P_u \]
  - Quadratic: \[ F(u) = P_0(1-u)^2 + P_12u(1-u) + P_u^2 \]
  - Cubic: \[ F(u) = P_0(1-u)^3 + P_13u(1-u)^2 + P_23u^2(1-u) + P_u^3 \]

- Degree n: \[ F(u) = \sum B^k(u) \]

- Explicit form for basis functions? Guess it?

Binomial coefficients in \([(1-u)+u]^n\)
Issues for Bezier Curves

Main question: Given control points and constraints (interpolation, tangent), how to construct curve?

- Algorithmic: deCasteljau algorithm
- Explicit: Bernstein-Bezier polynomial basis
- 4x4 matrix for cubics
- Properties: Advantages and Disadvantages

Properties (brief discussion)

- Demo of HW 3
- Interpolation: End-points, but approximates others
- Single piece, moving one point affects whole curve (no local control as in B-splines later)
- Invariant to translations, rotations, scales etc. That is, translating all control points translates entire curve
- Easily subdivided into parts for drawing (next lecture): Hence, Bezier curves easiest for drawing