

# Surfaces

# Polynomial Curves

---

- Bezier, Hermite, ...
- Polynomial of degree  $n$
- Example (cubic Bezier curve)

$$\mathbf{q}_b^T(u) = \begin{pmatrix} u^3 & u^2 & u^1 & 1 \end{pmatrix} \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} (\mathbf{p}_1)_1 & (\mathbf{p}_1)_2 & (\mathbf{p}_1)_3 \\ (\mathbf{p}_2)_1 & (\mathbf{p}_2)_2 & (\mathbf{p}_2)_3 \\ (\mathbf{p}_3)_1 & (\mathbf{p}_3)_2 & (\mathbf{p}_3)_3 \\ (\mathbf{p}_4)_1 & (\mathbf{p}_4)_2 & (\mathbf{p}_4)_3 \end{pmatrix}$$

- Or equivalently (see Buss, VII.1)  $q_j(u) = \sum_i B_i(u) (\mathbf{p}_i)_j$

# Splines: Piecewise Curves

---

- Catmull-Rom, B-spline, ...
- Interpolate or approximate control points by assembling polynomial curves.
- Example (Catmull-Rom spline):
  - Piecewise cubic
  - Interpolates input points  $p_1, p_2, \dots, p_n$ .
  - Unlike Bezier, there can be more than four points
  - $C^1$  (tangent line) continuity
  - More details (Buss, VII.15.1)

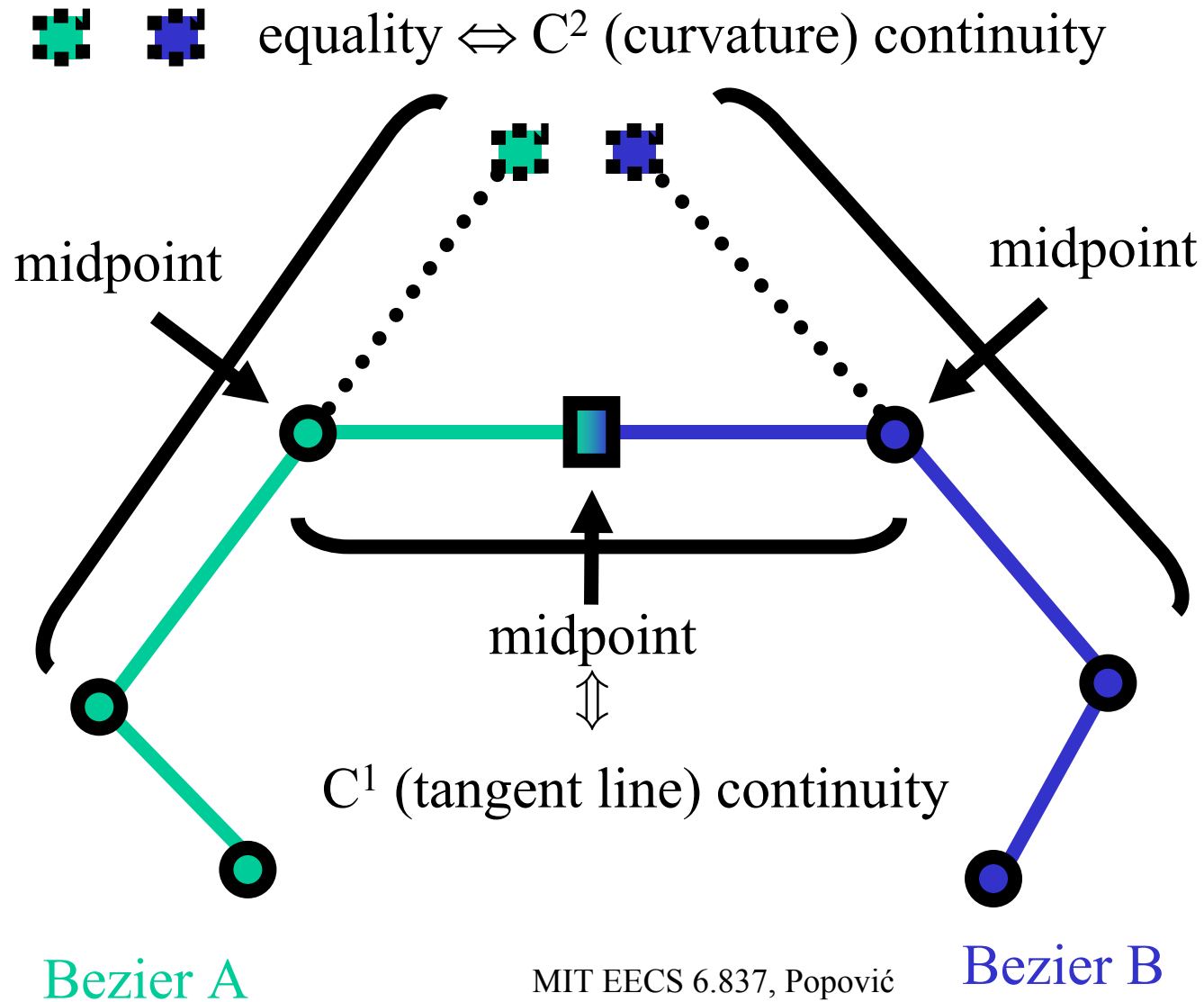
# Example: Uniform B-spline

---

- Piecewise cubic
- **Approximates** input points.
- Unlike Bezier, there can be more than four points
- Unlike Catmull-Rom, it does not interpolate
- $C^2$  (curvature) continuity
  
- Algebraic construction (Buss, VIII.1)

# B-spline: Geometric Construction

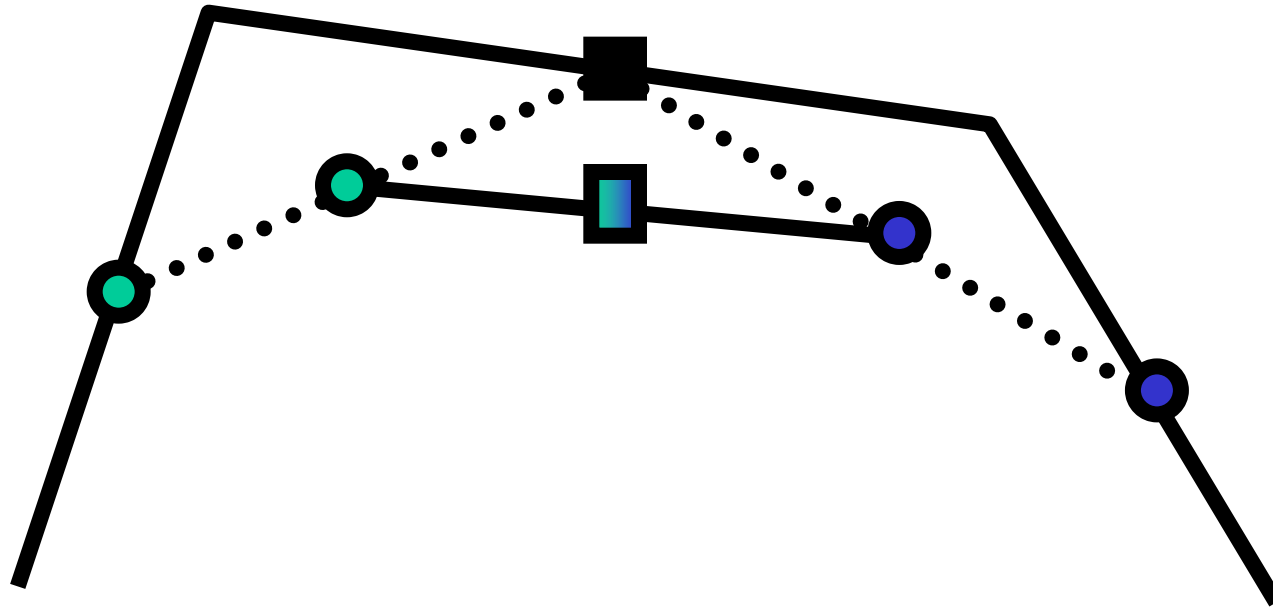
---



# B-spline: Proof

---

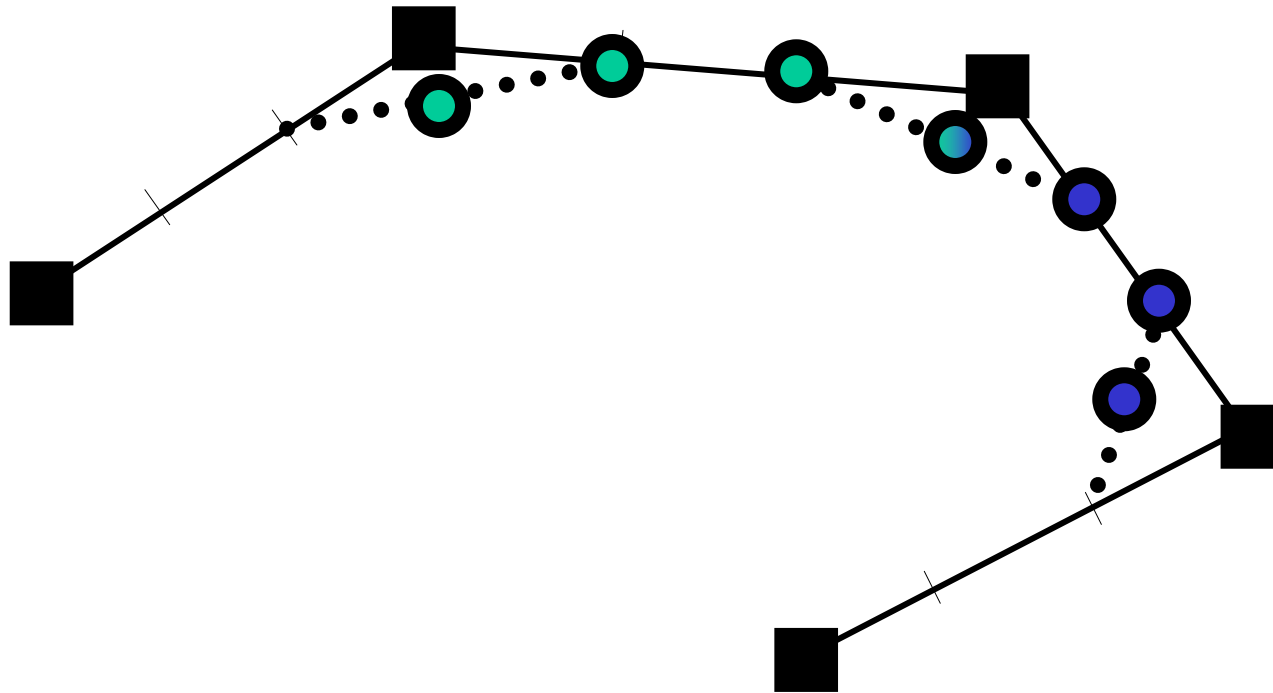
- Proof of Bezier  $C^2$  continuity via Subdivision/de Casteljau construction



# B-spline

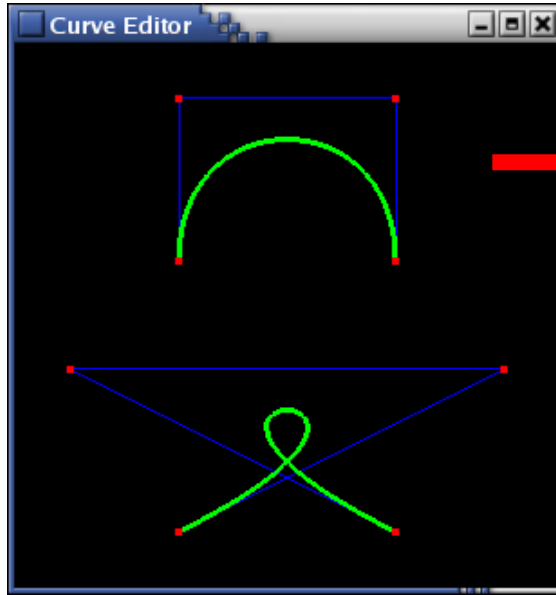
---

- B-spline control points are black squares
- Control points for two Bezier curves are circles

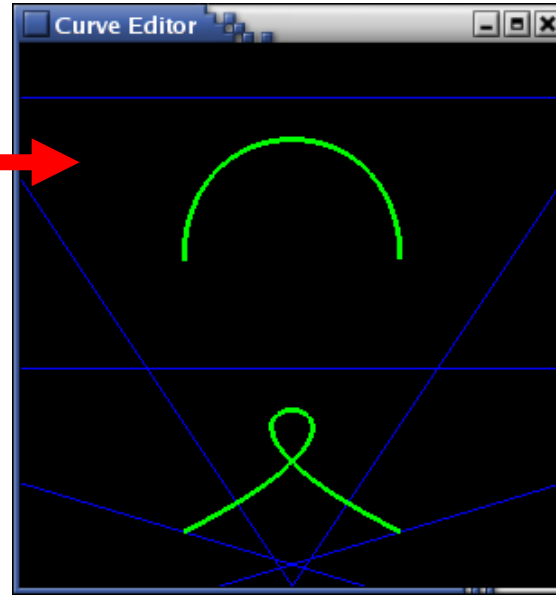


# Converting between Bézier & BSpline

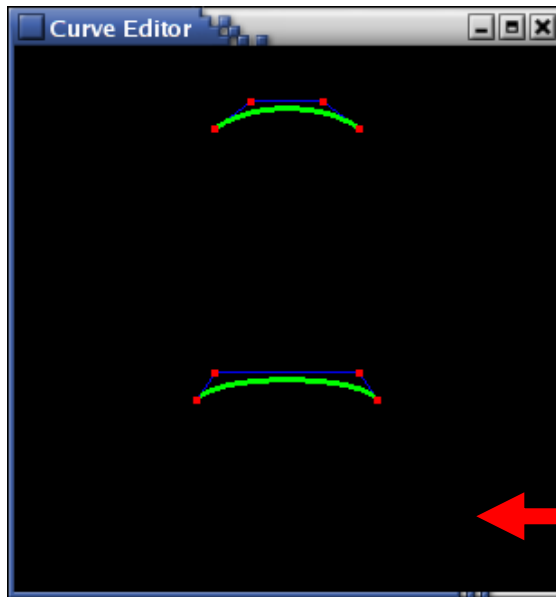
original  
control  
points as  
Bézier



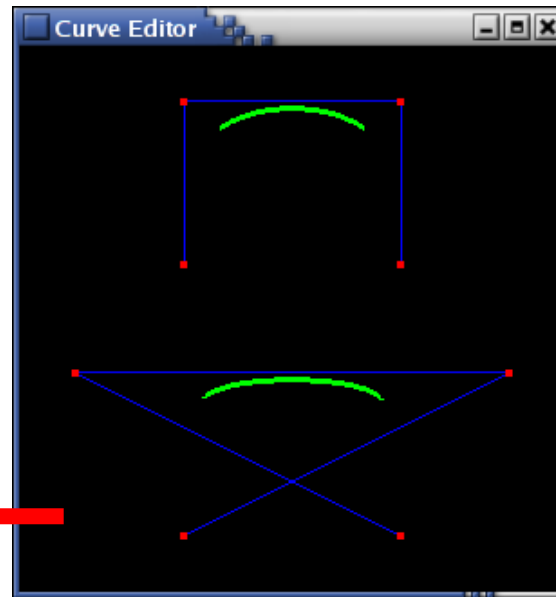
new  
BSpline  
control  
points to  
match  
Bézier



new  
Bézier  
control  
points to  
match  
BSpline



original  
control  
points as  
BSpline



# Conversion: Analytic Form

---

- From B-spline control points to Bezier control points.
- Slide  $\mathbf{G}_s$  as a window of four B-spline points to obtain control points for each Bezier curve.

$$\mathbf{G}_b = \frac{1}{6} \begin{pmatrix} 1 & 4 & 1 & 0 \\ 0 & 4 & 2 & 0 \\ 0 & 2 & 4 & 0 \\ 0 & 1 & 4 & 1 \end{pmatrix} \mathbf{G}_s$$

# Surfaces

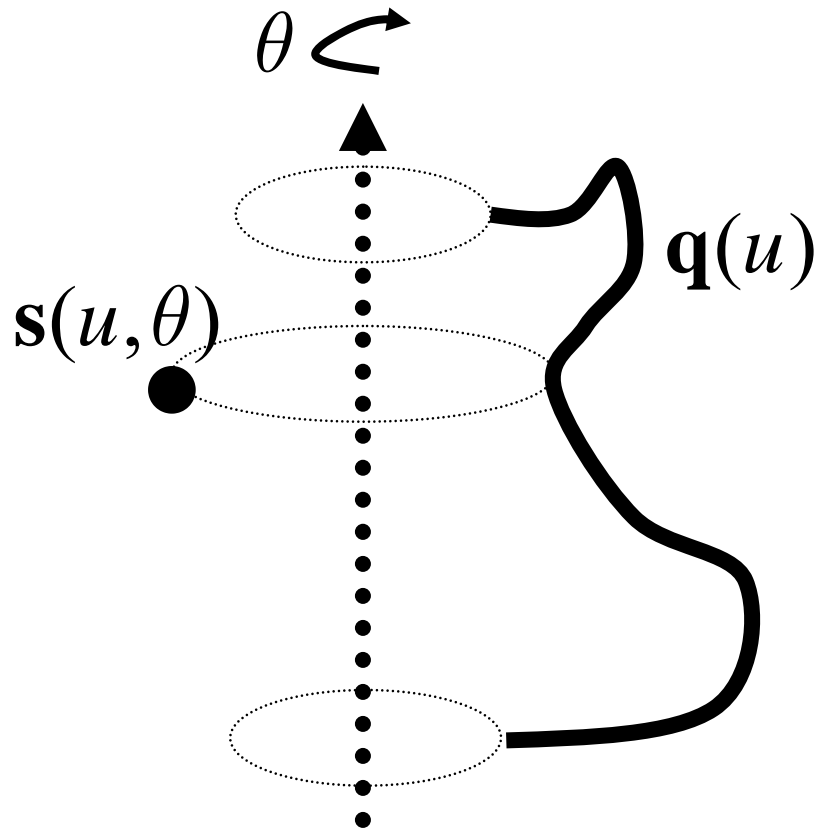
---

- Swept Surfaces
  - Surfaces of revolution
  - General surfaces
- Tensor-Product Surfaces

# Surfaces of Revolution

---

- Rotate a 2D profile curve around an axis.



$$\mathbf{s}(u, \theta) = \mathbf{R}(\theta)\mathbf{q}(u)$$

# Surface Normal Vectors

---

- Normal vectors are needed for shading
  - Normal vector perpendicular to tangent plane
  - Tangent plane spanned by partial derivatives

– So:

$$\vec{\mathbf{n}}(u, v) = \text{normalize} \left( \frac{\partial \mathbf{s}(u, v)}{\partial u} \times \frac{\partial \mathbf{s}(u, v)}{\partial v} \right)$$

– In the special case of surface of revolution:

$$\vec{\mathbf{n}}(u, \theta) = \mathbf{R}(\theta) \text{ normal}(\mathbf{q}(u))$$

# General Sweep Surfaces

---

- Trace out surface by moving a profile curve along a trajectory.
- Orientation options:
  - Align profile curve with an axis.
  - Align profile curve with a Frenet frame.
  - Analytic form uses a matrix and a trajectory to transform the profile curve:

$$\mathbf{s}(u, v) = \mathbf{M}(\mathbf{c}(u))\mathbf{q}(u)$$

# Bezier Tensor Products

---

- Use a 4x4 grid of control  $\mathbf{p}_{ij}$  points to build a surface:
  - Use the four rows as control points for four Bezier curves:  $\mathbf{q}_0(u)$ ,  $\mathbf{q}_1(u)$ ,  $\mathbf{q}_2(u)$ ,  $\mathbf{q}_3(u)$
  - Define a point on the surface  $s(u, v)$  by evaluating another Bezier curve (for parameter  $v$ ) using the four control points defined by for row Bezier curves (for some value  $u$ ).

# Bezier Tensor Products

---

Notation:  $\mathbf{CB}(P_1, P_2, P_3, P_4, \alpha)$  is Bézier curve with control points  $P_i$  evaluated at  $\alpha$

Define “Tensor-product” Bézier surface

$$\mathbf{s}(u, v) = \mathbf{CB}(\dots$$

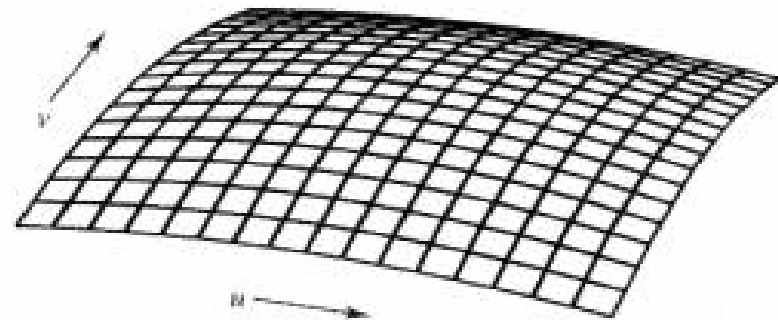
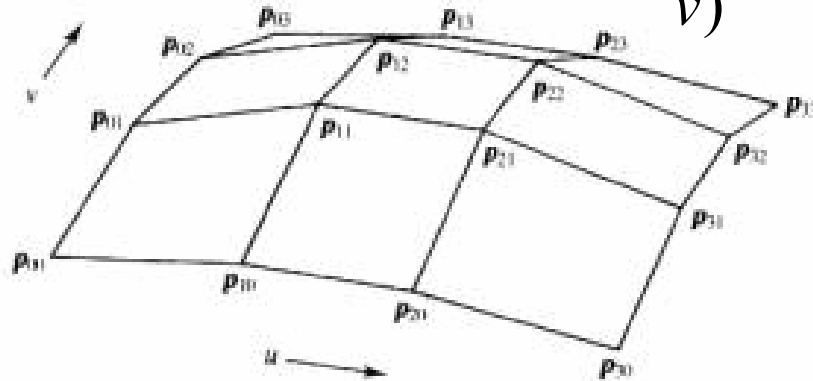
$$\mathbf{CB}(\mathbf{p}_{00}, \mathbf{p}_{01}, \mathbf{p}_{02}, \mathbf{p}_{03}, u),$$

$$\mathbf{CB}(\mathbf{p}_{10}, \mathbf{p}_{11}, \mathbf{p}_{12}, \mathbf{p}_{13}, u),$$

$$\mathbf{CB}(\mathbf{p}_{20}, \mathbf{p}_{21}, \mathbf{p}_{22}, \mathbf{p}_{23}, u),$$

$$\mathbf{CB}(\mathbf{p}_{30}, \mathbf{p}_{31}, \mathbf{p}_{32}, \mathbf{p}_{33}, u),$$

$v)$



# Basis Form

---

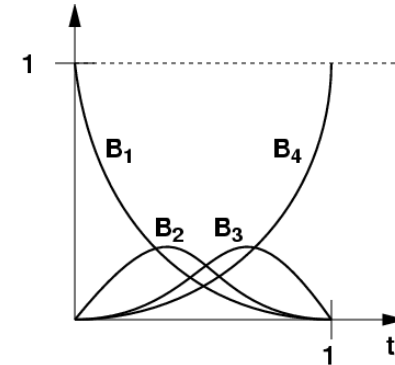
- Derivation (Buss VII.10)

$$\mathbf{s}(u, v) = \sum_{i=0}^3 \sum_{j=0}^3 B_i(u) B_j(v) \mathbf{p}_{ij}$$

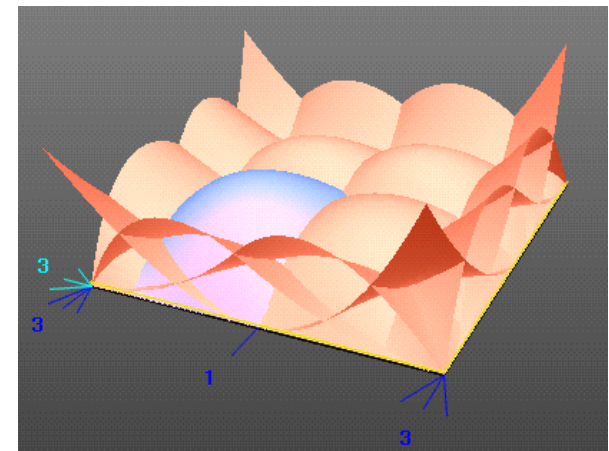
- Tensor-product basis by definition of the tensor product:

$$B_{ij}(u, v) = B_i(u) B_j(v)$$

curve basis



surface basis



# Matrix Form

---

- First coordinate only:

$$\begin{aligned} (\mathbf{s}(u, v))_1 &= \begin{pmatrix} B_0(u) & B_1(u) & B_2(u) & B_3(u) \end{pmatrix} \begin{pmatrix} (\mathbf{p}_{00})_1 & \cdots & (\mathbf{p}_{03})_1 \\ \vdots & \ddots & \vdots \\ (\mathbf{p}_{30})_1 & \cdots & (\mathbf{p}_{33})_1 \end{pmatrix} \begin{pmatrix} B_0(v) \\ B_1(v) \\ B_2(v) \\ B_3(v) \end{pmatrix} \\ &= \begin{pmatrix} u^3 & u^2 & u^1 & 1 \end{pmatrix} \mathbf{B}_{\text{bezier}} \mathbf{P} \mathbf{B}_{\text{bezier}}^T \begin{pmatrix} v^3 \\ v^2 \\ v \\ 1 \end{pmatrix} \end{aligned}$$

# Tensor-Product B-splines

---

- Use a  $m \times m$  grid of control points.
- Composed of many Bezier surface patches.

The  $(k,l)$  patch:

$$\mathbf{s}_{kl}(u, v) = \sum_{i=0}^{m-1} \sum_{j=0}^{m-1} N_i(u) N_j(v) \mathbf{p}_{ij}$$

# What You Should Know To Do

---

- List definitions and properties of polynomial curves, splines, B-splines, surfaces of revolution, generalized sweep surfaces, tensor-product surfaces.
- List definitions of C and G continuity and recognize differences visually.
- Derive analytic expressions for polynomial curves and spline from constraints indicating locations, tangents, and continuity.
- Evaluate Bezier and B-splines with geometric construction.
- Display polynomial curves and splines using line segments.