Curves & Surfaces

CPSC 314

Motivation

Geometric representations so far:

- Discrete geometry
  - *Triangles, line segments*
  - *Rendering pipeline, ray-tracing*
- Specific objects
  - *Spheres*
  - *Ray-tracing*

Want more general representations:

- Flexible like triangles
- But smooth!

Curves&Surfaces as Parametric Functions

Curves&surfaces in arbitrary dimensions

- Curves: $\mathbf{x} = F(t); F : \mathbb{R} \mapsto \mathbb{R}^d$
- Surfaces: $\mathbf{x} = F(s,t); F : \mathbb{R}^2 \mapsto \mathbb{R}^d$

In practice:

- Restrict to specific class of functions
  - *e.g. polynomials of certain degree*

$$\mathbf{x} = \sum_{i=0}^{m} \mathbf{b}_i t^i$$

In 2D: $$\begin{pmatrix} x \\ y \end{pmatrix} = \sum_{i=0}^{m} \begin{pmatrix} b_{x,i} \\ b_{y,i} \end{pmatrix} t^i$$

Polynomial Curves

Advantages:

- Computationally easy to handle
  - $\mathbf{b}_0 \ldots \mathbf{b}_m$ *uniquely describe curve (finite storage, easy to represent)*

Disadvantages:

- Not all shapes representable
  - *Partially fix with piecewise functions (splines)*
- Still not very intuitive
  - *Fix: represent polynomials in different basis*
  - *For example: Bernstein polynomials*
  - *This is what is called a Bézier curve*

Polynomial Bases

Reminder

- The set of all polynomials of degree ≤ m over R forms a vector space with the common polynomial operations
  - *What are those operations?*
  - *Dimension of this space is m+1*
- One common basis for this space are the monomials

$$\{1, t, t^2, \ldots, t^m\}$$

- Problem: the relationship between this basis and a geometric shape is quite unintuitive
- Thus: use another basis later!

Interpolation

Find a polynomial y(t) such that $y(t_i)=y_i$

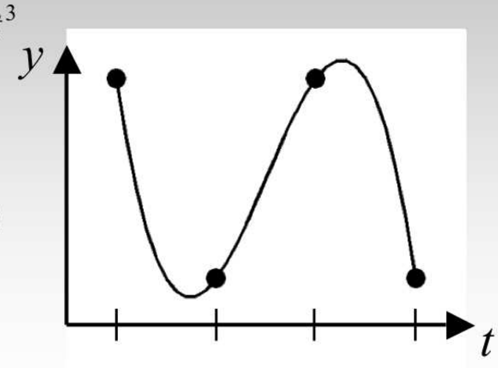
- *For 4 points t_i: need cubic polynomial*

$$y(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3$$

$$\begin{pmatrix} 1 & t & t^2 & t^3 \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = y(t)$$

basis

coefficients



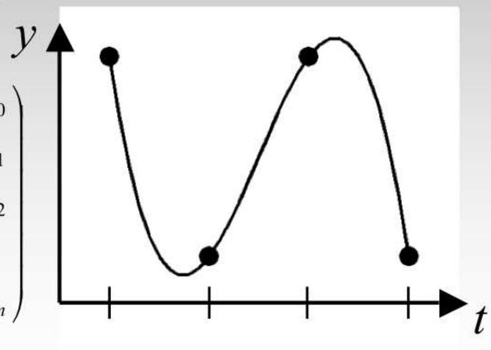
Interpolation

***Find a polynomial* y(t) *such that* y(t$_i$)=y$_i$**

$$y(t) = c_0 + c_1 t + c_2 t^2 + c_3 t^3$$

$$\begin{pmatrix} 1 & t_0 & t_0^2 & \dots & t_0^n \\ 1 & t_1 & t_1^2 & \dots & t_1^n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & t_n & t_n^2 & \dots & t_n^n \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

Vandermonde matrix

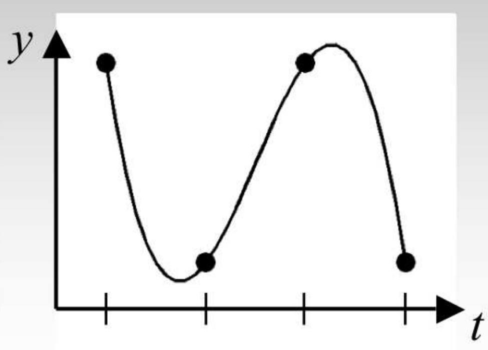


Interpolation

***Find a polynomial* y(t) *such that* y(t$_i$)=y$_i$**

Example:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \\ 1 & 3 & 9 & 27 \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \\ 3 \\ 1 \end{pmatrix}$$

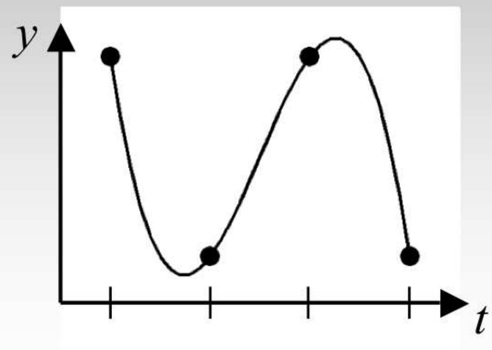


Interpolation

Find a polynomial y(t) such that $y(t_i)=y_i$

Example:

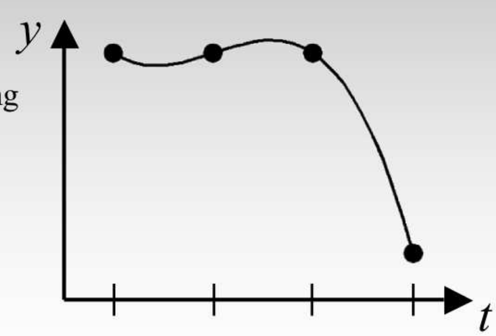
$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 3 \\ -20/3 \\ 6 \\ -1/3 \end{pmatrix}$$



Interpolation

Find a polynomial y(t) such that $y(t_i)=y_i$

Intuitive control of curve using control points!



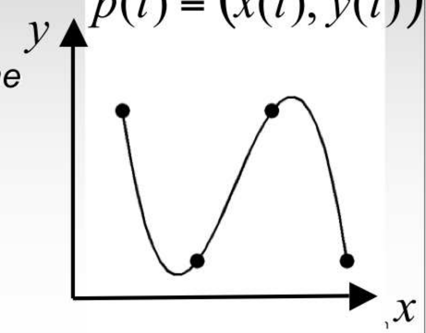
© Wolfgang Heidrich

Interpolation

Parametric setting:

- Perform interpolation separately for x, y, (and z in 3D)
- Assign arbitrary parameter values to control points
  - *I.e. choose t_i, such that $f(t_i)= (x_i,y_i)$*
  - *This choice **will** affect the curve shape!*

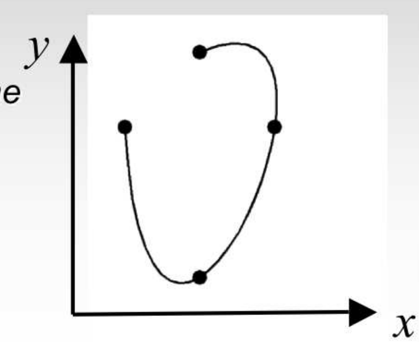
$$p(t) = (x(t), y(t))$$



Interpolation

Parametric setting:

- Perform interpolation separately for x, y, (and z in 3D)
- Assign arbitrary parameter values to control points
  - *I.e. choose t_i, such that $f(t_i)= (x_i,y_i)$*
  - *This choice **will** affect the curve shape!*



Generalized Vandermonde Matrices

Assume different basis functions $f_i(t)$

$$y(t) = \sum_i c_i f_i(t)$$

$$\begin{pmatrix} f_0(t_0) & f_1(t_0) & f_2(t_0) & \dots & f_n(t_0) \\ f_0(t_1) & f_1(t_1) & f_2(t_1) & \dots & f_n(t_1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ f_0(t_n) & f_1(t_n) & f_2(t_n) & \dots & f_n(t_n) \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

Other Bases for Polynomials

Example: Lagrange Polynomials

- Given: m+1 parameter values $t_0 \dots t_m$
- Define

$$L_i^m(t) := \prod_{j=0..m, j\neq i} \frac{t - t_j}{t_i - t_j}; i = 0 \dots m$$

- Clear from definition:
  - *All L_i^m are polynomials of degree m*
  - $L_i^m(t_j) = \begin{cases} 1; i = j \\ 0; else \end{cases}$
  - *In particular, all L_i^m are linearly independent!*

Other Bases for Polynomials

Lagrange Polynomials (cont):

- The L_i^m are linearly independent and there are m+1 of them, therefore they are a basis for the polynomials of degree up to m
- Therefore can write any of polynomial of degree up to m as

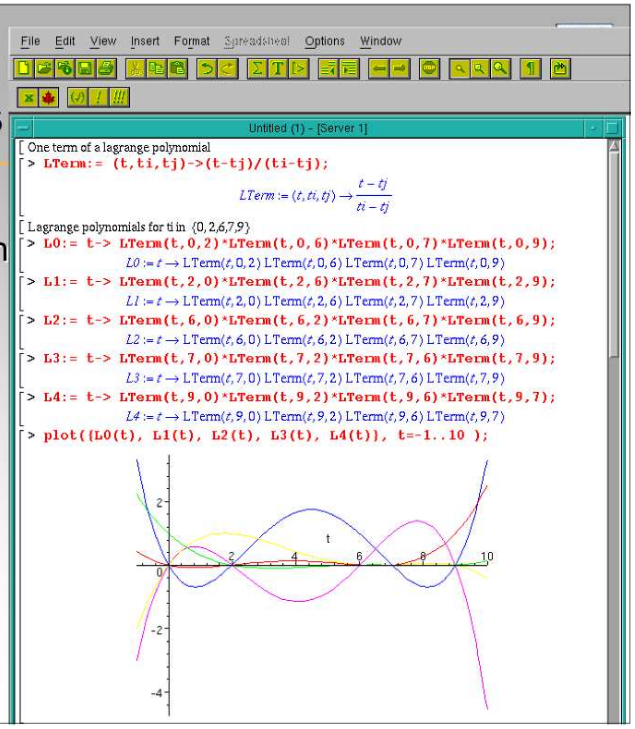
$$F(t) = \sum_{i=0}^{m} L_i^m(t_j) \cdot b_i$$

- In addition, we have for all *i*: $F(t_i) = b_i$
  - *In other words, the polynomial interpolates the points (t_i, b_i)*

Lagrange Polynomials

Example

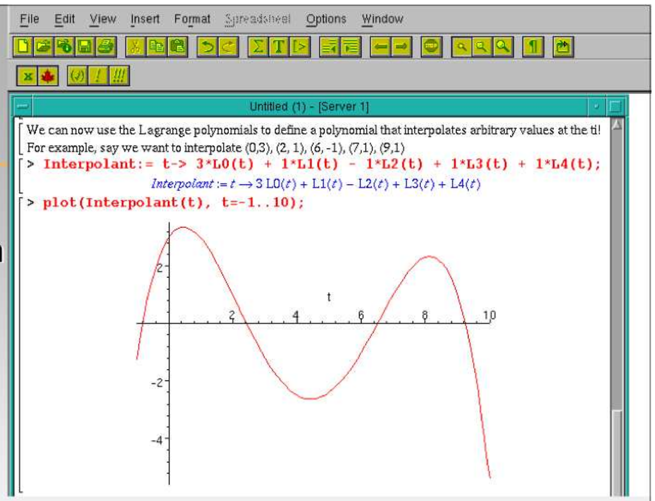
- Basis function



Lagrange Polnomials

Example:

- Interpolation (explicit function)



Note:

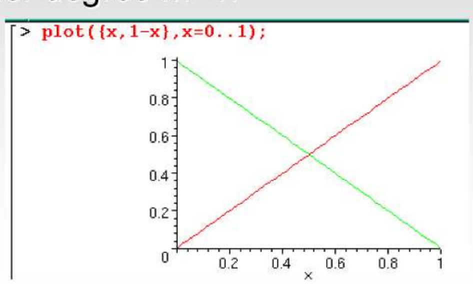
- Same works in parametric setting
- The coefficients then become points to be interpolated!

Other Bases for Polynomials

Bernstein Polynomials

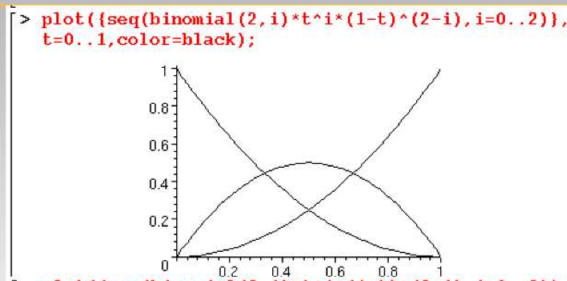
$$B_i^m(t) := \binom{m}{i} t^i (1-t)^{m-i}; i = 0..m; t \in [0,1]$$

- Graph for degree m=1:

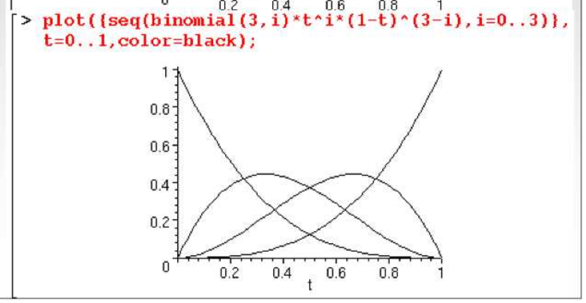


Bernstein Polynomials

- Graph for m=2:



- Graph for m=3:



Bernstein Polynomials

$$B_i^m(t) := \binom{m}{i} t^i (1-t)^{m-i}; i = 0..m; t \in [0,1]$$

Properties:

- $B_i^m(t)$ is a polynomial of degree m
- $B_i^m(t) \geq 0$ for $t \in [0,1]; B_0^m(0) = 1; B_i^m(0) = 0$ for $i \neq 0$
- $B_i^m(t) = B_{m-i}^m(1-t)$
- $B_i^m(t)$ has exactly one maximum in the interval 0..1. It is at t=i/m (proof: compute derivative…)
- W/o proof: all (m+1) functions B_i^m are linearly independent
  - *Thus they form a basis for all polynomials of degree ≤ m*

Bernstein Polynomials

More properties

- $\sum_{i=0}^{m} B_i^m(t) = (t + (1-t))^m \equiv 1$
- $B_i^m(t) = t \cdot B_{i-1}^{m-1}(t) + (1-t) \cdot B_i^{m-1}(t)$
- Both are quite important a fast evaluation algorithm of Bézier curves (de Casteljau algorithm)

Bézier Curves

Definition:

- A Bézier curve is a polynomial curve that uses the Bernstein polynomials as a basis

$$F(t) = \sum_{i=0}^{m} \mathbf{b}_i B_i^m(t)$$

- The b_i are called control points of the Bézier curve
- The control polygon is obtained by connecting the control points with line segments

Advantage of Bézier curves:

- The control points and control polygon have clear geometric meaning and are intuitive to use

Properties of Bézier Curves (Pierre Bézier, Renault, about 1960)

Easy to see:

- The endpoints b_0 and b_m of the control polygon are interpolated and the corresponding parameter values are t=0 and t=1

More properties:

- The Bézier curve is tangential to the control polygon in the endpoints
- The curve completely lies within the convex hull of the control points
- The curve is *affine invariant*
- There is a fast, recursive evaluation algorithm

Bézier Curve Properties

$$F(t) = \sum_{i=0}^{m} \mathbf{b}_i B_i^m(t)$$

Recall:

- Bernstein polynomials have values between 0 and 1 for t∈[0,1], and

$$\sum_{i=0}^{m} B_i^m(t) \equiv 1$$

  - *Therefore: every point on Bézier curve is convex combination of control points*
  - *Therefore: Bézier curve lies completely within convex hull of control points*

De Casteljau Algorithm

Also recall:

- Recursive formula for Bernstein polynomials:

$$B_i^m(t) = t \cdot B_{i-1}^{m-1}(t) + (1-t) \cdot B_i^{m-1}(t)$$

Plug into Bézier curve definition:

$$F(t) = \sum_{i=0}^{m} \mathbf{b}_i \left(t \cdot B_{i-1}^{m-1}(t) + (1-t) \cdot B_i^{m-1}(t) \right)$$

$$= t \cdot \sum_{i=1}^{m} \mathbf{b}_i B_{i-1}^{m-1}(t) + (1-t) \cdot \sum_{i=0}^{m-1} \mathbf{b}_i B_i^{m-1}(t)$$

De Casteljau Algorithm

Consequence:

- Every point $F(t_0)$ on a Bézier curve of degree m is the convex combination of two points $G(t_0)$ and $H(t_0)$ that lie on Bézier curves of degree m-1.
- The control points of $G(t)$ are the first m control points of $F(t)$
- The control points of $H(t)$ are the last m control points of $F(t)$

De Casteljau Algorithm

Recursion:

- Every point on a Bézier curve can be generated through successive convex combinations of the degree 0 Bézier curves
- Degree 0 Bézier curves are the control points!

$$F(t) = \sum_{i=0}^{0} \mathbf{b}_i B_i^0(t) = \mathbf{b}_i \cdot 1 \equiv \mathbf{b}_i$$

De Casteljau Algorithm

After working out the math we get:

$$F(t) = \mathbf{b}_0^m(t) \;\; ; \text{where}$$

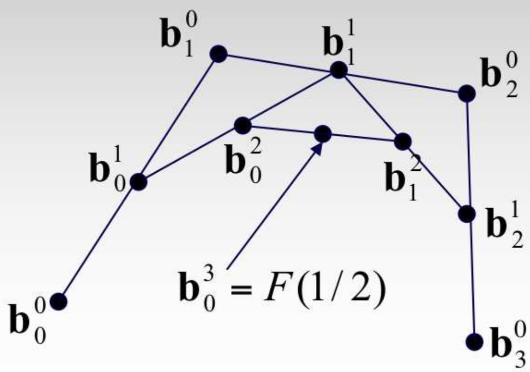
$$\mathbf{b}_i^0(t) := \mathbf{b}_i(t); \quad i = 0 \ldots m$$

$$\mathbf{b}_i^l(t) := (1-t) \cdot \mathbf{b}_i^{l-1}(t) + t \cdot \mathbf{b}_{i+1}^{l-1}(t)$$

De Casteljau Algorithm

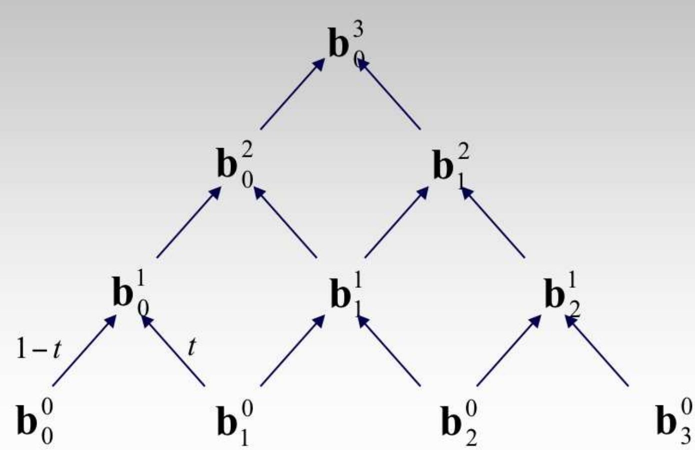
Graphical Interpretation:

- Determine point *F(1/2)* for the cubic Bézier curve given by the following four points:



De Casteljau Algorithm

Evaluation scheme (cubic case):



Tensor Product Surfaces

What about surfaces?

- Use basis functions as in the case of curves
- Apply them independently to the parametric directions s and t
- Works for arbitrary basis

Example:

- Bézier curve: $F(t) = \sum_{i=0}^{m} B_i^m(t) \cdot \mathbf{b}_i$
- Tensor product Bézier patch:

$$F(s,t) = \sum_{i=0}^{m_s} \sum_{j=0}^{m_t} B_i^{m_s}(s) \cdot B_j^{m_t}(t) \cdot \mathbf{b}_{i,j}$$

Tensor Product Surfaces

Notes:

- The surface is polynomial in s and t, depending on basis
  - *The degree in s is m_s*
  - *The degree in t is m_t*
  - *The total degree is $m_s + m_t$*
- The algorithms from the curves transfer directly to tensor product surfaces
- The properties of these surfaces are directly related to the properties of the corresponding curves

Tensor Product Surfaces

Properties:

- Convex hull
- Affine invariance
- The control points of the edge curves are the boundary points of the control mesh
- A Bézier patch interpolates the corner vertices of its control mesh

More on Curves & Surfaces

This was a (very) quick overview

- More details in CPSC 424 (Geometric Modeling)
- Taught by Alla Sheffer in 2008/09

Upcoming Lectures

Tuesday:

- Research topics in graphics
- Tour of graphics labs

Thursday:

- Q&A session for final prep