02561 Computer Graphics

Spline Surfaces and Subdivision

Jeppe Revall Frisvad

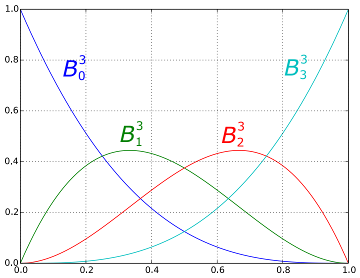
November 2020

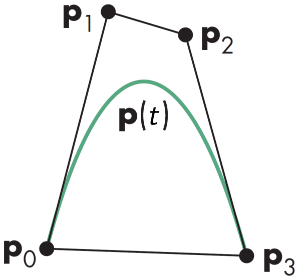
Bézier curves

- A parametric curve is a vector valued function of a single variable.
- Bernstein basis functions: $B_k^n(t) = \begin{pmatrix} n \\ k \end{pmatrix} (1-t)^{n-k} t^k$, $\quad \begin{pmatrix} n \\ k \end{pmatrix} = \frac{n!}{k!(n-k)!}$.
- Bézier curve:

  $$\boldsymbol{p}(t) = \boldsymbol{P} \begin{bmatrix} B_0^n(t) & B_1^n(t) & \cdots & B_n^n(t) \end{bmatrix}^T, \quad \boldsymbol{P} = \begin{bmatrix} \boldsymbol{p}_0 & \boldsymbol{p}_1 & \cdots & \boldsymbol{p}_n \end{bmatrix},$$

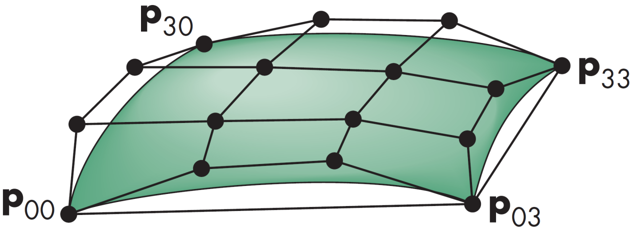
  where $\boldsymbol{p}_k$ are $n+1$ Bézier control points.





Tensor product Bézier surface

- Bézier curve: $\boldsymbol{p}(t) = \sum_{k=0}^{n} \boldsymbol{p}_k B_k^n(t)$.
- Bézier surface: $\boldsymbol{p}(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} \boldsymbol{p}_{ij} B_i^n(u) B_j^m(v)$.
- The surface is a tensor product of two curves.



Finding a point on a curve or on a surface

- Let us define operators producing a modified set of control points:

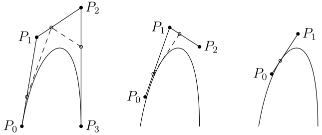
$$\begin{aligned} R &: (\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_k) \mapsto (\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_{k-1}) \\ L &: (\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_k) \mapsto (\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_k)\,. \end{aligned}$$

- We can then define the parameter-dependent *de Casteljau* operator

$$C(t) = (1-t)R + tL\,.$$

- A point on a Bézier curve is then given by n-fold application of $C(t)$:

$$\boldsymbol{p}(t) = C(t)^n \boldsymbol{P}\,.$$



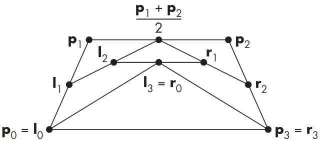
- A point on a Bézier surface is then given by

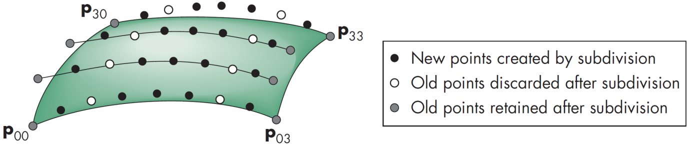
$$\boldsymbol{p}(u,v) = C_1(u)^n C_2(v)^m \boldsymbol{P}\,,$$

where $\boldsymbol{P}$ is a control point mesh.

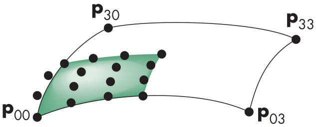
Subdivision

- The points generated by $C(\frac{1}{2})^n$ can be used as a refined control point mesh.



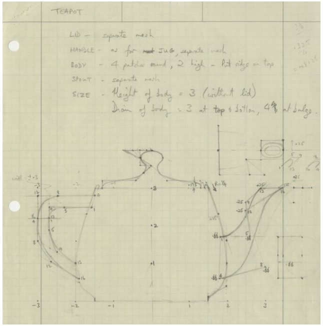


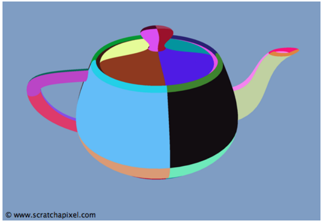
- In the case of a surface, we apply the de Casteljau operator to each curve in the tensor product.
- The result is four connected quadrants with the same number of control points as the original control point mesh.
- The control mesh generated by recursive subdivision converges to the actual Bézier curve or Bézier surface that it describes.
- We can thus use the control mesh as a subdivision surface.



Newell's teapot

- One of the first computer graphics objects to be modeled using spline surfaces.





- This Utah teapot consists of 32 bicubic Bézier patches defined by 302 vertices.
- The textbook describes an implementation of the teapot as a subdivision surface.
- Some patches are degenerate, which means that care must be taken when computing surface normals.

Differentiation to get surface normals

- The *forward difference* operator: $\Delta = L - R$.
- If applied to a Bézier curve of degree n, the result is a Bézier curve of degree $n-1$.
- We observe that $C(t) = (1-t)R + tL = R + t\Delta$, and we have $\frac{d}{dt}C(t) = \Delta$.
- For surfaces, we use a subscript to denote whether the operator pertains to rows (Δ_1) or columns (Δ_2) in the control mesh $\boldsymbol{P}$.
- We then have the following differential vectors:

$$\begin{aligned} \frac{\partial \boldsymbol{p}}{\partial u}(u,v) &= nC_1(u)^{n-1}C_2(v)^m(v)\Delta_1\boldsymbol{P} \\ \frac{\partial \boldsymbol{p}}{\partial v}(u,v) &= mC_1(u)^{n}C_2(v)^{m-1}(v)\Delta_2\boldsymbol{P} \end{aligned}$$

  and the surface normal is $\boldsymbol{n}(u,v) = \frac{\partial \boldsymbol{p}}{\partial u}(u,v) \times \frac{\partial \boldsymbol{p}}{\partial v}(u,v)$.

- We can use this to get vertex normals for a given subdivision of the control mesh.

WebGL demonstrator (**A**: 11.10)

subdivide coarsen