Topics in Computer Graphics
Chap 7: Polynomial Curve Constructions

fall, 2011

University of Seoul
School of Computer Science
Minho Kim

Table of contents

Motivation

- Given n points in 2D/3D,
  - Interpolation: find a (polynomial) curve that passes through the data exactly.
    - Aitken's algorithm (7.1)
    - Lagrange polynomials (7.2)
    - The Vandermonde approach (7.3)
    - The Newton form and forward differencing (7.11)
  - Approximation: find a (polynomial) curve that goes through the data closely in a reasonable way.
    - Least squares approximation (7.8)
    - Smoothing equations (7.9)
- Normal/tangent data in addition to position data.
  - Cubic/quintic Hermite interpolation (7.5&7.6)
  - Point-normal interpolation (7.7)

Outline

Aitken's Algorithm

- Problem: Given points $\{\mathbf{p}_i\}_{i=0}^n$ with corresponding parameter values $\{t_i\}_{i=0}^n$, find a curve $\mathbf{p}(t)$ such that

$$\mathbf{p}(t_i) = \mathbf{p}_i, \quad i = 0, \ldots, n.$$

Aitken's Algorithm (cont'd)

1. Base case: Let

$$\mathbf{p}_i^1(t) = \frac{t_{i+1} - t}{t_{i+1} - t_i}\mathbf{p}_i + \frac{t - t_i}{t_{i+1} - t_i}\mathbf{p}_{i+1}, \quad i = 0, \ldots, n-1.$$

   - $\mathbf{p}_i^1(t)$ is a line segment for $t_i \leqslant t \leqslant t_{i+1}$.
   - $\mathbf{p}_i^1(t)$ interpolates $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$: $\mathbf{p}_i^1(t_i) = \mathbf{p}_i$ and $\mathbf{p}_i^1(t_{i+1}) = \mathbf{p}_{i+1}$.
2. Let $\mathbf{p}_0^{n-1}(t)$ is a polynomial of degree $n-1$ that interpolates $\mathbf{p}_0, \ldots, \mathbf{p}_{n-1}$ and $\mathbf{p}_1^{n-1}(t)$ is one that interpolates $\mathbf{p}_1, \ldots, \mathbf{p}_n$.
3. Then

$$\mathbf{p}_0^n(t) = \frac{t_n - t}{t_n - t_0}\mathbf{p}_0^{n-1}(t) + \frac{t - t_0}{t_n - t_0}\mathbf{p}_1^{n-1}(t)$$

   - is a polynomial curve of degree n and
   - interpolates $\mathbf{p}_0, \ldots, \mathbf{p}_n$.

Aitken's Algorithm (cont'd)

With $\mathbf{p}_i^0(t) := \mathbf{p}_i$, the interpolating curve $\mathbf{p}_0^n(t)$ is defined recursively as

$$\mathbf{p}_i^r(t) = \frac{t_{i+r} - t}{t_{i+r} - t_i}\mathbf{p}_i^{r-1}(t) + \frac{t - t_i}{t_{i+r} - t_i}\mathbf{p}_{i+1}^{r-1}(t), \qquad \begin{cases} r = 1, \dots, n \\ i = 0, \dots, n - r. \end{cases}$$

Properties

- Affine invariance
- Linear precision
- No convex hull property
  Is it possible for a smooth interpolating curve has a convex hull property?
- No variation diminishing property
- Does not answer if such polynomial curve is unique.
- Not a closed form

Outline

Lagrange Polynomials

The interpolating curve is given by

$$\mathbf{p}(t) = \sum_{i=0}^{n} \mathbf{p}_i L_i^n(t)$$

where $L_i^n(t)$ are *Lagrange polynomials* defined as

$$L_i^n(t) := \frac{\prod_{\substack{j=0 \\ j\neq i}}^{n} (t - t_j)}{\prod_{\substack{j=0 \\ j\neq i}}^{n} (t_i - t_j)}$$

- Lagrange polynomials are *cardinal*:

  $$L_i^n(t_j) = \delta_{i,j}$$

  where $\delta_{i,j}$ is the Kronecker delta.
- $\{L_i^n(t)\}_{i=0}^n$ form a basis of all polynomials of degree n
- Interpolating polynomial (of degree n) is unique
- $\sum_{i=0}^{n} L_i^n(t) \equiv 1(t)$

Outline

The Vandermonde Approach

What is the monomial form

$$\mathbf{p}^n(t) = \sum_{j=0}^{n} \mathbf{a}_j t^j$$

of the interpolating polynomial?

The Vandermonde Approach (cont'd)

1.

$$
\begin{aligned}
\mathbf{p}^n(t_0) &= \mathbf{p}_0 = \mathbf{a}_0 + \mathbf{a}_1 t_0 + \cdots + \mathbf{a}_n t_0^n \\
\mathbf{p}^n(t_1) &= \mathbf{p}_1 = \mathbf{a}_0 + \mathbf{a}_1 t_1 + \cdots + \mathbf{a}_n t_1^n \\
&\vdots \\
\mathbf{p}^n(t_n) &= \mathbf{p}_n = \mathbf{a}_0 + \mathbf{a}_1 t_n + \cdots + \mathbf{a}_n t_n^n
\end{aligned}
$$

2. $\begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_n \end{bmatrix} = \begin{bmatrix} 1 & t_0 & \cdots & t_0^n \\ 1 & t_1 & \cdots & t_1^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & t_n & \cdots & t_n^n \end{bmatrix} \begin{bmatrix} \mathbf{a}_0 \\ \mathbf{a}_1 \\ \vdots \\ \mathbf{a}_n \end{bmatrix} \quad \rightarrow \quad \mathbf{p} = T\mathbf{a}$
3. If $\{t_j\}_{j=0}^n$ are distinct, $\det T \neq 0$. (Why?)
   $\rightarrow \det T$ is the Vandermonde of the interpolation problem
4. Solution: $\mathbf{a} = T^{-1}\mathbf{p}$

Generalized Vandermonde

For some basis $\{F_j^n(t)\}_{j=0}^n$, (e.g. Bernstein basis) the interpolating polynomial is expressed as

$$\mathbf{p}^n(t) = \sum_{j=0}^{n} \mathbf{c}_j F_j^n(t)$$

and the coefficients $\{\mathbf{c}_j\}_{j=0}^n$ can be found as

$$\rightarrow \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \\ \vdots \\ \mathbf{c}_n \end{bmatrix} = \begin{bmatrix} F_0^n(t_0) & F_1^n(t_0) & \cdots & F_n^n(t_0) \\ F_0^n(t_1) & F_1^n(t_1) & \cdots & F_n^n(t_1) \\ \vdots & \vdots & \ddots & \vdots \\ F_0^n(t_n) & F_1^n(t_n) & \cdots & F_n^n(t_n) \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_n \end{bmatrix}$$

Outline

Limits of Lagrange Interpolation

- Small change of input results in large change in output
  - The process is *ill-conditioned*.
  - Polynomial interpolation is not *shape preserving*.
  - Runge phenomonen
- Expensive to solve
  - for construction (solving dense linear system)
  - for evaluation (for high n)
- Piecewise schemes are better! (e.g. spline interpolation)

Outline

Cubic Hermite Interpolation

Find the cubic polynomial that interpolates the following:

$$\begin{aligned}
\mathbf{p}(0) &= \mathbf{p}_0 \in \mathbb{E}^3 \\
\dot{\mathbf{p}}(0) &= \mathbf{m}_0 \in \mathbb{R}^3 \\
\dot{\mathbf{p}}(1) &= \mathbf{m}_1 \in \mathbb{R}^3 \\
\mathbf{p}(1) &= \mathbf{p}_1 \in \mathbb{E}^3,
\end{aligned}$$

Easier to work with Bézier form

$$\mathbf{p}(t) = \sum_{j=0}^{3} \mathbf{b}_j B_j^3(t)$$

1. $\mathbf{b}_0 = \mathbf{p}_0$ and $\mathbf{b}_3 = \mathbf{p}_1$
2. $\dot{\mathbf{p}}(0) = 3\Delta\mathbf{b}_0 = \mathbf{m}_0$ and $\dot{\mathbf{p}}(1) = 3\Delta\mathbf{b}_2 = \mathbf{m}_1$
   $\rightarrow \mathbf{b}_1 = \mathbf{p}_0 + \frac{1}{3}\mathbf{m}_0$ and $\mathbf{b}_2 = \mathbf{b}_1 - \frac{1}{3}\mathbf{m}_1$

Cubic Hermite Interpolation: Cardinal Form

$$\mathbf{p}(t) = \mathbf{p}_0 B_0^3(t) + \left(\mathbf{p}_0 + \frac{1}{3}\mathbf{m}_0\right) B_1^3(t) + \left(\mathbf{p}_1 - \frac{1}{3}\mathbf{m}_1\right) B_2^3(t) + \mathbf{p}_1 B_3^3(t)$$

$$\rightarrow \mathbf{p}(t) = \mathbf{p}_0 H_0^3(t) + \mathbf{m}_0 H_1^3(t) + \mathbf{m}_1 H_2^3(t) + \mathbf{p}_1 H_3^3(t)$$

$\rightarrow$

$$\begin{aligned} H_0^3(t) &= B_0^3(t) + B_1^3(t) \\ H_1^3(t) &= \frac{1}{3} B_1^3(t) \\ H_2^3(t) &= -\frac{1}{3} B_2^3(t) \\ H_3^3(t) &= B_2^3(t) + B_3^3(t) \end{aligned}$$

Cubic Hermite Interpolation: Cardinal Form

$$
\begin{array}{llll}
H_0^3(0) = 1 & \frac{d}{dt}H_0^3(0) = 0 & \frac{d}{dt}H_0^3(1) = 0 & H_0^3(1) = 0 \\
H_1^3(0) = 0 & \frac{d}{dt}H_1^3(0) = 1 & \frac{d}{dt}H_1^3(1) = 0 & H_1^3(1) = 0 \\
H_2^3(0) = 0 & \frac{d}{dt}H_2^3(0) = 0 & \frac{d}{dt}H_2^3(1) = 1 & H_2^3(1) = 0 \\
H_3^3(0) = 0 & \frac{d}{dt}H_3^3(0) = 0 & \frac{d}{dt}H_3^3(1) = 0 & H_3^3(1) = 1
\end{array}
$$

- $H_0^3(t) + H_3^3(t) \equiv 1(t)$ (Why?)
- Not invariant under affine domain transformations
- Not symmetric

Outline

Outline

Outline

Least Squares Approximation

- $P+1$ number of data points can be interpolated by a polynomial of degree n if $d \geqslant P$.
- What if $n < P$? $\rightarrow$ approximation required
- How to measure the closeness of the approximating curve and the data points?
- *least squares approximation*: To find a polynomial curve $\mathbf{x}(t)$ that minimizes

$$f(\mathbf{c}_0, \ldots, \mathbf{c}_n) = \sum_{j=0}^{P} \|\mathbf{x}(t_j) - \mathbf{p}_j\|^2.$$

Least Squares Approximation (cont'd)

1. The approximating curve is expressed as

$$\mathbf{x}(t) = c_0 C_0^n(t) + \cdots + c_n C_n^n(t)$$

   for some basis $\{C_j^n\}_{j=0}^n$.

2. Assuming $\mathbf{x}(t)$ interpolates all the points $\{\mathbf{p}_j\}_{j=0}^P$, we get a linear system

$$\begin{bmatrix} C_0^n(t_0) & \cdots & C_n^n(t_0) \\ \vdots & \ddots & \vdots \\ C_0^n(t_P) & \cdots & C_n^n(t_P) \end{bmatrix} \begin{bmatrix} \mathbf{c}_0 \\ \vdots \\ \mathbf{c}_n \end{bmatrix} = \begin{bmatrix} \mathbf{p}_0 \\ \vdots \\ \mathbf{p}_P \end{bmatrix} \quad \rightarrow \quad M\mathbf{C} = \mathbf{P}$$

   where

$$M \in \mathbb{R}^{(P+1)\times(n+1)}, \mathbf{C} \in \mathbb{R}^{(n+1)\times 3}, \mathbf{P} \in \mathbb{R}^{(P+1)\times 3}, \quad P > n$$

   - Linear systems for each component
   - underdetermined (no solution in general)

Least Squares Approximation (cont'd)

3. Can be converted to a solvable linear system by multiplying M^T:

$$M^T M \mathbf{C} = M^T \mathbf{P}$$

- $M^T M$ is always invertible. (Why?)
- The solution minimizes

$$f(\mathbf{c}_0, \ldots, \mathbf{c}_n) = \sum_{j=0}^{P} \|\mathbf{x}(t_j) - \mathbf{p}_j\|^2.$$

  (Why?)
  The solution satisfies

$$\frac{\partial f}{\partial \mathbf{c}_k^d} = 0, \quad k = 0, \ldots, n, d = 1, 2, 3.$$

Outline

Smoothing Equations

- Minimizing the least square error does not guarantee a good curve. (Fig 7.8)
- We want a curve that does not "wiggle" much. (How?)
- If the control polygon behaves nicely, then so does the curve.
- How to minimizes wiggling (of the polygon)? → minimize the 2nd differences $\Delta^2 \mathbf{b}_i$

$$\left.\begin{aligned} \mathbf{b}_0 - 2\mathbf{b}_1 + \mathbf{b}_2 &= 0 \\ &\vdots \\ \mathbf{b}_{n-2} - 2\mathbf{b}_{n-1} + \mathbf{b}_n &= 0 \end{aligned}\right. \quad \rightarrow S\mathbf{B} = \mathbf{0}$$

Smoothing Equations (cont'd)

- Combined with the least squares problem,

$$\begin{bmatrix} M \\ S \end{bmatrix} \mathbf{B} = \begin{bmatrix} \mathbf{P} \\ \mathbf{0} \end{bmatrix}$$

- Balancing the error and shape

$$\begin{bmatrix} (1-\alpha)M \\ \alpha S \end{bmatrix} \mathbf{B} = \begin{bmatrix} (1-\alpha)\mathbf{P} \\ \mathbf{0} \end{bmatrix} \quad 0 \leqslant \alpha \leqslant 1$$

  - Larger α for noisier data

Outline

Designing with Bézier Curves

- "What if we want to move a point on the curve to change the shape?"
- Given a Bézier curve

$$\mathbf{x}(t) = \sum_{i=0}^{n} \mathbf{b}_i B_i^n(t),$$

if we change a point $\mathbf{x}(\hat{t})$ to $\mathbf{y}$, how should we change $\{\mathbf{b}_i\}$ to $\{\hat{\mathbf{b}}_i\}$ such that the curve passes through $\mathbf{y}$ at $t = \hat{t}$?

$$\mathbf{y} = \sum_{i=0}^{n} \hat{\mathbf{b}}_i B_i^n(\hat{t})$$

  - Assume $\mathbf{b}_0$ and $\mathbf{b}_n$ are unchanged.
  - Underdetermined: 1 equation and $n - 1$ unknowns

Designing with Bézier Curves (cont'd)

1.
$$\begin{bmatrix} B_1^n(\hat{t}) & \cdots & B_{n-1}^n(\hat{t}) \end{bmatrix} \begin{bmatrix} \hat{\mathbf{b}}_1 \\ \vdots \\ \hat{\mathbf{b}}_{n-1} \end{bmatrix} = \mathbf{y} - \mathbf{b}_0 B_0^n(\hat{t}) - \mathbf{b}_n B_n^n(\hat{t}) =: \mathbf{z}$$

$$\to A\hat{\mathbf{B}} = \mathbf{z}$$

2. Assume all the points $\mathbf{b}_1, \ldots, \mathbf{b}_{n-1}$ move along (unknown) $\mathbf{c}$ direction:

$$\hat{\mathbf{B}} = \mathbf{B} + A^T\mathbf{c} \to A\hat{\mathbf{B}} = A\mathbf{B} + AA^T\mathbf{c} \to \mathbf{z} - A\mathbf{B} = AA^T\mathbf{c}$$

3. Since AA^T is a scalar, with $a := AA^T$

$$\mathbf{c} = \frac{1}{a}(\mathbf{y} - \mathbf{x}) \to \hat{\mathbf{B}} = \mathbf{B} + \frac{1}{a}A^T(\mathbf{y} - \mathbf{x}).$$

Outline

Aitken's Algorithm

Lagrange Polynomials

The Vandermonde Approach

Limits of Lagrange Interpolation

Cubic Hermite Interpolation

Quintic Hermite Interpolation

Point-Normal Interpolation

Least Squares Approximation

Smoothing Equations

Designing with Bézier Curves

The Newton Form and Forward Differencing

The Newton Form of Interpolating Polynomial

- For equally spaced parameter intervals: $t_{i+1} = t_i + h$
- For cubic case,

$$\begin{aligned}\mathbf{p}(t) =& \mathbf{p}_0 + \frac{1}{h}(t - t_0)\Delta\mathbf{p}_0 + \frac{1}{2!h^2}(t - t_0)(t - t_1)\Delta^2\mathbf{p}_0 \\ &+ \frac{1}{3!h^3}(t - t_0)(t - t_1)(t - t_2)\Delta^3\mathbf{p}_0\end{aligned}$$

where

$$\Delta^i\mathbf{p}_j := \Delta^{i-1}\mathbf{p}_{j+1} - \Delta^{i-1}\mathbf{p}_j \quad \text{and} \quad \Delta^0\mathbf{p}_j = \mathbf{p}_j.$$

- With $g := 1/h$,

$$\begin{array}{llll}\boxed{\mathbf{p}_0} & & & \\ \mathbf{p}_1 & \boxed{g\Delta\mathbf{p}_0} & & \\ \mathbf{p}_2 & g\Delta\mathbf{p}_1 & \boxed{g^2\Delta^2\mathbf{p}_0} & \\ \mathbf{p}_3 & g\Delta\mathbf{p}_2 & g^2\Delta^2\mathbf{p}_1 & \boxed{g^3\Delta^3\mathbf{p}_0}\end{array}$$

Fast Evaluation for Plotting

- Efficient way to evaluate $\mathbf{p}(t_2+h), \mathbf{p}(t_2+2h), \dots$

$$
\begin{array}{llll}
\mathbf{p}_0 & & & \\
\mathbf{p}_1 & g\Delta\mathbf{p}_0 & & \\
\mathbf{p}_2 & g\Delta\mathbf{p}_1 & g^2\Delta^2\mathbf{p}_0 & \\
\mathbf{p}_3 & g\Delta\mathbf{p}_2 & g^2\Delta^2\mathbf{p}_1 & g^3\Delta^3\mathbf{p}_0 \\
\mathbf{p}_4 & g\Delta\mathbf{p}_3 & g^2\Delta^2\mathbf{p}_2 & g^3\Delta^3\mathbf{p}_1 \\
\mathbf{p}_5 & g\Delta\mathbf{p}_4 & g^2\Delta^2\mathbf{p}_3 & g^3\Delta^3\mathbf{p}_2 \\
\vdots & \vdots & \vdots & \vdots
\end{array}
$$

- $\{g\Delta^3\mathbf{p}_j\}$ are all equal. (Why?)
- From right to left

$$
\begin{aligned}
g^2\Delta^2\mathbf{p}_2 = g^3\Delta^3\mathbf{p}_1 + g^2\Delta^2\mathbf{p}_1 &\to g\Delta\mathbf{p}_3 = g^2\Delta^2\mathbf{p}_2 + g\Delta\mathbf{p}_2 \\
&\to \mathbf{p}_4 = g\Delta\mathbf{p}_3 + \mathbf{p}_3
\end{aligned}
$$

- In general, with $\mathbf{q}_j^i := g^i\Delta^i\mathbf{p}_j$,

$$
\mathbf{q}_j^i = \mathbf{q}_{j-1}^{i+1} + \mathbf{q}_{j-1}^i, \quad i = 2, 1, 0
$$

and $\mathbf{p}_j = \mathbf{q}_j^0$.

Fast for Plotting (cont'd)

- No multiplication involved! (after start-up phase) → extremely fast
- Given a polynomial in cubic Bézier form,
  1. Compute initial four points.
  2. Evaluate sequence of points and plot using line segments.
- Roundoff error is accumulated → The curve may deviates more and more...