CS480: Computer Graphics
Curves and Surfaces

Sung-Eui Yoon
(윤성의)

Course URL:
http://jupiter.kaist.ac.kr/~sungeui/CG

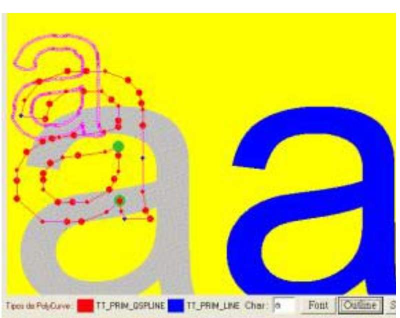
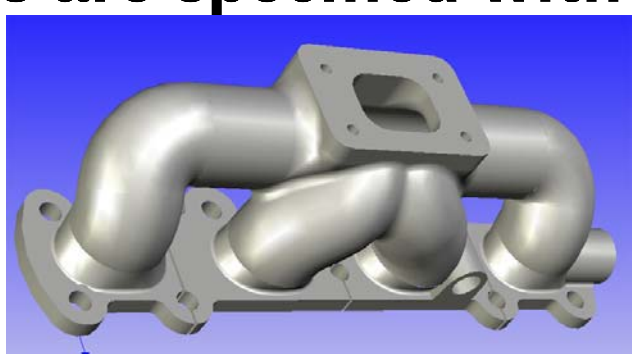
KAIST

Today's Topics

- **Surface representations**
- **Smooth curves**
- **Subdivision**

Smooth Curves and Surfaces

- Triangles
  - Requires many triangles to represent high-resolution geometry, but has limited resolution in the end
- Smooth curves and surfaces are preferred in many applications
  - Art, industrial design, mathematics, architecture, computer-aided design (CAD), etc
  - Even fonts are specified with curves

http://www.flyinmiata.com

http://www.acrobatusers.com

Three Representations of Curves

- **Parametric:**
  - **$C(t) = (x(t), y(t))$, where t is parameter**
  - **E.g., parabola: (t, t^2)**
- **Non-parametric explicit**
  - **$y = f(x)$**
  - **Its parametric form $C(t) = (t, f(t))$**
- **Implicit:**
  - **$F(x, y) = 0$**

KAIST

Rendering Explicit Functions

- **Explicit functions are easy to render**
  - **Loop over the independent variables generating vertices and normals**

$$v_{ij} = \begin{bmatrix} x_i \\ y_j \\ f(x_i, y_j) \\ 1 \end{bmatrix} \qquad n_{ij} = \begin{bmatrix} -\frac{\partial f(x_i, y_j)}{\partial x} \\ -\frac{\partial f(x_i, y_j)}{\partial y} \\ 1 \\ 0 \end{bmatrix}$$

- **However, the class of surfaces they describe is too limited**

Shortcomings of Explicit Functions

- **Consider the following representations of a plane as the following:** $z = Ax + By + C$
  - **For any values of A, B, and C, the resulting surface will be a plane**
  - **However, not every plane can be specified in this form (e.g., the x-z or y-z planes)**
- **Similarly, we cannot completely describe a sphere centered at the origin as a simple function:**

$$z = \sqrt{r^2 - x^2 - y^2}$$

KAIST

Implicit Representations

- **Many surfaces can be described as implicit functions, in which all variables are independent and are the "zero-set" of a 3-D function**

$$0 = f(x, y, z)$$

- **This representation treats all dimensions equivalently**
  - **As a result, it can describe a wider class of surfaces**
  - **For instance, all planes can be described using an implicit function of the form:** $Ax + By + Cz + D = 0$
  - **Likewise, we can describe spheres centered at the origin implicitly:**

$$x^2 + y^2 + z^2 - r^2 = 0$$

Algebraic Surfaces

- Subclasses of implicit surfaces
  - Particularly, those for which f(x,y,z) is polynomial in the three independent variables
  - It is interesting, because it forms a vector space
  - As a result, we can define operations like addition, and multiplication by a scalar for them

Quadrics

- **The algebraic surfaces of degree 2, have the following form:**

$$Ax^2 + By^2 + Cz^2 + Dxy + Exz + Fyz + Gx + Hy + Iz + J = 0$$

  - **These surfaces are called the "quadrics"**
  - **Include spheres, ellipsoids, paraboliods, disks, and cones**

- **Implicit functions are more powerful than explicit functions**
  - **There is no simple procedural way to generate points on them**

Parametric Functions

- **Define a general "parameter space" and provide separate explicit functions for each variable as a function of these parameters**

$$x = f_x(u, v)$$
$$y = f_y(u, v)$$
$$z = f_z(u, v)$$

- **Parametric functions are mappings from a simple parameter space to the surface**
  - **A common example of a parametric mapping is the sphere:**

$$x = r\cos(\theta)\cos(\phi) \quad y = r\sin(\theta)\cos(\phi) \quad z = r\sin(\phi)$$

Rendering Parametric Functions

- **Parametric functions are easy to render**
  - **Step through the parameter space computing the vertices and normals:**

$$v_{ij} = \begin{bmatrix} f_x(u_i, v_j) \\ f_y(u_i, v_j) \\ f_z(u_i, v_j) \\ 1 \end{bmatrix} \qquad n_{ij} = \begin{bmatrix} \frac{\partial f_x(u_i, v_j)}{\partial u} \\ \frac{\partial f_y(u_i, v_j)}{\partial u} \\ \frac{\partial f_z(u_i, v_j)}{\partial u} \\ 0 \end{bmatrix} \times \begin{bmatrix} \frac{\partial f_x(u_i, v_j)}{\partial v} \\ \frac{\partial f_y(u_i, v_j)}{\partial v} \\ \frac{\partial f_z(u_i, v_j)}{\partial v} \\ 0 \end{bmatrix}$$

- **There is also a special class of "polynomial parametric functions" of the form:**

$$f(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} c_{ij} u^i v^j$$

  - **Where the degree of the function is m+n, and it has 3(n+1)(m+1) coefficients**

KAIST

Surface Design

- We now have a framework for specifying a wide range of surfaces
  - In the case of polynomial function, we need only provide a set of coefficients → Very non-intuitive

- In general, we would prefer to specify a surface more directly
  - For instance we might want to specify points on the surface, or provide other various controls

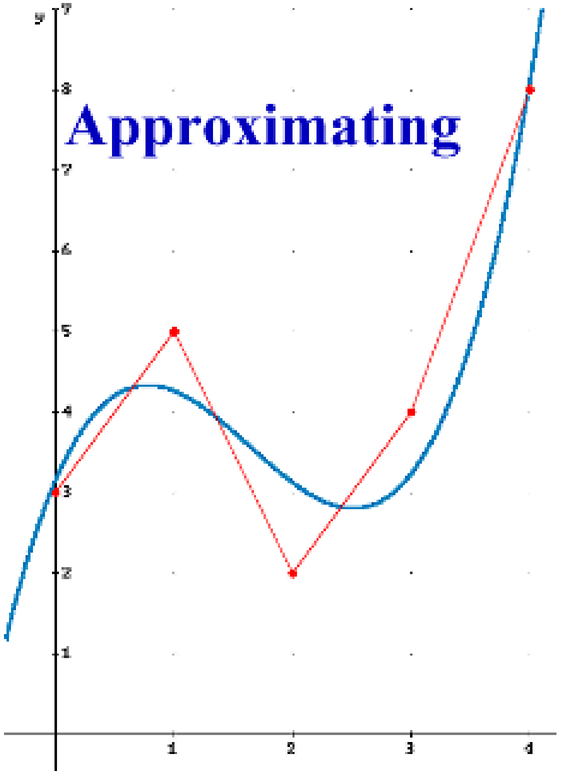
- To simplify our discussion, we will first consider curves in the plane

Specifying Curves

Control points - a set of points that influence the curve's shape

Interpolating spline - curve passes through all control points

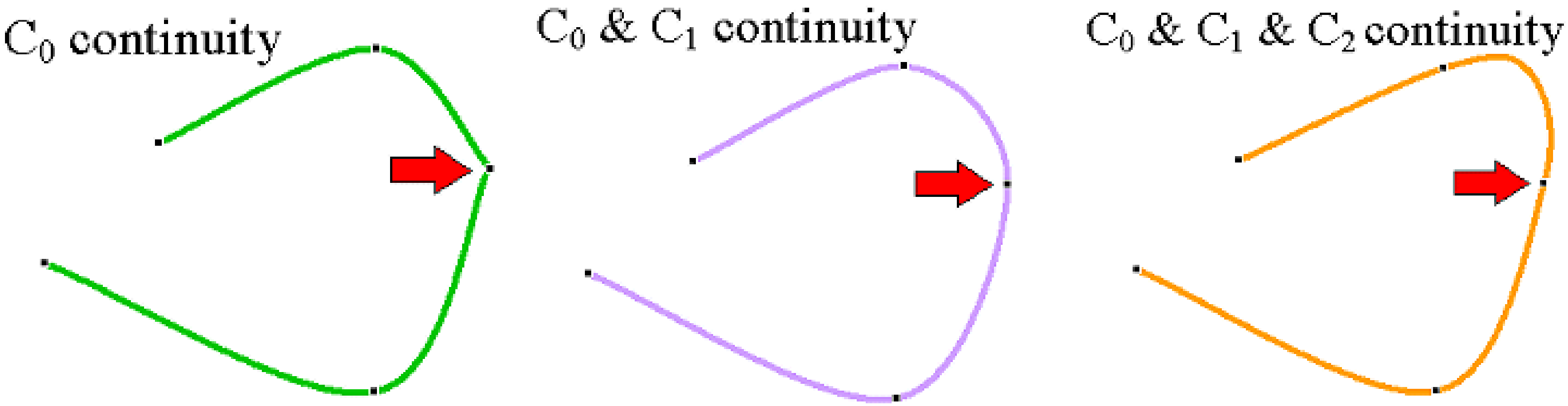
Approximating spline - control points merely influence shape



KAIST

Piecewise Curve Segments

- **Often we will want to represent a curve as a series of curves pieced together**
  - **But we will want these curves to fit together reasonably**
- **Parametric continuity:**



  - **A curve has C^k, or parametric, continuity in the interval $t \in [a,b]$, if all derivatives, up through the k^{th}, exist and are continuous at all points within the interval**

Parametric Cubic Curves

- **Suppose that we want to assure C2 continuity our functions**
  - **Then, the functions must be of at least degree 3**
  - **Here's what a parametric cubic spline function looks like:**

$$x = a_x t^3 + b_x t^2 + c_x t + d_x$$

$$y = a_y t^3 + b_y t^2 + c_y t + d_y$$

- **Alternatively, it can be written in matrix form:**

$$\begin{bmatrix} x & y \end{bmatrix} = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix}$$

Solving for Coefficients

The whole story of polynomial splines is deriving their coefficients

How?

By satisfying constraints given control points and continuity conditions

An Illustrative Example

- Cubic Hermite splines
  - Specified by 2 control points and 2 tangent vectors at the curve's endpoints

Hermite Specification

The Gradient of a Cubic Spline

- **Expressions for the tangent vectors**
  - **Computed by taking derivatives of the parametric function**
  - **These derivatives are also functions of unknown coefficients**

$$\begin{bmatrix} \frac{dx}{dt} & \frac{dy}{dt} \end{bmatrix} = \begin{bmatrix} 3t^2 & 2t & 1 & 0 \end{bmatrix} \begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix}$$

Hermite Specification

- **Here is the full specification of the Hermite constraints given in the form of a matrix equation:**

[t^3,t^2,t, 1] evaluated at **t = 0**

[t^3, t^2,t, 1] evaluated at **t = 1**

$$\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \frac{dx_1}{dt} & \frac{dy_1}{dt} \\ \frac{dx_2}{dt} & \frac{dy_2}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix}$$

[3t^2,2t,1, 0] evaluated at **t = 0**

[3t^2,2t,1, 0] evaluated at **t = 1**

Solve for the Hermite Coefficients

- **Finding the coefficients it is a simple matter of algebra**

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \frac{dx_1}{dt} & \frac{dy_1}{dt} \\ \frac{dx_2}{dt} & \frac{dy_2}{dt} \end{bmatrix} = \begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix}$$

Spline Basis and Geometry Matrices

- **In this form, we give special names to each term of our spline specification:**

$$\underbrace{\begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{M}_{Hermite}} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \frac{dx_1}{dt} & \frac{dy_1}{dt} \\ \frac{dx_2}{dt} & \frac{dy_2}{dt} \end{bmatrix}}_{\mathbf{G}_{Hermite}} = \begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix}$$

KAIST

Cubic Hermite Spline Equation

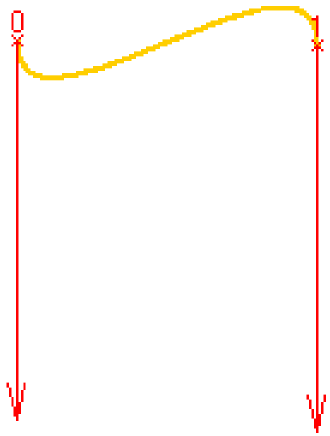
- **Now we have a full specification of our curve:**

$$[x \quad y] = [t^3 \quad t^2 \quad t \quad 1] \underbrace{\begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{M}_{Hermite}} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \frac{dx_1}{dt} & \frac{dy_1}{dt} \\ \frac{dx_2}{dt} & \frac{dy_2}{dt} \end{bmatrix}}_{\mathbf{G}_{Hermite}}$$

Hermite Spline Demonstration

Discussion:

- Is a tangent vector *really* an intuitive control?
- Piecewise issues:
  - C_0 easy
  - C_1 reasonable



KAIST

Another Way to Think About Splines

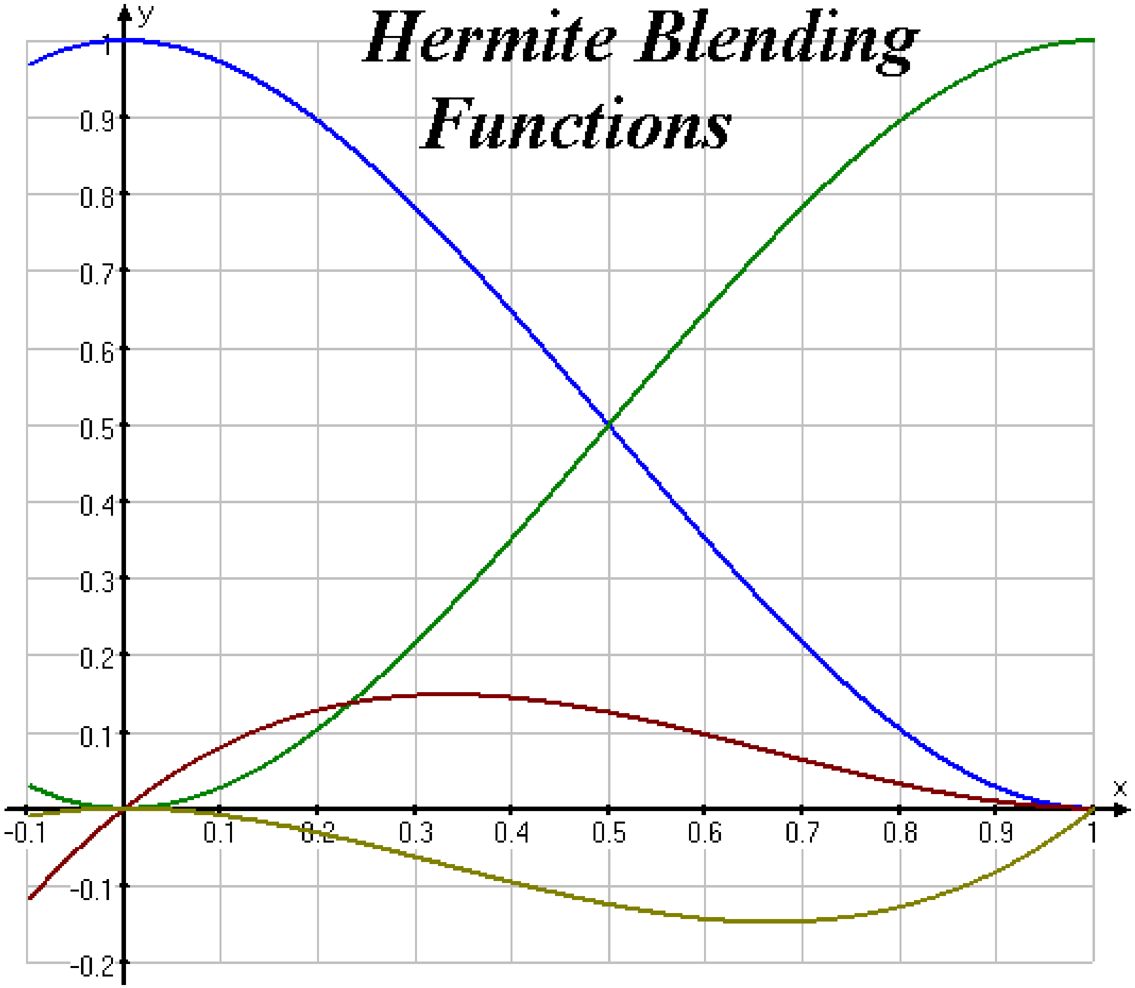
- **The contribution of each geometric factor can be considered separately**
  - **This approach gives a so-called *blending function* associated with each factor**

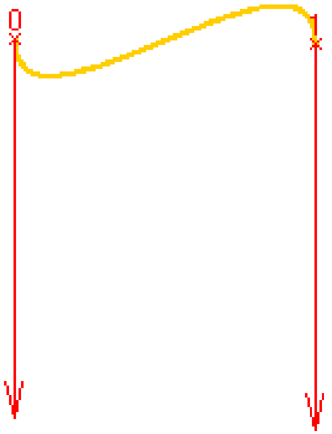
- **Reordering multiplications gives:**

$$[x \quad y] = [t^3 \quad t^2 \quad t \quad 1] \underbrace{\begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{M}_{Hermite}} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \frac{dx_1}{dt} & \frac{dy_1}{dt} \\ \frac{dx_2}{dt} & \frac{dy_2}{dt} \end{bmatrix}}_{\mathbf{G}_{Hermite}} \longrightarrow p(t) = \begin{bmatrix} 2t^3 - 3t^2 + 1 \\ -2t^3 + 3t^2 \\ t^3 - 2t^2 + t \\ t^3 - t^2 \end{bmatrix}^{\mathbf{T}} \begin{bmatrix} p_1 \\ p_2 \\ \nabla p_1 \\ \nabla p_2 \end{bmatrix}$$

KAIST

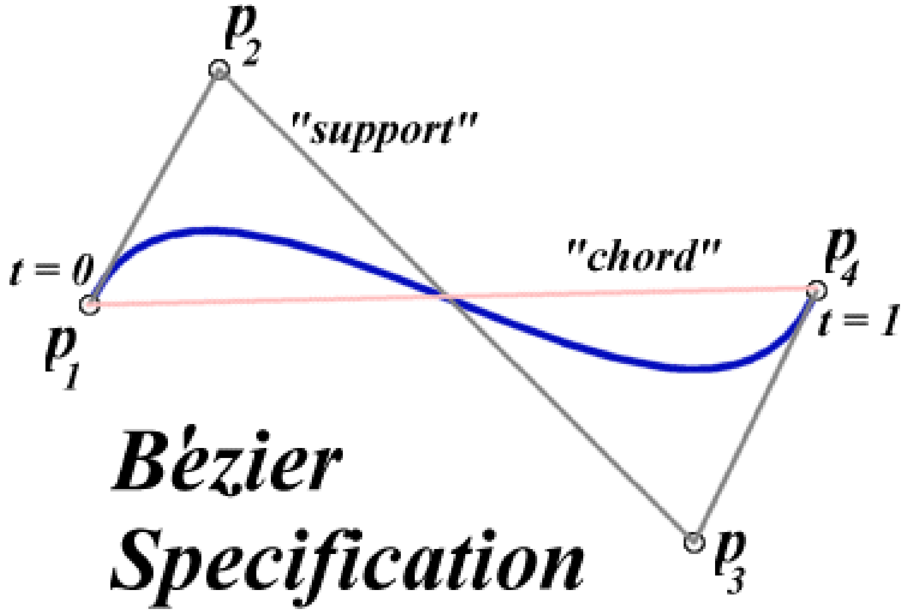
Hermite Blending Functions





Bezier Curves

- Cubic Hermite splines present some user friendliness problems
- Next we will define a new spline class that has more intuitive controls



Coefficients for Cubic Bezier Splines

- **The gradients at the control points of a Bezier Spline**
  - **Expressed in terms of the adjacent control points:**

$$\nabla p_1 = 3(p_2 - p_1)$$
$$\nabla p_4 = 3(p_4 - p_3)$$

- **Using such a specification is reasonable, but what makes 3 a magic number?**

Here's the Trick!

- **Knowing this we can formulate a Bezier spline in terms of the Hermite geometry spec**

$$\underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \frac{dx_1}{dt} & \frac{dy_1}{dt} \\ \frac{dx_2}{dt} & \frac{dy_2}{dt} \end{bmatrix}}_{\mathbf{G}_{Hermite}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{bmatrix} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix}}_{\mathbf{G}_{Bezier}}$$

- **And substituting gives:**

$$\begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix} = \underbrace{\begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{M}_{Hermite}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -3 & 3 & 0 & 0 \\ 0 & 0 & -3 & 3 \end{bmatrix} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix}}_{\mathbf{G}_{Bezier}}$$

Basis and Geometry Matrices for Bezier Splines

- **Now we can compute our spline coefficients given a Bezier Specification**

$$\begin{bmatrix} a_x & a_y \\ b_x & b_y \\ c_x & c_y \\ d_x & d_y \end{bmatrix} = \underbrace{\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{M}_{Bezier}} \underbrace{\begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix}}_{\mathbf{G}_{Bezier}}$$

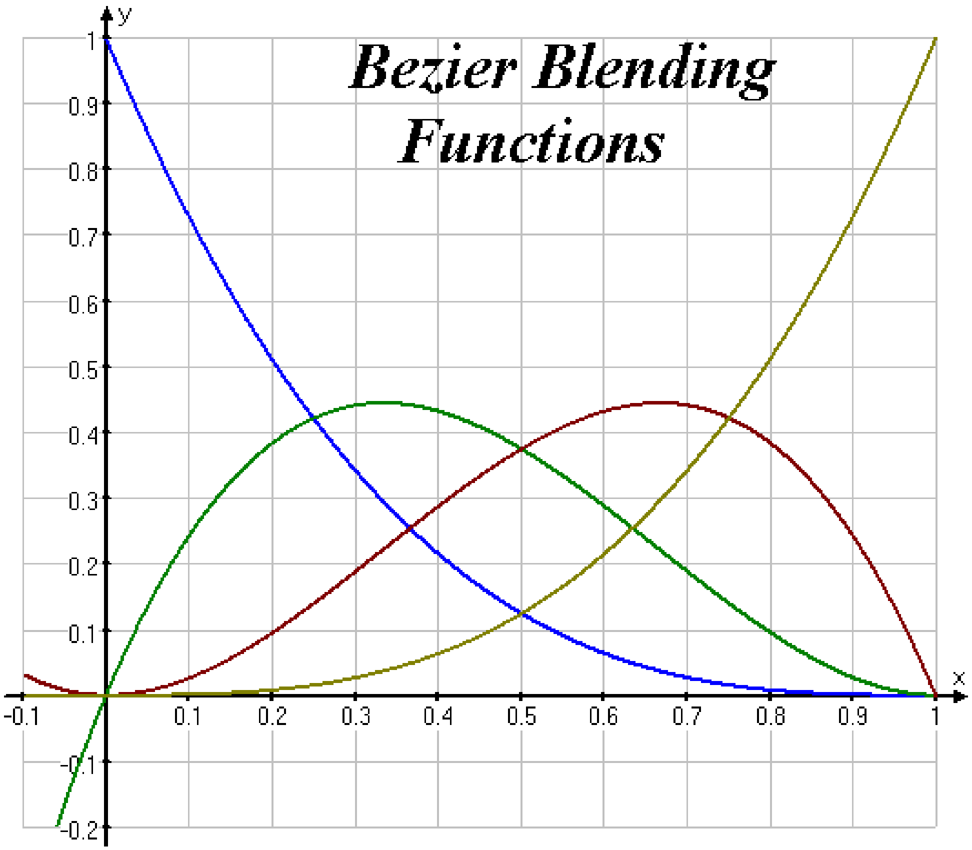
Bezier Blending Functions

- **The justification for Bezier spline basis can only be approached by considering its blending functions:**

$$p(t) = \begin{bmatrix} (1-t)^3 \\ 3t(1-t)^2 \\ 3t^2(1-t) \\ t^3 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}$$

- **This family of polynomials (called order-3 Bernstein polynomials) have the following unique properties:**
  - **They are all positive in the interval [0, 1]**
  - **Their sum is equal to 1 (Where have we seen this before?)**

Plots of Bezier Blending Functions

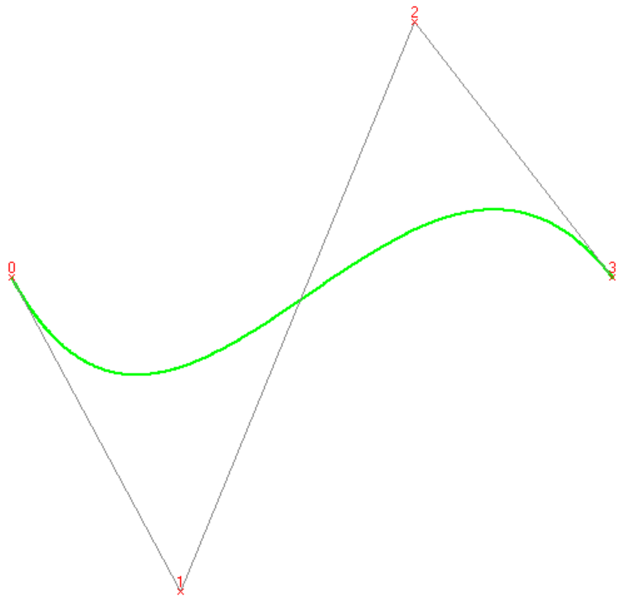


- **Every point on the curve is an Affine combination of the control points**
  - **Since the sum of these blending weights is 1**
- **The weights of this combination are all positive**
  - **Thus, the curve is also a *Convex combination* of the control points!**

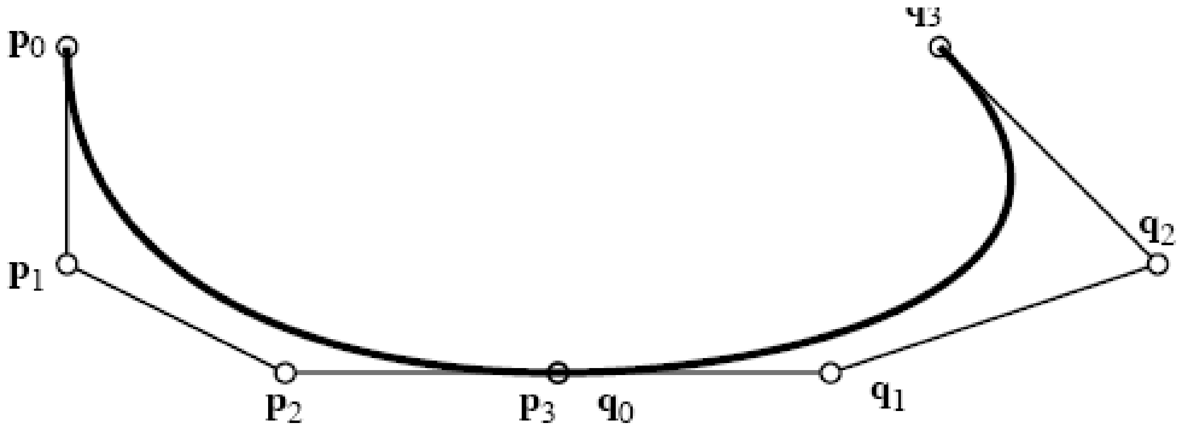
Bezier Demonstration

Discussion:

- **Strange mix of points on and off the curve**



- **Piecewise issues:**
  - **C_0 easy**
  - **C_1 easy**

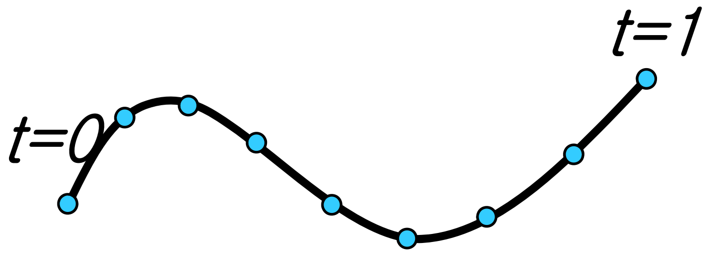


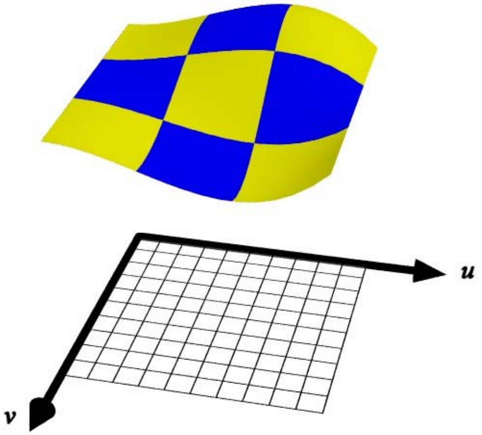
Spline Rendering: Take 1

Step 1: Given a spline specification, compute the coefficients by multiplying the spline's basis matrix by the geometry vector

Step 2: Take uniform steps in the parameter space (t = 0, 0.1, 0.2, …, 1.0), and generate new points on the curve

Step 3: Connect these points with line segments

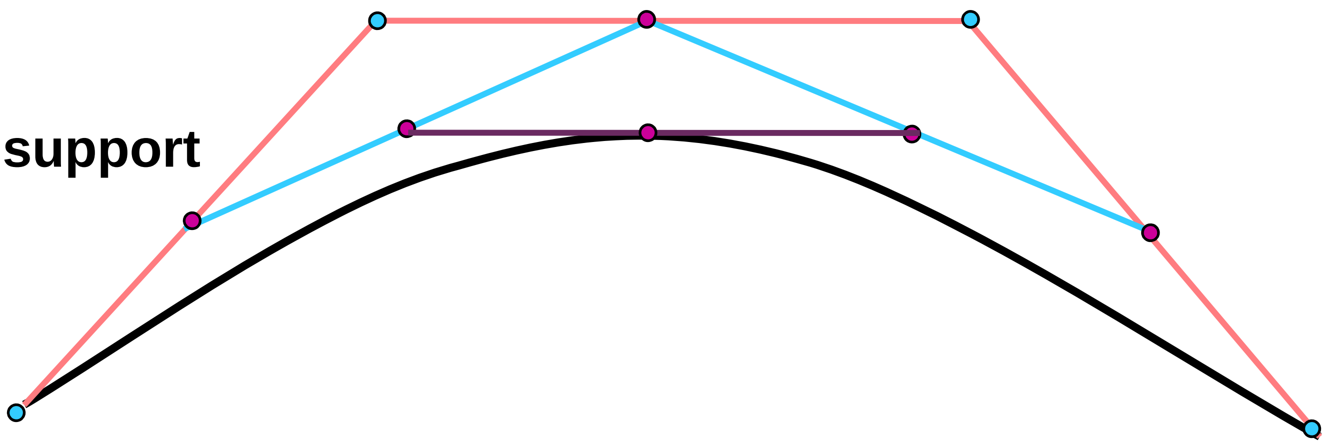




Spline Rendering: Take 2

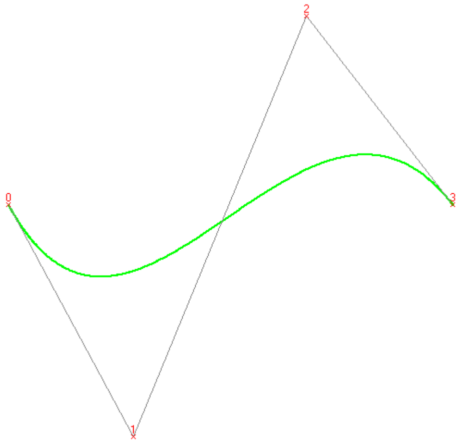
- "de Casteljau" Algorithm
  - Recursively generate new control points for arbitrary fractions of the domain from the initial control points

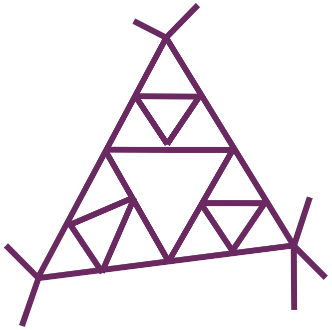
1. Find midpoints of support
2. Connect with new segments
3. Find midpoints of new segments
4. Connect with new segment
5. Find its midpoint



Subdivision

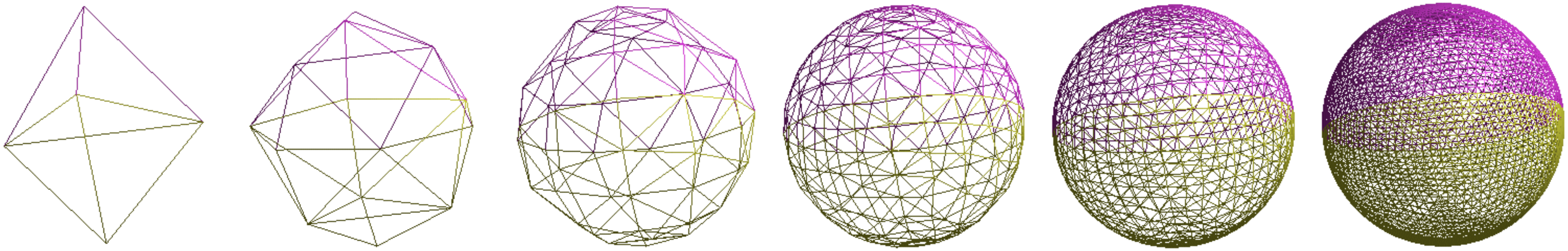
- **This process can be repeated recursively**
  - **The resulting scaffolding is a good approximation of the actual surface**
- **Why use subdivision (recursion) instead of uniform domain sampling (iteration)?**
  - **Stopping conditions can be based on local shape properties (curvature)**
  - **Subdivision can be generalized to non-square domains, in particular to triangular**
  - **(Link for more examples)**





Example of Generalized Subdivision

Here is a sample of generalized subdivision:



0–levels *1–level* *2–levels* *3–levels* *4–levels* *5–levels*

Geri, Pixar animation

Bezier Surfaces

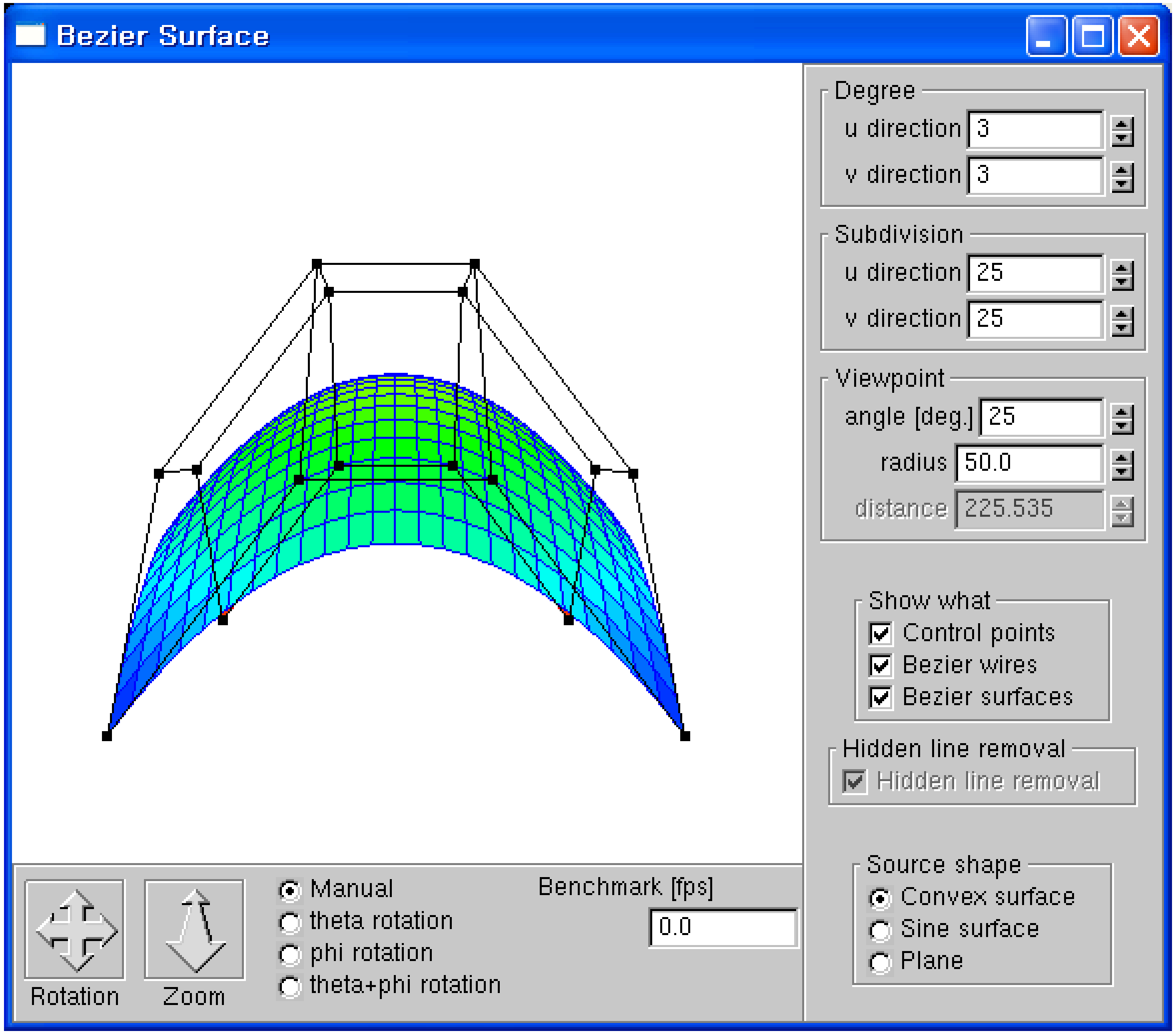
- **Introduce two parameters, s and t**
  - **Let $B_{i,n}(s)$ and $B_{j,m}(t)$ be the Bernstein basis functions of degrees n and m in s and t**
- **Then, a Bezier surfaces with control points $p_{i,j}$ is defined as the follow:**

$$S(s,t) = \sum_{i=0}^{n}\sum_{j=0}^{m} p_{i,j} B_{i,n}(s) B_{j,m}(t) \text{ for } (s,t) \in [0,1]\times[0,1]$$

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

  - **Requires 4x4 control points for degrees 3 and 3 in s and t**

Demonstration of Bezier Surfaces



http://www.mizuno.org/gl/bs/