Realtime Computer Graphics on GPUs

Curves and Surfaces

Jan Kolomazník

Department of Software and Computer Science Education
Faculty of Mathematics and Physics
Charles University in Prague

Overview

Curve and surface definition

Explicit : $y = sin(x),\ z = x^2 + y^2$

Implicit : $0 = x^2 + y,\ 0 = x^2 + y^2 + z^2$

Parametric :

- $\mathbf{Q}(t) = \begin{bmatrix} cos(t) \\ sin(t) \end{bmatrix}$
- $\mathbf{P}(u,v) = \begin{bmatrix} cos(u)(R + r * cos(v)) \\ sin(u)(R + r * cos(v)) \\ r * sin(v) \end{bmatrix}$

PARAMETRIC CURVES AND SURFACES

- Parameter(s) have specified range ($[0, 1]$)
- Same curve (surface)can have multiple parametrizations
  - Arc length parametrization - animation, uniform sampling, ...
  - In general case cannot be expressed analyticaly
- Sampling the parameter space $\rightarrow$ discrete points in space
  - Approximating by polyline or polygonal mesh

Cubic Curves

Cubic Curves – Why?

- Simple to work with
- Physical motivation (natural spline)
- Humans do not need higher order derivatives to perceive smoothness

Cubic Curves

Basic parametric cubic curve equation:

$$\mathbf{Q}(t) = \mathbf{a} + \mathbf{b}t + \mathbf{c}t^2 + \mathbf{d}t^3$$

CUBIC CURVES

Basic parametric cubic curve equation:

$$
\begin{aligned}
Q_x(t) &= a_x + b_x t + c_x t^2 + d_x t^3 \\
Q_y(t) &= a_y + b_y t + c_y t^2 + d_y t^3 \\
Q_z(t) &= a_z + b_z t + c_z t^2 + d_z t^3
\end{aligned}
$$

Cubic Curves

Rewrite in a matrix form:

$$\mathbf{Q}(t) = \begin{bmatrix} a_x & b_x & c_x & d_x \\ a_y & b_y & c_y & d_y \\ a_z & b_z & c_z & d_z \end{bmatrix} \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

CUBIC CURVES

Rewrite in a matrix form:

$$\mathbf{Q}(t) = \begin{bmatrix} a_x & b_x & c_x & d_x \\ a_y & b_y & c_y & d_y \\ a_z & b_z & c_z & d_z \end{bmatrix} \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

Compact matrix form:

$$\mathbf{Q}(t) = \mathbf{C}\mathbf{T}(t)$$

C is coefficient matrix (constant)

CUBIC CURVES

Simple derivative:

$$\mathbf{Q}'(t) = \mathbf{C}\frac{d}{dt}\mathbf{T}(t) = \mathbf{C}\begin{bmatrix} 0 \\ 1 \\ 2t \\ 3t^2 \end{bmatrix}$$

GEOMETRICAL CONSTRAINTS

- Certain curves can be defined as weighted sum of four geometrical constraints.
- B_k – blending functions

$$
\begin{aligned}
\mathbf{Q}(t) &= \sum_{k=0}^{3} B_k(t)\mathbf{g_k} \\
&= (a_1 + b_1 t + c_1 t^2 + d_1 t^3)\mathbf{g_1} \\
&+ (a_2 + b_2 t + c_2 t^2 + d_2 t^3)\mathbf{g_2} \\
&+ (a_3 + b_3 t + c_3 t^2 + d_3 t^3)\mathbf{g_3} \\
&+ (a_4 + b_4 t + c_4 t^2 + d_4 t^3)\mathbf{g_4}
\end{aligned}
$$

GEOMETRICAL CONSTRAINTS

Rewrite in a matrix form:

$$\mathbf{Q}(t) = \begin{bmatrix} \mathbf{g_1} & \mathbf{g_2} & \mathbf{g_3} & \mathbf{g_4} \end{bmatrix} \begin{bmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \\ a_4 & b_4 & c_4 & d_4 \end{bmatrix} \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

GEOMETRICAL CONSTRAINTS

Rewrite in a matrix form:

$$\mathbf{Q}(t) = \begin{bmatrix} \mathbf{g_1} & \mathbf{g_2} & \mathbf{g_3} & \mathbf{g_4} \end{bmatrix} \begin{bmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \\ a_4 & b_4 & c_4 & d_4 \end{bmatrix} \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix}$$

Compact matrix form:

$$\mathbf{Q}(t) = \mathbf{GMT}(t)$$

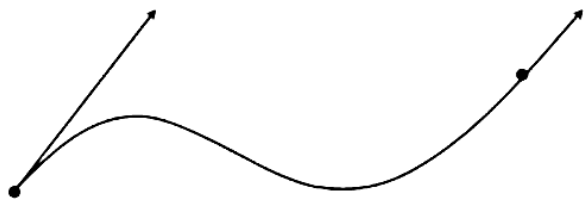
$\mathbf{G}$ geometry matrix, $\mathbf{M}$ basis matrix

Hermite Curves

- Endpoints $\mathbf{P_1}, \mathbf{P_2}$
- Tangents at those points $\mathbf{T_1}, \mathbf{T_2}$

$$\mathbf{G} = \begin{bmatrix} \mathbf{P_1} & \mathbf{P_2} & \mathbf{T_1} & \mathbf{T_2} \end{bmatrix}$$

$$\mathbf{M_H} = \begin{bmatrix} 1 & 0 & -3 & 2 \\ 0 & 0 & 3 & -2 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

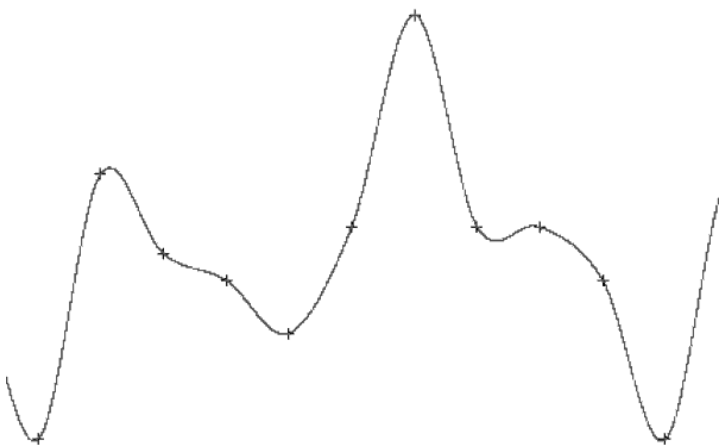


CATMULL-ROM SPLINES

- Control points $\mathbf{P_1}, \ldots \mathbf{P_n}$
- Tangent at point: $\mathbf{T_i} = \frac{1}{2}(\mathbf{P_{i+1}} - \mathbf{P_{i-1}}$
- Catmull-Rom $\rightarrow$ Hermite

$$\mathbf{G_H} = \begin{bmatrix} \mathbf{P_{i-1}} & \mathbf{P_i} & \mathbf{P_{i+1}} & \mathbf{P_{i+2}} \end{bmatrix}$$

$$\mathbf{M_{CR}} = \frac{1}{2}\begin{bmatrix} 0 & -1 & 2 & -1 \\ 2 & 0 & -5 & 3 \\ 0 & 1 & 4 & -3 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$



Bézier Curves

- P. de Casteljau (Citroen) – numerically stable algorithm using linear interpolations
- P. Bézier (Renault) – Bernstein polynomials as blending functions

$$B_{n,k}(t) = \binom{n}{k} t^k (1-t)^{n-k}$$

- Four control points per segment

$$\mathbf{B}(t) = \sum_{k=0}^{3} B_{3,k}(t)\mathbf{P}_k$$

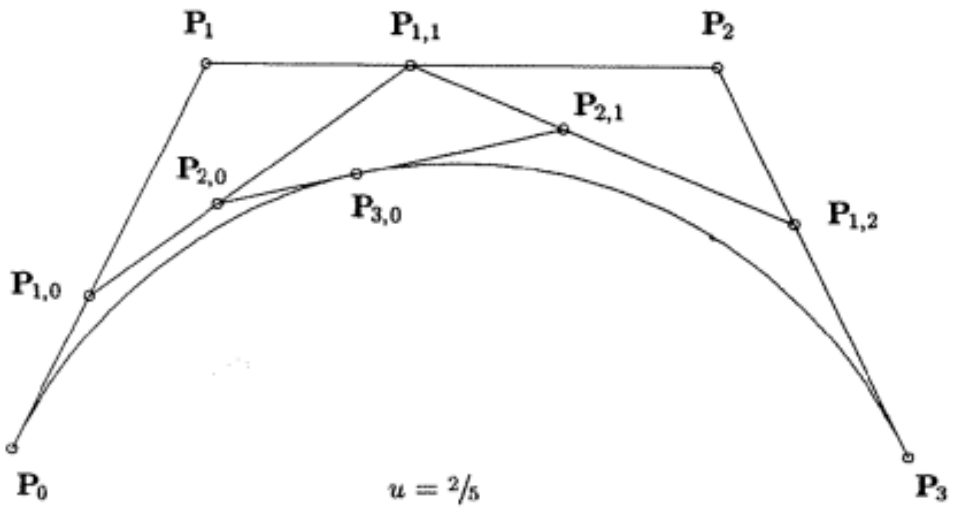
Basis matrix

$$\mathbf{G_B} = \begin{bmatrix} \mathbf{P_1} & \mathbf{P_2} & \mathbf{P_3} & \mathbf{P_4} \end{bmatrix}$$

$$\mathbf{M_B} = \begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

DE CASTELJAU ALGORITHM

- Repeated linear interpolation
- Each *lerp* uses same weight



Surfaces

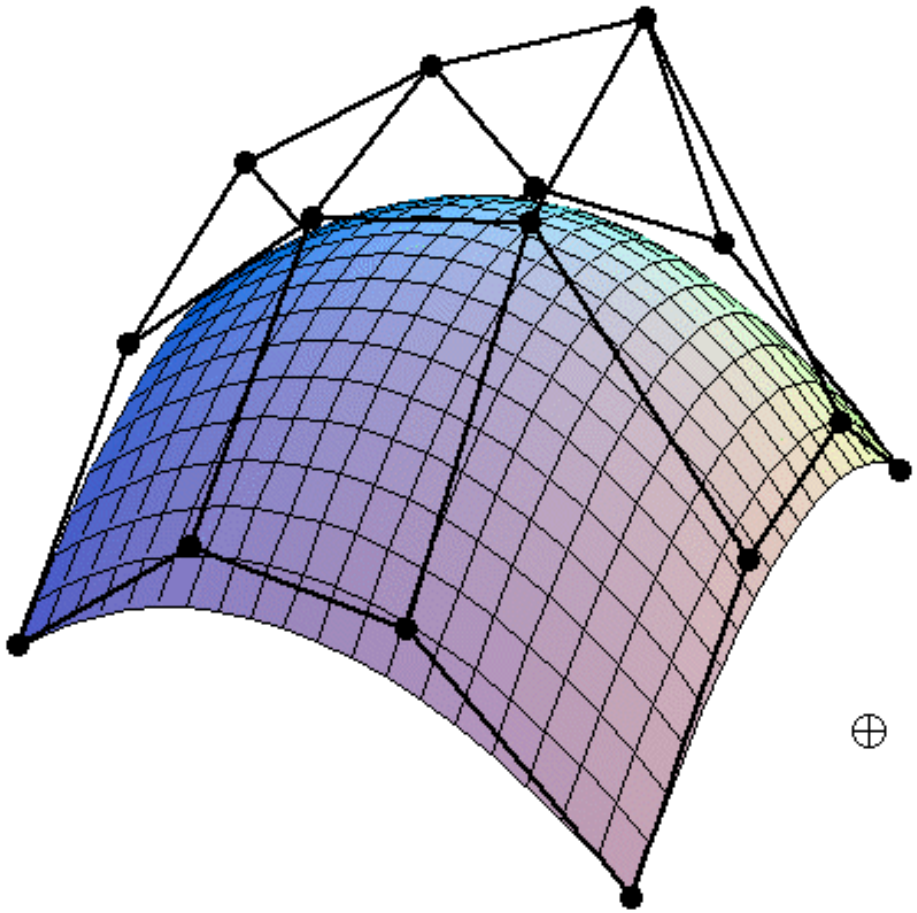
Bicubic Surfaces

- Extension of bilinear interpolation concept
- Tensor product surfaces
- 16 geometrical constraints - $\mathbf{G} - 4 \times 4 \times 3$ coefficients

$$\mathbf{Q}(u,v) = \sum_{k=0}^{3} \sum_{l=0}^{3} B_k(u) B_l(v) \mathbf{g}_{kl}$$

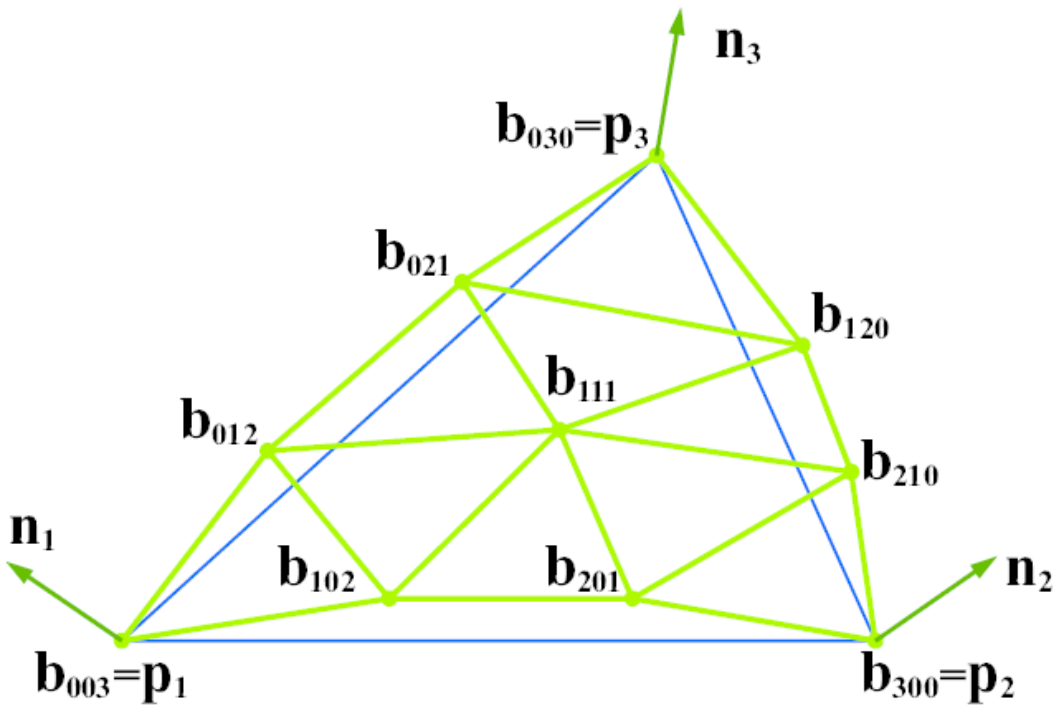
$$Q^r(u,v) = \mathbf{S}^T(v) \mathbf{M}^t \mathbf{G}^r \mathbf{M} \mathbf{T}(u)$$

Bézier Patch



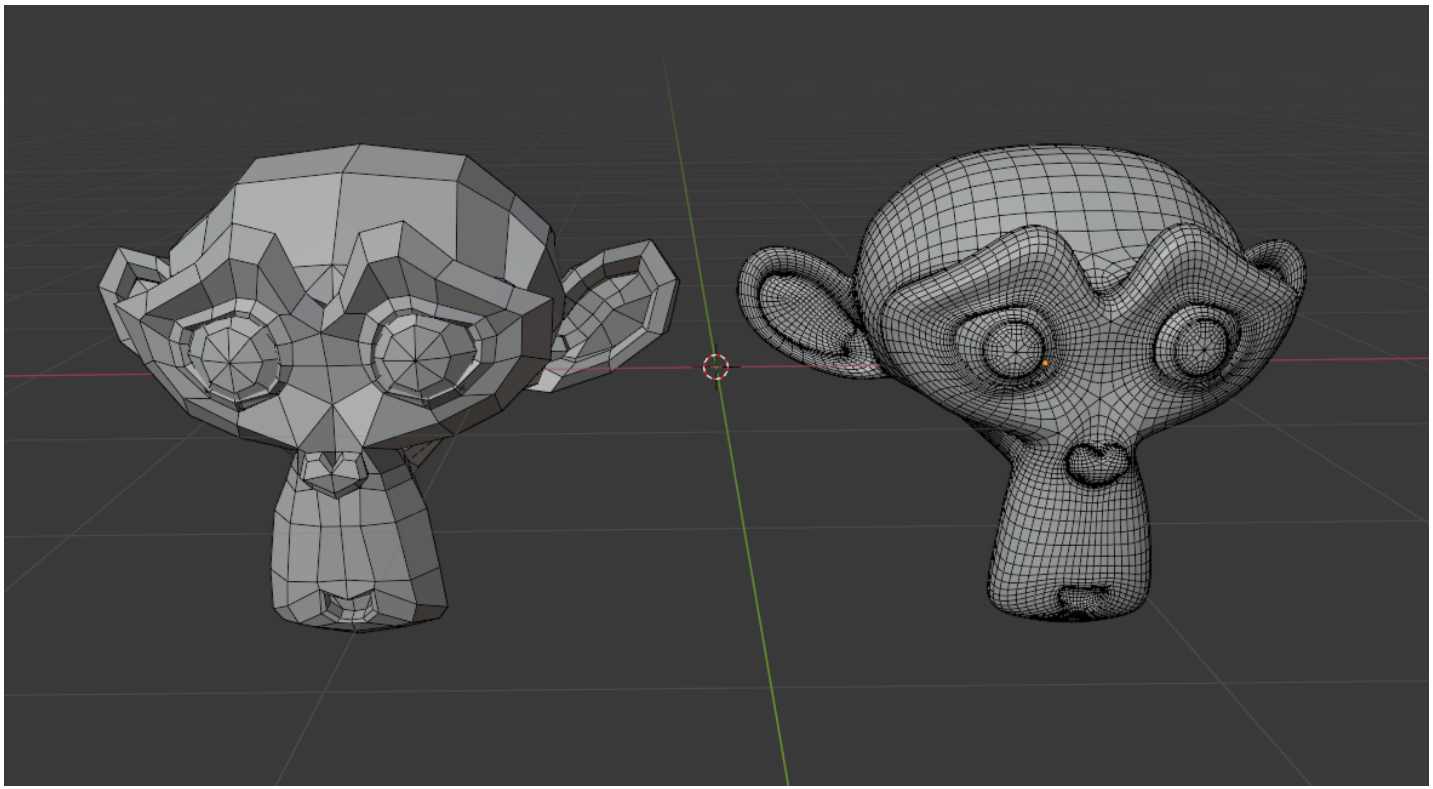
Bézier Triangle

- de Casteljau extension for triangles - barycentric coordinates instead of lerp
- 10 control points
  - Simplification for triangle meshes with vertex normals – PN-triangles



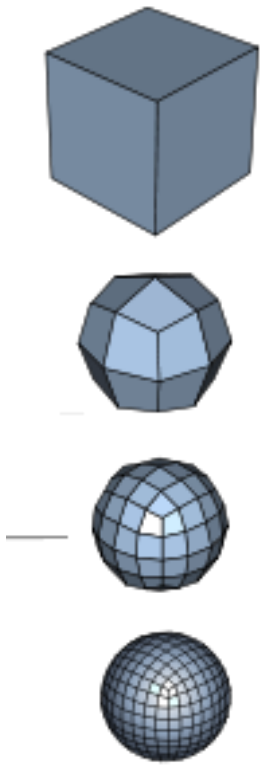
Subdivision Surfaces

- Recursive algorithms
- Two steps:
  - Refine mesh topology
  - Adjust vertex positions
- Interpolating vs approximating



CATMULL-CLARK SCHEME

- Produces quad mesh - keeps clean topology
- Vertices inserted into edges and face centers



Loop Scheme

- Defined for triangle meshes
- Spliting edges
- New position - weighted average of vertices from incident triangles