Tensor Product Bezier Patches

- Earlier work by de Casteljau (1959-63)
- Later work by Bezier made them popular for CAGD
- The simplest tensor product patch is a *bilinear patch,* based on the concept of bilinear interpolation (fitting simplest surface among 4 points)
- **Bilinear Patch**: Given 4 points, $\mathbf{b}_{00,}\, \mathbf{b}_{01}\,, \mathbf{b}_{10}\,, \mathbf{b}_{11,}$ the patch is defined as:

$$\mathbf{P}(u,w) = \sum_{i=0}^{1} \sum_{j=0}^{1} \mathbf{b}_{ij} B_i^1(u) B_j^1(w)$$

or in matrix form, it can be expressed as:

$$[1-u \quad u] \begin{bmatrix} \mathbf{b}_{00} & \mathbf{b}_{01} \\ \mathbf{b}_{10} & \mathbf{b}_{11} \end{bmatrix} \begin{bmatrix} 1-w \\ w \end{bmatrix}$$

Bilinear Tensor Product Bezier Patches

- They are called a *hyperbolic paraboloid*
- Consider the surface, $z = xy$. It is the bilinear interpolant of 4 points

$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

- If we intersect with a plane parallel to the *x,y* plane, the resulting curve is a *hyperbola*. If we intersect it with a plane containing the z-axis, the resulting curve is a *parabola*.

Direct de Casteljau Algorithm

- Extension of linear interpolation (or convex combination) algorithm for curves to surfaces
- Given a rectangular array of points, $\mathbf{b}_{ij}$, $0 \leq i,j \leq n$ and parameter value *(u,w)*. Compute a point on a surface determined by this array of points by setting:

$$\mathbf{b}_{i,j}^{r,r} = [1-u \quad u] \begin{bmatrix} \mathbf{b}_{i,j}^{r-1,r-1} & \mathbf{b}_{i,j+1}^{r-1,r-1} \\ \mathbf{b}_{i+1,j}^{r-1,r-1} & \mathbf{b}_{i+1,j+1}^{r-1,r-1} \end{bmatrix} \begin{bmatrix} 1-w \\ w \end{bmatrix}$$

r = 1,….,n

i,j = 0,….,n-r and $\mathbf{b}_{i,j}^{0,0} = \mathbf{b}_{i,j}$

$\mathbf{b}_{0,0}^{n,n}$ is the point with parameter values *(u,w)* on the surface

Direct de Casteljau Algorithm

- The net of $\mathbf{b}_{ij,}$ is called the *Bezier net* or the *control net*.
- Tensor product formulation: A surface is the locus of a curve that is moving thru space and thereby changing its shape
- The same surface can also be represented using Bernstein basis as:

$$\mathbf{P}(u,w) = \sum_{i=0}^{n} \sum_{j=0}^{n} \mathbf{b}_{ij} B_i^n(u) B_j^n(w)$$

- de Casteljau's be easily extended when the control points along *u* and *w* directions are different (say *m X n*)
  - Apply the recursive formulation till $\mathbf{b}_{i,j}^{k,k}$ is computed, where $k = \min(m,n)$
  - After that use the univariate de Casteljau's algorithm

Properties of Bezier Patches

- Affine invariance
- Convex hull property
- Boundary curves: are Bezier curves
- Variation dimishing property: Not clear whether it holds for surfaces
  - Easy to come with an counter example

Degree Elevation

- Goal: To raise the degree from *(m,n)* to *(m+1,n)*
- Compute new coefficients, $\mathbf{b}_{i,j}^{1,0}$, such that the surface can be expressed as:

$$\mathbf{P}(u,w) = \sum_{j=0}^{n} \left[\sum_{i=0}^{m+1} \mathbf{b}_{ij}^{(1,0)} B_i^{m+1}(u) \right] B_j^n(w)$$

- The *n+1* terms in square brackets represent *n+1* univariate degree elevation
- The new control points can be computed, by using the univariate formula:

$$\mathbf{b}_{i,j}^{1,0} = (1 - \tfrac{i}{m+1})\mathbf{b}_{i,j} + \tfrac{i}{m+1}\mathbf{b}_{i-1,j} \quad \{ \quad \begin{matrix} i = 0, ..., m+1 \\ j = 0, ...n \end{matrix}$$

Derivatives

- Goal: To compute partials along *u* or *w* directions
- A partial derivative is the tangent vector of an isoparametric curve:

$$\frac{\partial}{\partial u}\ \mathbf{P}(u,w) = \sum_{j=0}^{n}\ \left[\frac{\partial}{\partial u}\sum_{i=0}^{m}\mathbf{b}_{ij}B_i^m(u)\right]B_j^n(w)$$

- It can be expressed as:

$$\frac{\partial}{\partial u}\mathbf{P}(u,w) = m\sum_{j=0}^{n}\sum_{i=0}^{m-1}\ \Delta^{(1,0)}\mathbf{b}_{ij}B_i^{m-1}(u)B_j^n(w),$$

where $\Delta^{1,0}\mathbf{b}_{i,j} = \mathbf{b}_{i+1,j} - \mathbf{b}_{i,j}$