Splines (4)

Xin (Shane) Li

Nov. 3, 2009

Parametric Solids

Can represent heterogeneous volumetric data:

- Tricubic solid

$$\mathbf{p}(u,v,w) = \sum_{i=0}^{3}\sum_{j=0}^{3}\sum_{k=0}^{3}\mathbf{a}_{ijk}u^i v^j w^k$$

$$u,v,w \in [0,1]$$

- Bezier solid

$$\mathbf{p}(u,v,w) = \sum_{i}\sum_{j}\sum_{k}\mathbf{p}_{ijk}B_i(u)B_j(v)B_k(w)$$

- B-spline solid

$$\mathbf{p}(u,v,w) = \sum_{i}\sum_{j}\sum_{k}\mathbf{p}_{ijk}B_{i,I}(u)B_{j,J}(v)B_{k,K}(w)$$

- NURBS solid

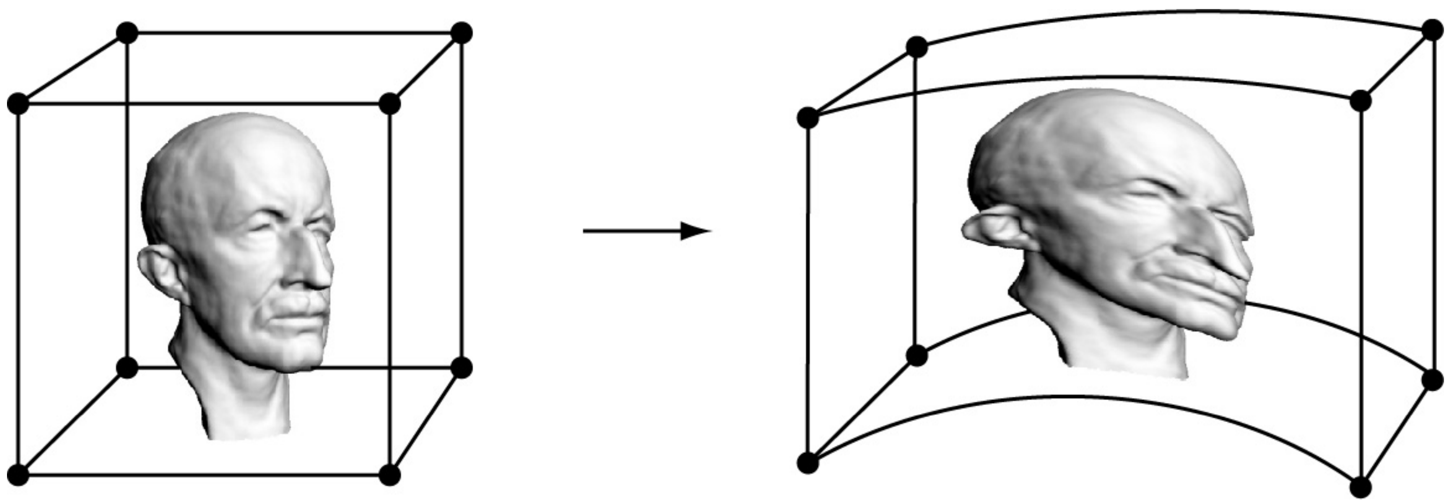
$$\mathbf{p}(u,v,w) = \frac{\sum_{i}\sum_{j}\sum_{k}\mathbf{p}_{ijk}q_{ijk}B_{i,I}(u)B_{j,J}(v)B_{k,K}(w)}{\sum_{i}\sum_{j}\sum_{k}q_{ijk}B_{i,I}(u)B_{j,J}(v)B_{k,K}(w)}$$

Free-form Deformation

- Geometric objects are embedded into a space
- The surrounding space is represented by using commonly-used, popular splines
- Free-form deformation of the surrounding space
- All the embedded (geometric) objects are deformed accordingly, the quantitative measurement of deformation is obtained from the displacement vectors of the trivariate splines that define the surrounding space
- Essentially, the deformation is governed by the trivariate, volumetric splines
- Popular in graphics and related fields

(Will be discussed in EE7000 course.)

Free-form Deformations



(courtesy of Pauly et al.)

Overview: Spline Fitting

- Introduction and Classification
- Interpolation
  - Global interpolation
    - Global curve interpolation
    - Global surface interpolation
  - Local interpolation
    - Local curve interpolation
    - Local surface interpolation
- Approximation
  - Global approximation
    - Least square curve approximation
    - Least square surface approximation
  - Local approximation

Piegl, L., Interactive data interpolation by rational Bézier curves, *IEEE Comput. Graph. and Appl.*, Vol. 7, No. 4, pp. 45–58, 1987.

Chou, J., and Piegl, L., Data reduction using cubic rational B-splines, *IEEE Comput. Graph. and Appl.*, Vol. 12, No. 3, pp. 60–68, 1992.

Introduction

- We have discussed forms and properties of splines
- This class: How to really construct them from given geometric data, so that the given data can be converted to the spline representation?
  - For some specific data with a known equation representation, we can construct the spline (e.g. a circle)
  - For most free-form shapes → fitting

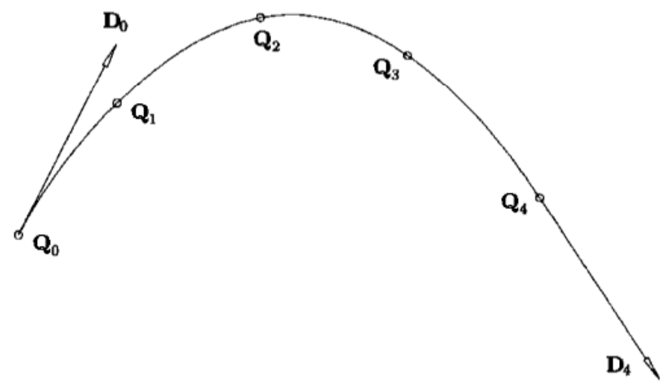
- Two types of fitting problems:
  - Interpolation
  - Approximation
- Two categories of fitting algorithms
  - Global
  - Local

Introduction (cont.)

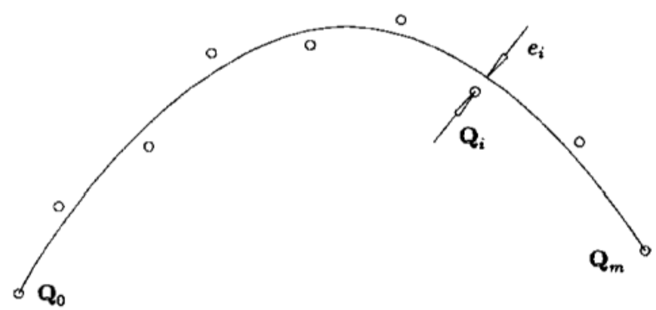
- Many fitting algorithms, hundreds of papers
  - → Which is the “right” answer?
  - → Given data never specifies a unique solution (infinitely many NURBS can be generated to interpolate/approximate the given data in mathematically correct ways)
- We can seek appropriate:
  1) Degree of splines (or it can be given by the user as the requirement)
     - If we want C^r continuity, then degree must satisfy: $p >= r + 1$
     - p=r+1 is generally adequate for interpolation, but p>r+1 may produce better results (e.g. less control points in approximation)
  2) Control points: most algorithms seek efficient way for their placement
  3) Knots : many methods on their choosing
  4) Weight : Little work on its setting

Problem Classification

- Interpolation
  - Construct a spline that satisfies the given data precisely
  - i.e. the curve passes through all given points



- Approximation
  - Construct a spline which do not necessarily satisfy the given data precisely, but only approximately
  - Only try to capture the shape not the wiggle, due to measurement of computational noise
  - Spline should be bounded by a preset deviation, and sometimes should satisfy given constraint points precisely



Algorithm Classification

- Global algorithms
  - A system of equations or an optimization problem is set up globally
  - <u>If:</u> (1) the given data consists of only points and derivatives, and
    (2) (we preset degree, knots, and weights) only solve the control points as unknowns
  - <u>Then:</u> the system is linear and can be efficiently solved.
  - <u>Otherwise:</u> when we need to fit curvature, when we need to solve knots/weights… → nonlinear optimization, and a perturbation of one input locally can change the shape globally
- Local algorithms
  - Construct the spline segment-wise, using only local data for each step
  - ✓ A perturbation only changes the shape locally
  - ✓ Algorithms are usually computationally less expensive
  - ✓ Can deal with cusps, straight segments, and other local data anomalies better
  - ❖ Need to work on getting desired continuity at segment boundaries
  - ❖ Multiple interior knots

Global Interpolation – Curve (1)

- **Input:** a set of points $\{\mathbf{Q}_k\}$, $k = 0, \ldots, n$
- **Output:** an p-degree nonrational B-spline curve, interpolating them
- If we assign
  - a parameter value $\bar{u}_k$, to each $\mathbf{Q}_k$
  - Appropriate knots vector $U = \{u_0, \ldots, u_m\}$
- Then we **solve** a (n+1)*(n+1) linear equation system:

$$\mathbf{Q}_k = \mathbf{C}(\bar{u}_k) = \sum_{i=0}^{n} N_{i,p}(\bar{u}_k)\mathbf{P}_i$$

(with the same coefficient matrix, solve a linear system on each axis direction)

Basis function, we denoted it as $B_{i,p}$ in previous slides. Here uses N to denote the non-uniform knots.

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \le u < u_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u)$$

Global Interpolation – Curve (2)

- **Choosing parameter value** $\bar{u}_k$ (assuming we put parameter inside the range $u \in [0,1]$):

  Three common methods:

  1) Equally spaced:

  $$\bar{u}_0 = 0 \qquad \bar{u}_n = 1$$
  $$\bar{u}_k = \frac{k}{n} \qquad k = 1, \ldots, n-1$$

  → not recommended, can produce erratic shapes when the data is unevenly spaced

  2) Chord length:

  $$d = \sum_{k=1}^{n} |\mathbf{Q}_k - \mathbf{Q}_{k-1}|$$
  $$\bar{u}_0 = 0 \qquad \bar{u}_n = 1 \qquad \bar{u}_k = \bar{u}_{k-1} + \frac{|\mathbf{Q}_k - \mathbf{Q}_{k-1}|}{d} \qquad k = 1, \ldots, n-1$$

  → Most widely used, generally adequate, approximates a uniform parameterization

  3) Centripetal method:

  $$d = \sum_{k=1}^{n} \sqrt{|\mathbf{Q}_k - \mathbf{Q}_{k-1}|}$$
  $$\bar{u}_0 = 0 \qquad \bar{u}_n = 1 \qquad \bar{u}_k = \bar{u}_{k-1} + \frac{\sqrt{|\mathbf{Q}_k - \mathbf{Q}_{k-1}|}}{d} \qquad k = 1, \ldots, n-1$$

  → Better results when the data takes very shape turns

Global Interpolation – Curve (3)

- **Selecting knots vector** U:
  - Two common methods:

1) Equally spaced:

$$u_0 = \cdots = u_p = 0 \qquad u_{m-p} = \cdots = u_m = 1$$

$$u_{j+p} = \frac{j}{n-p+1} \qquad j = 1, \ldots, n-p$$

→ not recommended, can produce a singular system of equations

2) Parameter Averaging:

$$u_0 = \cdots = u_p = 0 \qquad u_{m-p} = \cdots = u_m = 1$$

$$u_{j+p} = \frac{1}{p} \sum_{i=j}^{j+p-1} \bar{u}_i \qquad j = 1, \ldots, n-p$$

- Knots reflect the distribution of the parameter value
- The linear system is positive, semibandwidth < p

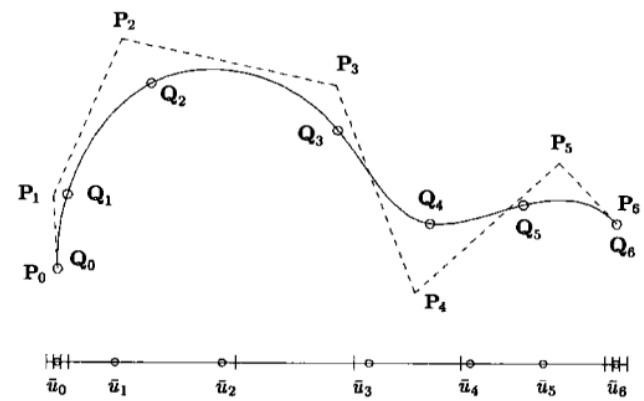
[De Boor, C., *A Practical Guide to Splines*, New York: Springer-Verlag, 1978.]

Global Interpolation – Curve (4)

- **Examples:**

(1)

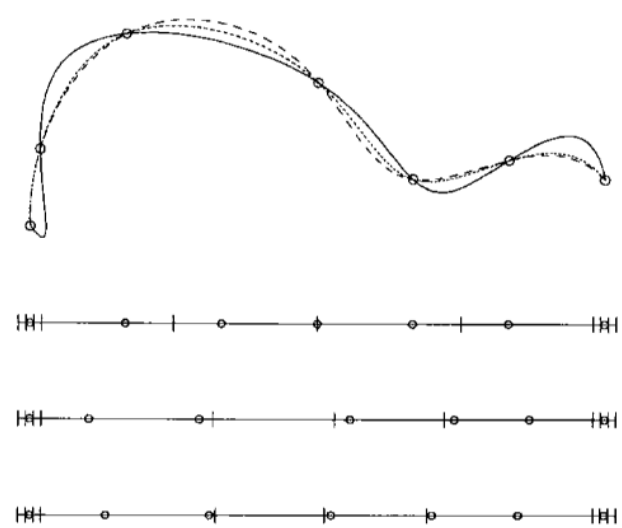
- Parameters chosen by the chord length method
- Knots obtained by parameter averaging



(2)

A cubic curve interpolating data with different parameterizations and knots:

- Uniform parameters + uniform knots (solid curve, top knot vector)
- Chord length parameters + parameter averaging (dashed, middle knot vector)
- Centripetal parameters + parameter averaging (dotted, bottom knot vector)



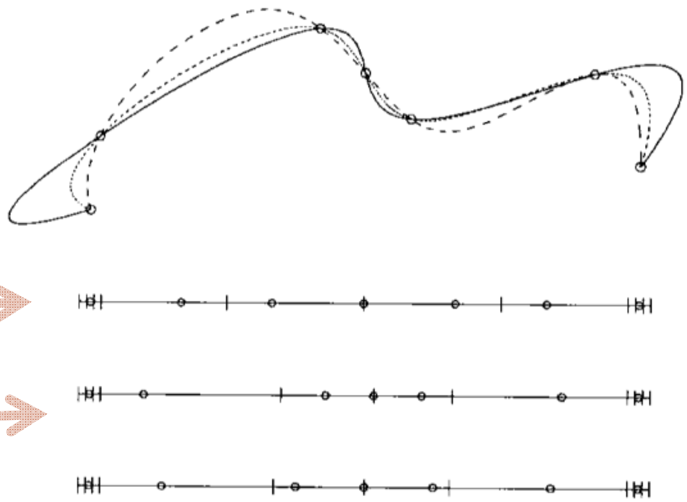
Global Interpolation – Curve (5)

- **Examples (cont.):**

  (3)

  A cubic curve interpolating data with different parameterizations and knots:

  - Uniform parameters + uniform knots (solid curve, top knot vector)
  - Chord length parameters + parameter averaging (dashed, middle knot vector)
  - Centripetal parameters + parameter averaging (dotted, bottom knot vector)



Global Interpolation – Surface (1)

- **Input:** (n+1)*(m+1) points $\{\mathbf{Q}_{k,\ell}\}$, $k = 0, \ldots, n$ and $\ell = 0, \ldots, m$,
- **Output:** an (p,q)-degree nonrational B-spline surface, interpolating these points:

$$\mathbf{Q}_{k,\ell} = \mathbf{S}(\bar{u}_k, \bar{v}_\ell) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(\bar{u}_k) N_{j,q}(\bar{v}_\ell) \mathbf{P}_{i,j}$$

- Again, we need to assign
  - parameter values $(\bar{u}_k, \bar{v}_\ell)$ and knots vector U and V
- Then we **solve** linear equation systems.

Global Interpolation – Surface (2)

- **Parameterization:**
  - Show how to compute $\bar{u}_k$, the $\bar{v}_\ell$ are analogous
  - A common way:

1) curve parameterization method (chord length) on $\bar{u}_0^\ell, \ldots, \bar{u}_n^\ell$ for each ℓ.
2) get $\bar{u}_k$ by averaging:

$$\bar{u}_k = \frac{1}{m+1} \sum_{\ell=0}^{m} \bar{u}_k^\ell \qquad k = 0, \ldots, n$$

- **Computing Knots Vectors:**
  - Simply use the parameter averaging method mentioned previously

$$u_0 = \cdots = u_p = 0 \qquad u_{m-p} = \cdots = u_m = 1$$

$$u_{j+p} = \frac{1}{p} \sum_{i=j}^{j+p-1} \bar{u}_i \qquad j = 1, \ldots, n-p$$

Global Interpolation – Surface (3)

- **Solving Control Points:**

  (1) direct method:

$$\mathbf{Q}_{k,\ell} = \mathbf{S}(\bar{u}_k, \bar{v}_\ell) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(\bar{u}_k) N_{j,q}(\bar{v}_\ell) \mathbf{P}_{i,j}$$

  → (n+1)*(m+1) linear equations in the unknown $P_{i,j}$

  (2) a simpler and more efficient method for tensor product surfaces : sequential curve interpolations

$$\mathbf{Q}_{k,\ell} = \sum_{i=0}^{n} N_{i,p}(\bar{u}_k) \left(\sum_{j=0}^{m} N_{j,q}(\bar{v}_\ell) \mathbf{P}_{i,j} \right) = \sum_{i=0}^{n} N_{i,p}(\bar{u}_k) \mathbf{R}_{i,\ell}$$

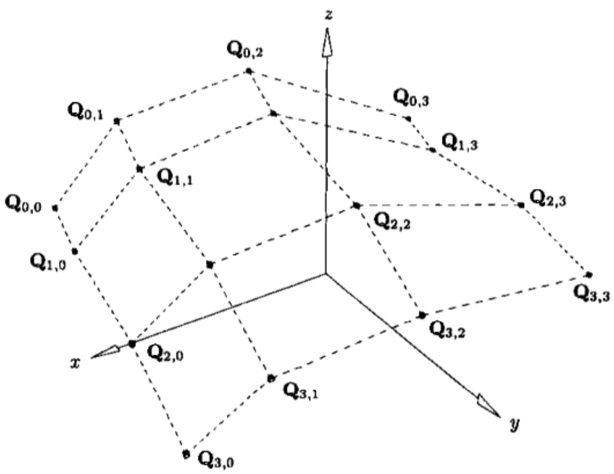
where

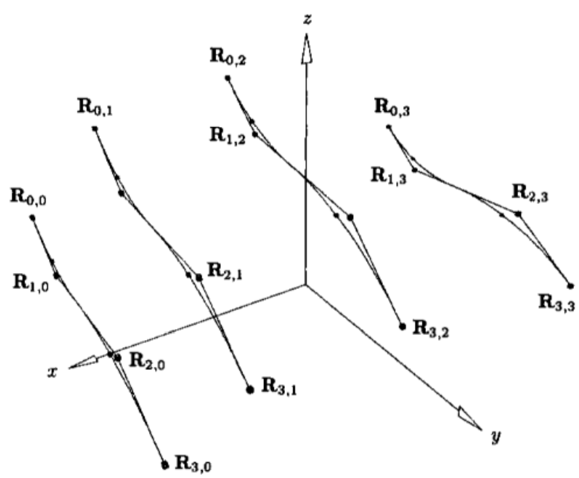
$$\mathbf{R}_{i,\ell} = \sum_{j=0}^{m} N_{j,q}(\bar{v}_\ell) \mathbf{P}_{i,j}$$

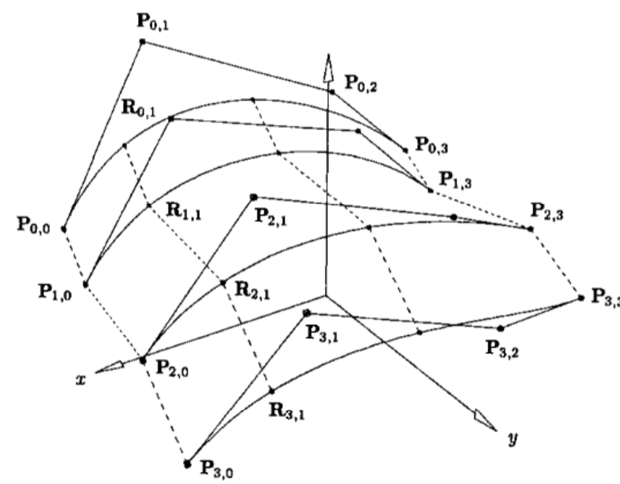
Global Interpolation – Surface (4)

- **Algorithm for (2) on the last page:**

1. using U and the $\bar{u}_k$, do $m+1$ curve interpolations through $\mathbf{Q}_{0,\ell}, \ldots, \mathbf{Q}_{n,\ell}$ (for $\ell = 0, \ldots, m$); this yields the $\mathbf{R}_{i,\ell}$
2. using V and the $\bar{v}_\ell$, do $n+1$ curve interpolations through $\mathbf{R}_{i,0}, \ldots, \mathbf{R}_{i,m}$ (for $i = 0, \ldots, n$); this yields the $\mathbf{P}_{i,j}$







- The algorithm is symmetric.

Homework 2