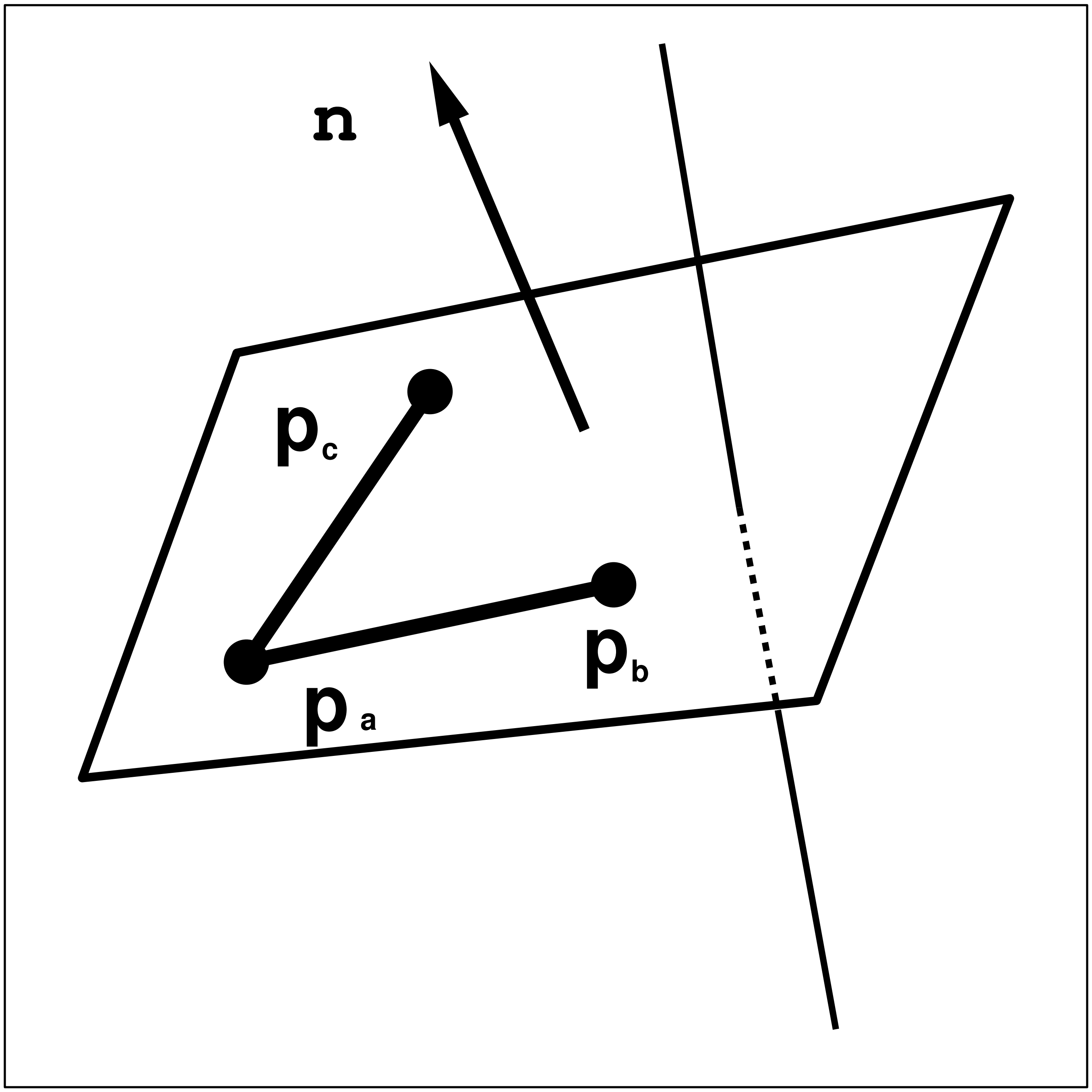
Plane and Intersection



Plane Equations

- **General plane equation**

$$ax + by + cz + d = 0$$

- **Normal of the plane**

$$\mathbf{n} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

- **Arbitrary point on the plane**

$$\mathbf{p}_a = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

- **Plane equation derivation**

$$(x - a_x)a + (y - a_y)b + (z - a_z)c = 0$$

$$ax + by + cz - (a_x a + a_y b + a_z c) = 0$$

- **parametric representation**

$$\mathbf{p}(u, v) = \mathbf{p}_a + (\mathbf{p}_b - \mathbf{p}_a)u + (\mathbf{p}_c - \mathbf{p}_a)v$$

where $\mathbf{p}_a$, $\mathbf{p}_b$, and $\mathbf{p}_c$ are non-colinear! All three points are on the plane.

- Explicit expression (if c is non-zero)

$$z = -\frac{1}{c}(ax + by + d)$$

- Line-Plane intersection

$$\mathbf{l}(u) = \mathbf{p}_0 + (\mathbf{p}_1 - \mathbf{p}_0)u$$

$$(\mathbf{n})(\mathbf{p}_0 + (\mathbf{p}_1 - \mathbf{p}_0)u) + d = 0$$

$$u = -\frac{\mathbf{n}\mathbf{p}_0}{\mathbf{n}\mathbf{p}_1 - \mathbf{n}\mathbf{p}_0} = -\frac{plane(\mathbf{p}_0)}{plane(\mathbf{p}_1) - plane(\mathbf{p}_0)}$$

- Parametric representation!

Quadric Surfaces

- **Implicit functions**

  $$a_0x^2+a_1y^2+a_2z^2+a_3xy+a_4xz+a_5yz+a_6x+a_7y+a_8z+a_9=0$$

- **Sphere**

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

- **parametric representation of sphere**

$$x=r\cos\alpha\cos\beta$$

$$y=r\cos\alpha\sin\beta$$

$$z=r\sin\alpha$$

  where $-\frac{\pi}{2}<=\alpha<=\frac{\pi}{2}$**, and** $-\pi<=\beta<=\pi$

- **Geometric meaning of these parameters**

- **Ellipsoid**

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}-1=0$$

- **Parametric representation of ellipsoid**

$$x = a \cos\alpha \cos\beta$$

$$y = b \cos\alpha \sin\beta$$

$$z = c \sin\alpha$$

where $-\frac{\pi}{2} <= \alpha <= \frac{\pi}{2}$**, and** $-\pi <= \beta <= \pi$

- **Torus**
- **Superquadrics**
- **blobby objects**

Surfaces

- **From curves to surfaces**
- **A simple curve example (Bezier)**

$$\mathbf{c}(u) = \sum_{i=0}^{3} \mathbf{p}_i B_i(u)$$

where $u \in [0, 1]$

- **Consider** $\mathbf{p}_i$ **is a curve** $\mathbf{p}_i(v)$
- **In particular, if** $\mathbf{p}_i$ **is also a bezier curve**

$$\mathbf{p}_i = \sum_{j=0}^{3} \mathbf{p}_{i,j} B_j(v)$$

where $v \in [0, 1]$

- **Then we have**

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

- **Matrix form**

$$\mathrm{s}(u,v) = \left[\begin{array}{cccc} B_0(u) & B_1(u) & B_2(u) & B_3(u) \end{array}\right]$$

$$\left[\begin{array}{cccc} \mathrm{p}_{0,0} & \mathrm{p}_{0,1} & \mathrm{p}_{0,2} & \mathrm{p}_{0,3} \\ \mathrm{p}_{1,0} & \mathrm{p}_{1,1} & \mathrm{p}_{1,2} & \mathrm{p}_{1,3} \\ \mathrm{p}_{2,0} & \mathrm{p}_{2,1} & \mathrm{p}_{2,1} & \mathrm{p}_{2,3} \\ \mathrm{p}_{3,0} & \mathrm{p}_{3,1} & \mathrm{p}_{3,2} & \mathrm{p}_{3,3} \end{array}\right] \left[\begin{array}{c} B_0(v) \\ B_1(v) \\ B_2(v) \\ B_3(v) \end{array}\right] = UMPM^\top V^\top$$

- **Tensor-product surfaces: Bi-cubic Bezier surface defined by 16 control points**

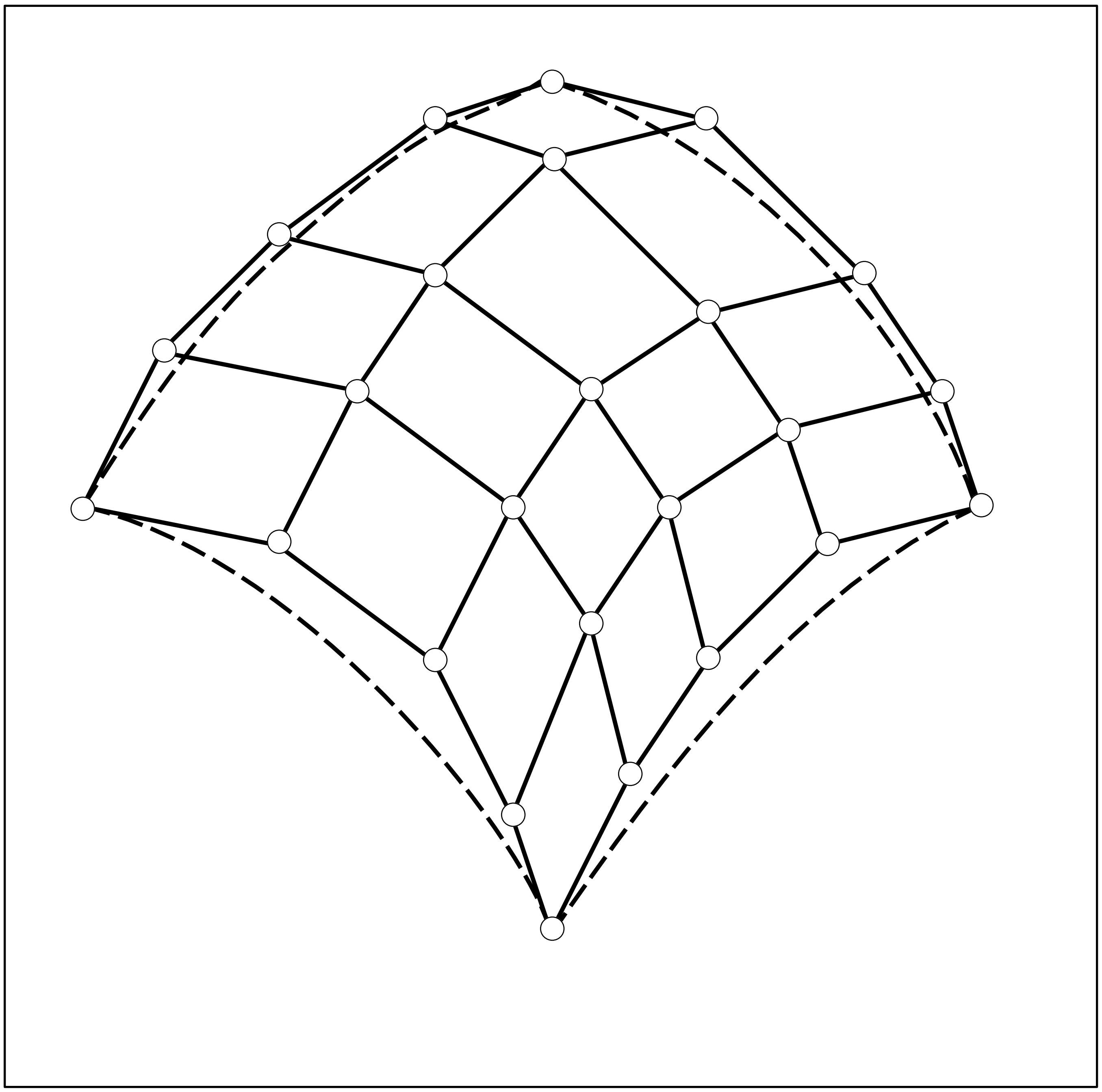
- **Further generalize to degrees of n and m along two parametric directions**

$$\mathrm{s}(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} \mathrm{p}_{i,j} B_i^n(u) B_j^m(v)$$

- **Question: which control points are interpolated!!!**

- **How about B-spline surfaces???**

Bezier Surface



B-Splines

- **B-Spline curves**

$$\mathbf{c}(u) = \sum_{i=0}^{n} \mathbf{p}_i B_{i,k}(u)$$

- **Tensor product B-splines**

$$\mathrm{s}(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} \mathbf{p}_{i,j} B_{i,k}(u) B_{j,l}(v)$$

- **Question again: which control points are interpolated???**

- **Another question: can we get NURBS surface this way???**

- **Answer: NO!!! NURBS are not tensor-product surface**

- **Another question: can we have NURBS surface?**

- **YES!!!**

- The same principle to generate curves via projection
- Idea: associated weights with control points
- Generalization of B-spline surface

NURBS

- **NURBS Surface mathematics**

$$\mathrm{s}(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m}\mathbf{p}_{i,j}w_{i,j}B_{i,k}(u)B_{j,l}(v)}{\sum_{i=0}^{n}\sum_{j=0}^{m}w_{i,j}B_{i,k}(u)B_{i,j}(v)}$$

- **Understand this geometric construction**

- **Question: why is it not the tensor-product formulation??? compare it with Bezier and B-spline construction.**

Tensor Product Surfaces

- **Where are they from?**
- **Monomial form**

$$\mathrm{s}(u,v) = \sum_i \sum_j \mathbf{a}_{i,j} u^i v^j$$

- **General case**

$$\mathrm{s}(u,v) = \sum_i \sum_j \mathbf{v}_{i,j} F_i(u) G_j(v)$$

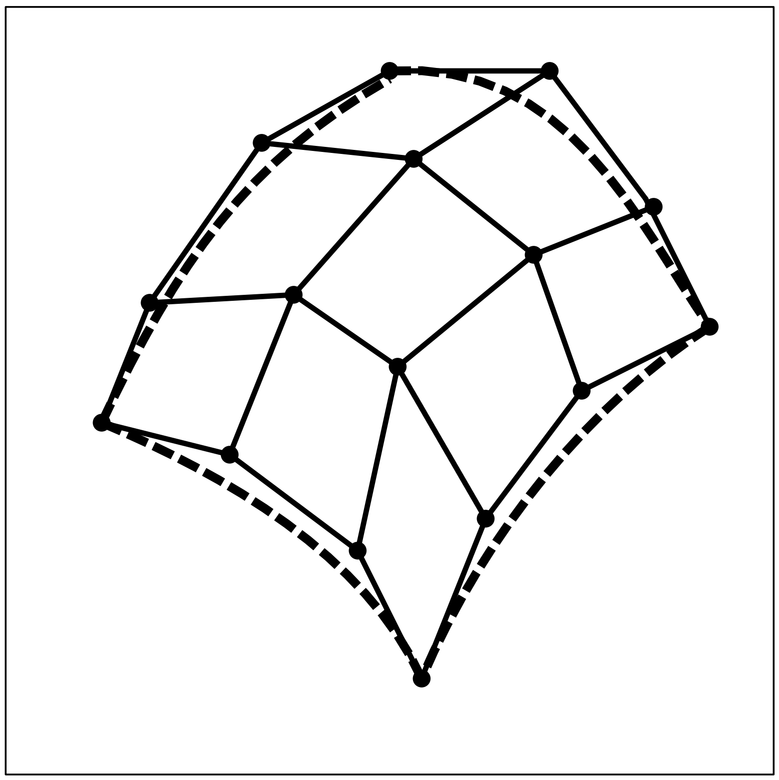
- **Bezier surface**

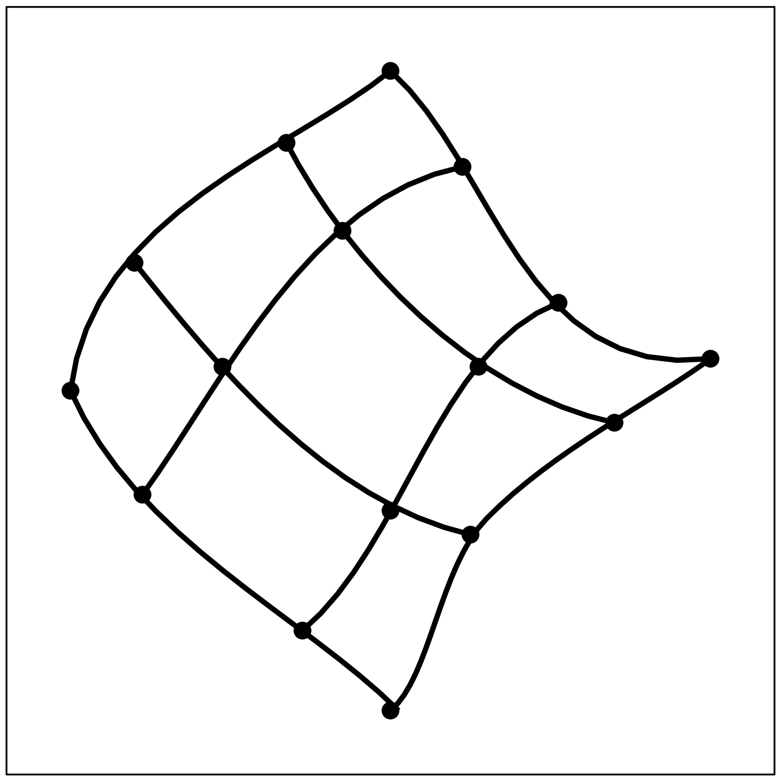
$$\mathrm{s}(u,v) = \sum_i \sum_j \mathbf{p}_{i,j} B_i^m(u) B_j^n(v)$$

- **B-spline surface**

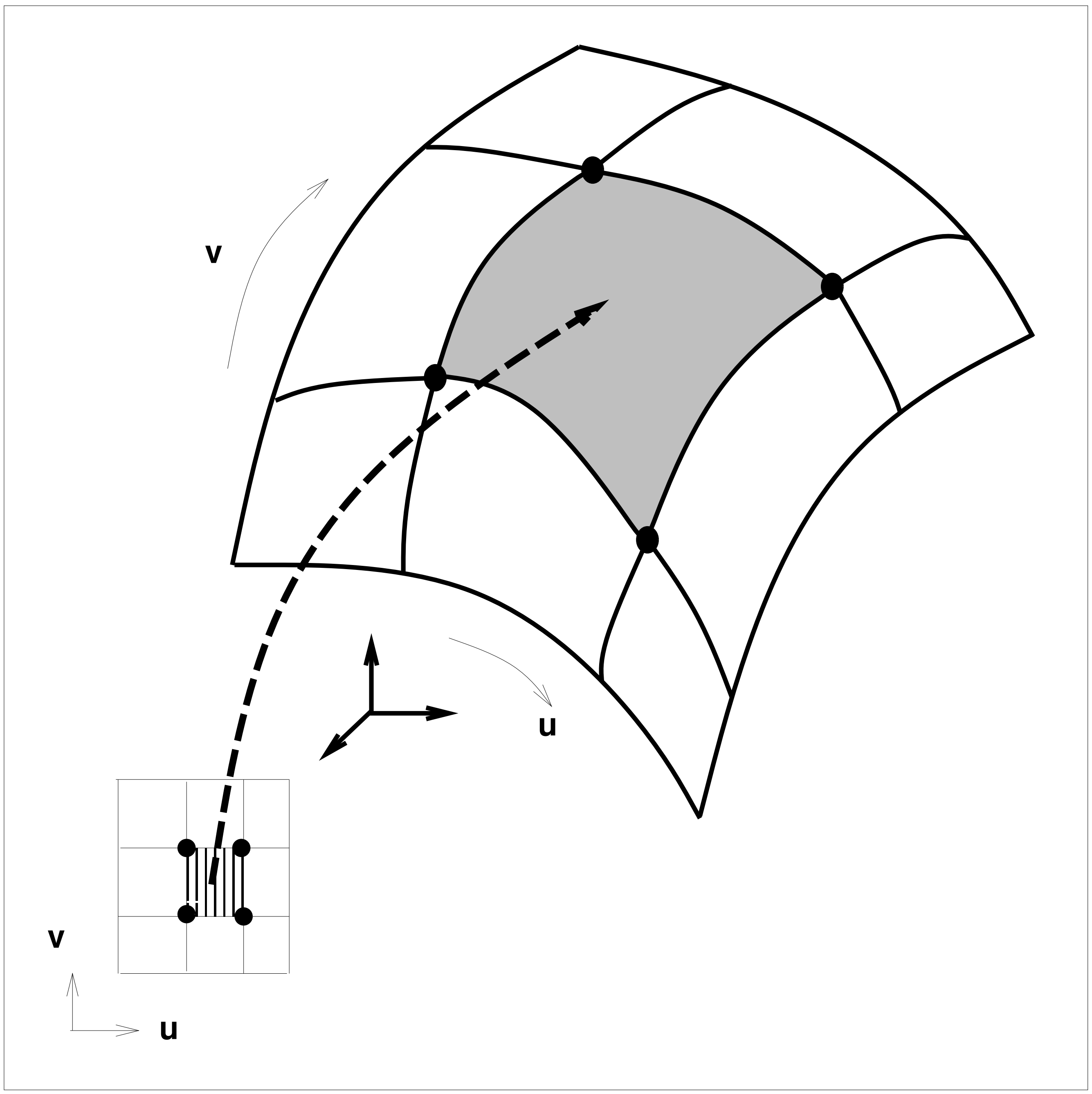
$$\mathrm{s}(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{p}_{i,j} B_{i,k}(u) B_{j,l}(v)$$

Regular Surfaces





Rectangular Surface



Surface Properties

- Inherit from their curve generators
- More!
- Efficient algorithms
- Continuity across boundaries
- Interpolation & approximation tools

Hermite Surfaces

- **How about hermite surfaces???**
- **Hermite curve**

$$\mathbf{c}(u) = \begin{bmatrix} f_0(u) & f_1(u) & f_2(u) & f_3(u) \end{bmatrix} = \begin{bmatrix} \mathbf{c}(0) \\ \mathbf{c}(1) \\ \mathbf{c}'(0) \\ \mathbf{c}'(1) \end{bmatrix}$$

- $\mathrm{c}(0)$ **is a curve** $\mathrm{s}(0, v)$ **which is also a Hermite curve:**

$$\mathrm{s}(0, v) = \begin{bmatrix} f_0(v) & f_1(v) & f_2(v) & f_3(v) \end{bmatrix} \begin{bmatrix} \mathrm{s}(0,0) \\ \mathrm{s}(0,1) \\ \mathrm{s}_v(0,0) \\ \mathrm{s}_v(0,1) \end{bmatrix}$$

- **Similarly,** $\mathrm{c}(1)$ **is a curve** $\mathrm{s}(1, v)$ **which is also a hermite curve:**

$$\mathrm{s}(1, v) = \begin{bmatrix} f_0(v) & f_1(v) & f_2(v) & f_3(v) \end{bmatrix} = \begin{bmatrix} \mathrm{s}(1,0) \\ \mathrm{s}(1,1) \\ \mathrm{s}_v(1,0) \\ \mathrm{s}_v(1,1) \end{bmatrix}$$

- **The same are for $\mathbf{c}'(0)$ and $\mathbf{c}'(1)$**

$$\mathbf{s}_u(0,v) = V \begin{bmatrix} \mathbf{s}_u(0,0) \\ \mathbf{s}_u(0,1) \\ \mathbf{s}_{uv}(0,0) \\ \mathbf{s}_{uv}(0,1) \end{bmatrix}$$

$$\mathbf{s}_v(1,v) = V \begin{bmatrix} \mathbf{s}_u(1,0) \\ \mathbf{s}_u(1,1) \\ \mathbf{s}_{uv}(1,0) \\ \mathbf{s}_{uv}(1,1) \end{bmatrix}$$

- **It's time to put them together!**

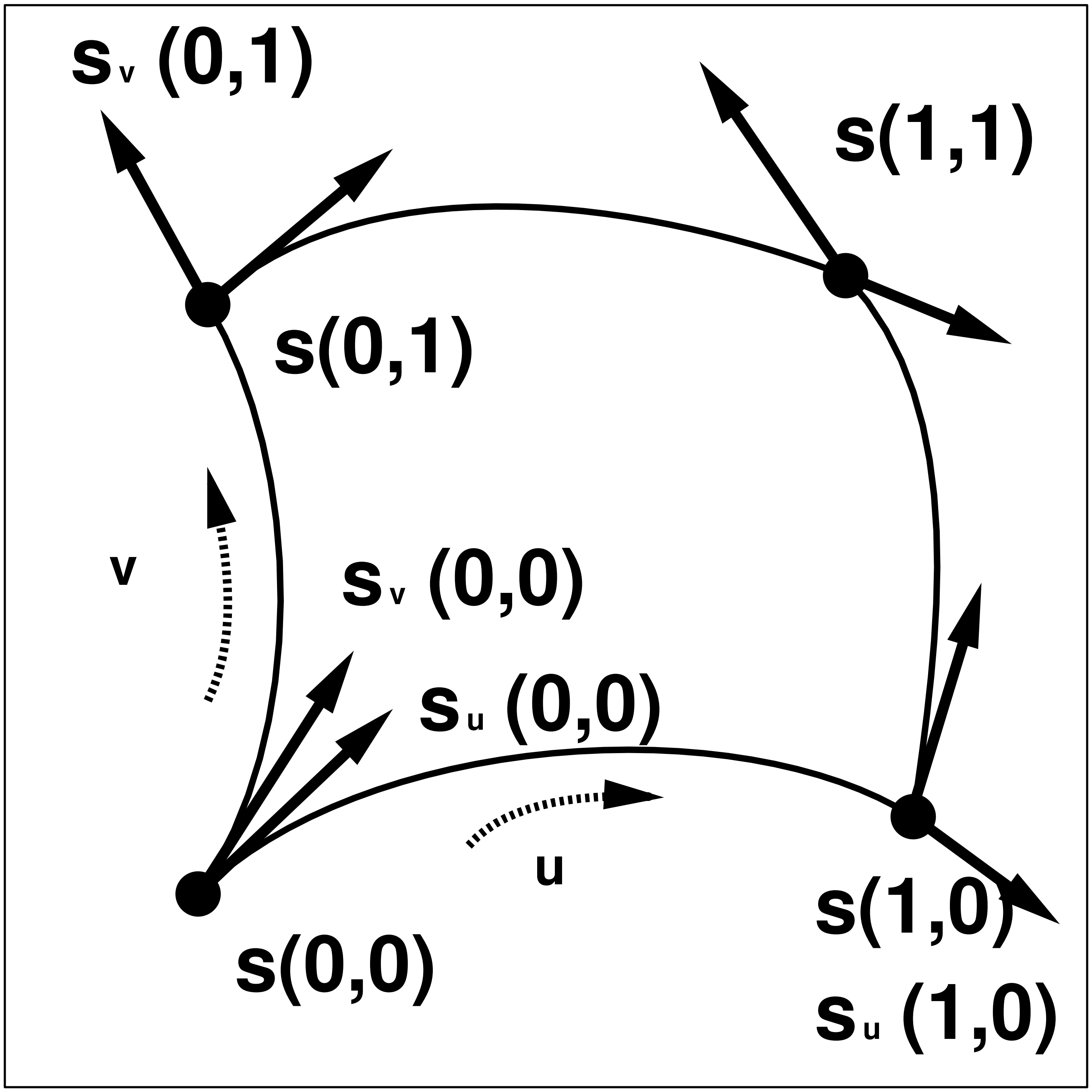
$$\mathbf{s}(u,v) = U \begin{bmatrix} \mathbf{s}(0,0) & \mathbf{s}(0,1) & \mathbf{s}_v(0,0) & \mathbf{s}_v(0,1) \\ \mathbf{s}(1,0) & \mathbf{s}(1,1) & \mathbf{s}_v(1,0) & \mathbf{s}_v(1,1) \\ \mathbf{s}_u(0,0) & \mathbf{s}_u(0,1) & \mathbf{s}_{uv}(1,0) & \mathbf{s}_{uv}(1,0) \\ \mathbf{s}_u(1,0) & \mathbf{s}_u(1,1) & \mathbf{s}_{uv}(1,0) & \mathbf{s}_{uv}(1,1) \end{bmatrix}$$

- **Continuity conditions for surfaces**

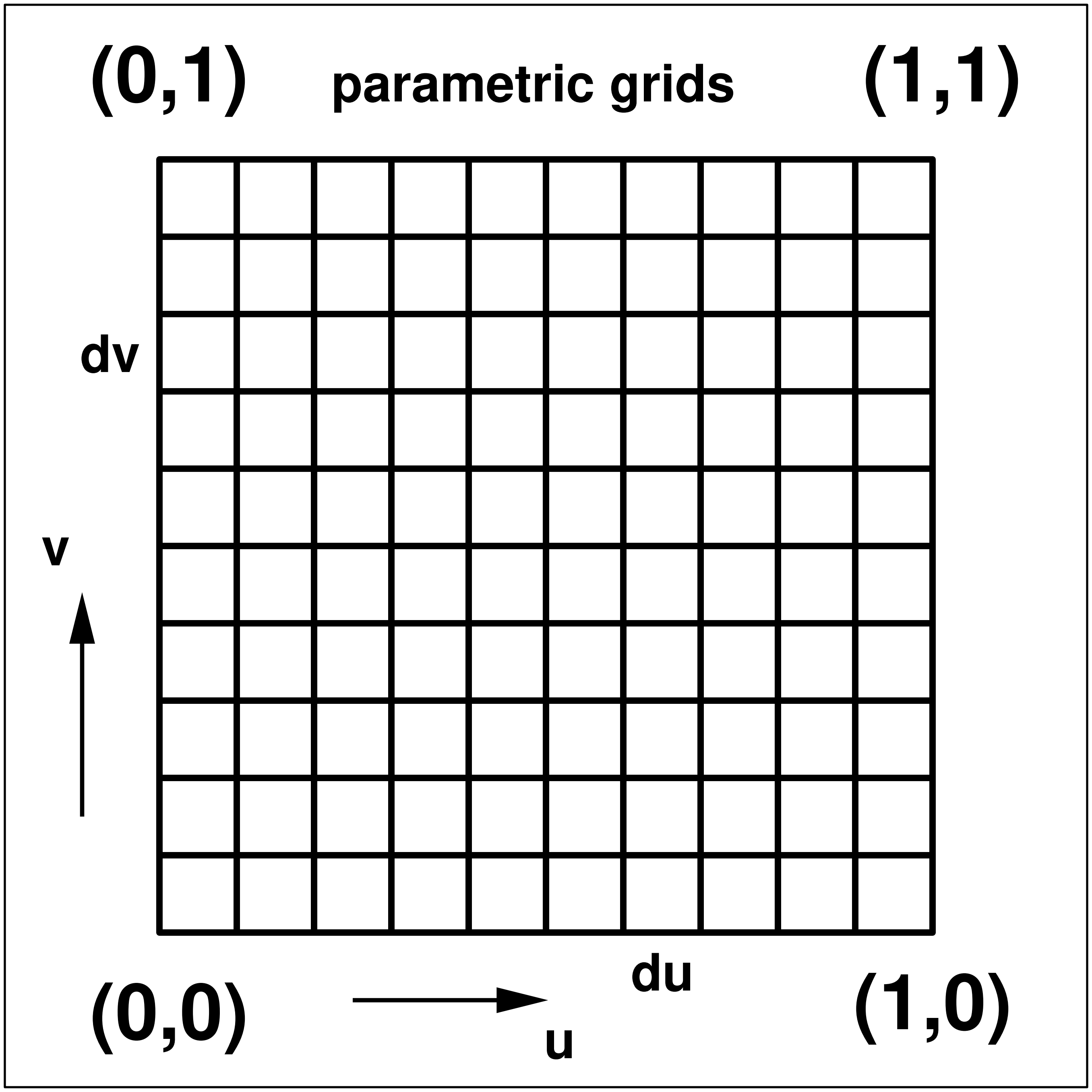
- **Bezier surfaces, b-splines, NURBS, Hermite surfaces**

- **C^1 and G^1 continuity**

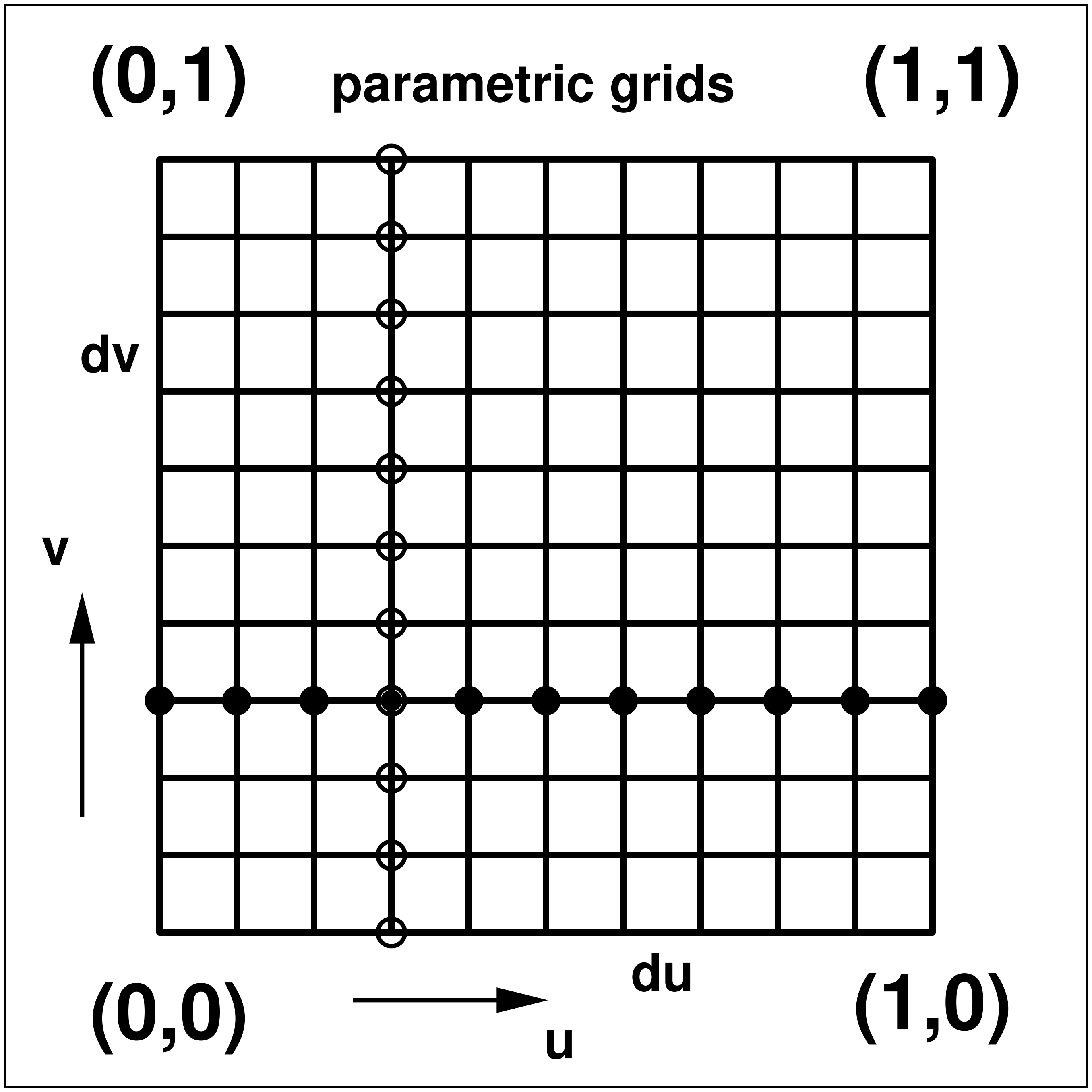
Hermite Surfaces



Surface Rendering



Surface Rendering



Surface (Patch) Rendering

- We use bicubic as an example
- The simplest method (naive): convert a bicubic patches into primitives that we always know how to render
- From curved surfaces to polygon (quadrilaterals (non-planar) and/or triangles (planar)
- Surface evaluation at grid points
- This is straight forward but inefficient
- Because it require many times of evaluation of $\mathrm{s}(u, v)$
- The total number is

$$3\frac{1}{\delta u}\frac{1}{\delta v}$$

- **Better approach: precomputation**

$$\mathrm{s}(u,v) = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} M \begin{bmatrix} v^3 \\ v^2 \\ v \\ 1 \end{bmatrix}$$

- M **is constant throughout the entire patch**

- **The following is the same along isoparametric lines**

$$\begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix}$$

$$\begin{bmatrix} v^3 & v^2 & v & 1 \end{bmatrix}$$

- **Use one dimensional array to compute and store (evaluation only once)**

- **How about many patches: the array is unchanged, if sampling rate is the same, this is more useful.**

- **How about adaptive sampling based on curvature information!!!**

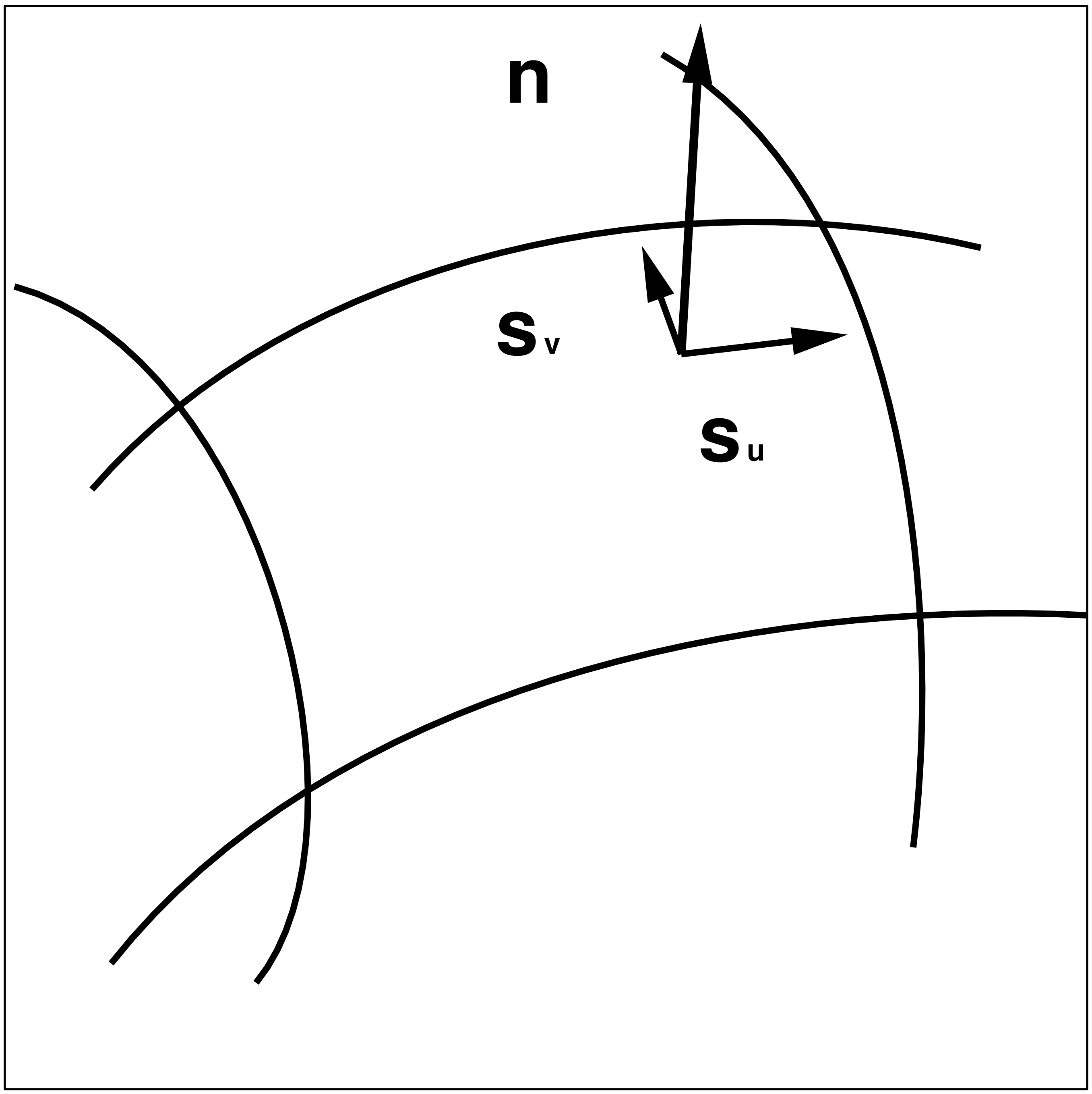
- **How to compute normal at any grid point**

(approximation)

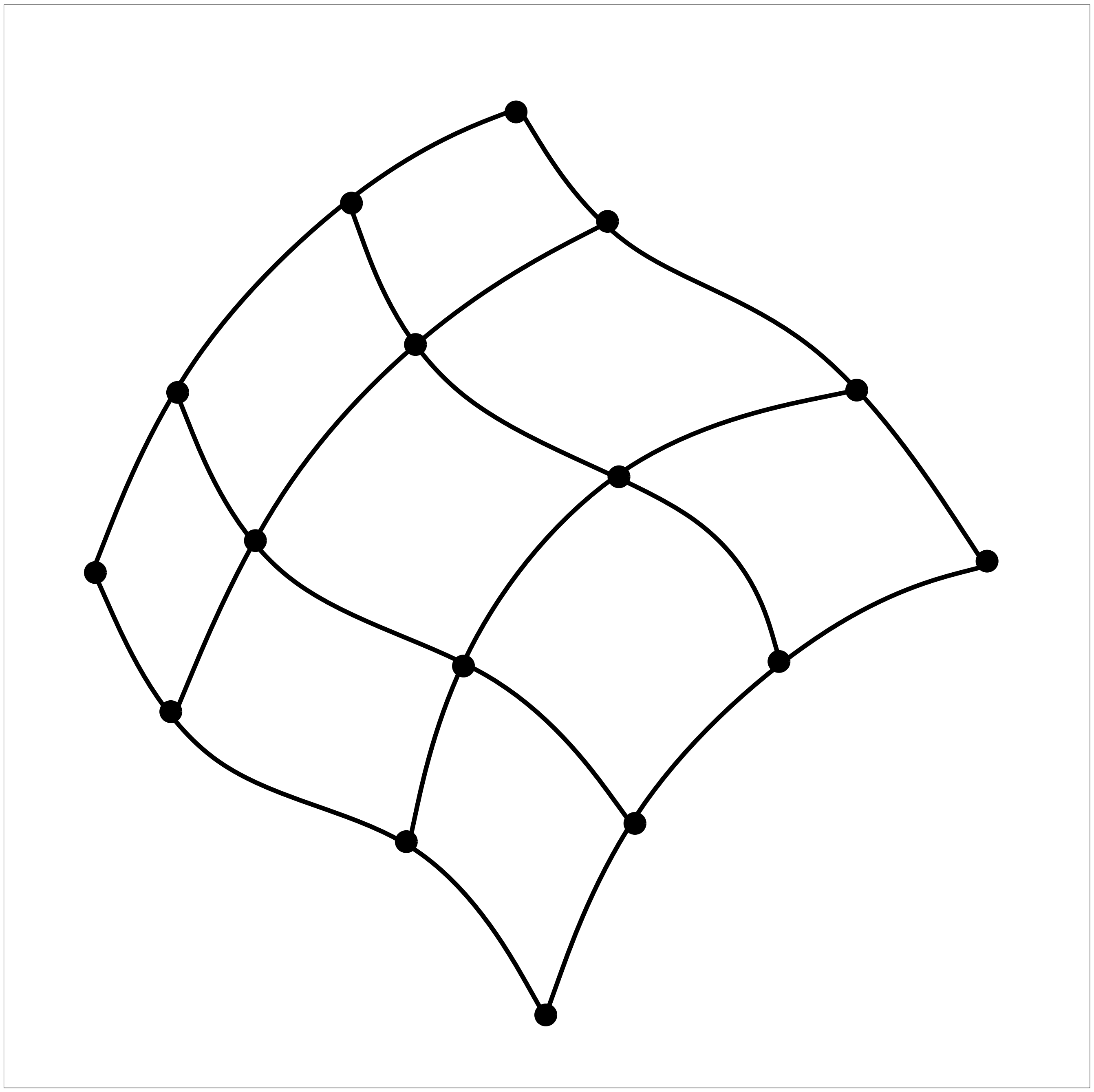
$$\mathbf{s}_u(u,v) \times \mathbf{s}_v(u,v)$$

$$(\mathbf{s}(u+\delta u, v) - \mathbf{s}(u,v)) \times (\mathbf{s}(u, v+\delta v) - \mathbf{s}(u,v))$$

Surface Normal



Curve Network



Transfinite Methods

- **Bilinearly blended Coons patch**
  - **interpolate four boundary curves**
- **Bicubically blended Coons patch**
- **Gordon surfaces**
  - **interpolate a curve network**
- **Triangular extension**
  - **interpolate over triangles**

Coons Patch

- **Bilinearly blended Coons patch**

$$(P)\mathbf{f} = (P_1 \oplus P_2)\mathbf{f} = (P_1 + P_2 - P_1 P_2)\mathbf{f}$$

$$(P_1)\mathbf{f} = \mathbf{f}(0,v)L_0^1(u) + \mathbf{f}(1,v)L_1^1(u)$$

$$(P_2)\mathbf{f} = \mathbf{f}(u,0)L_0^1(v) + \mathbf{f}(u,1)L_1^1(v)$$

- **Bicubically blended Coons patch**

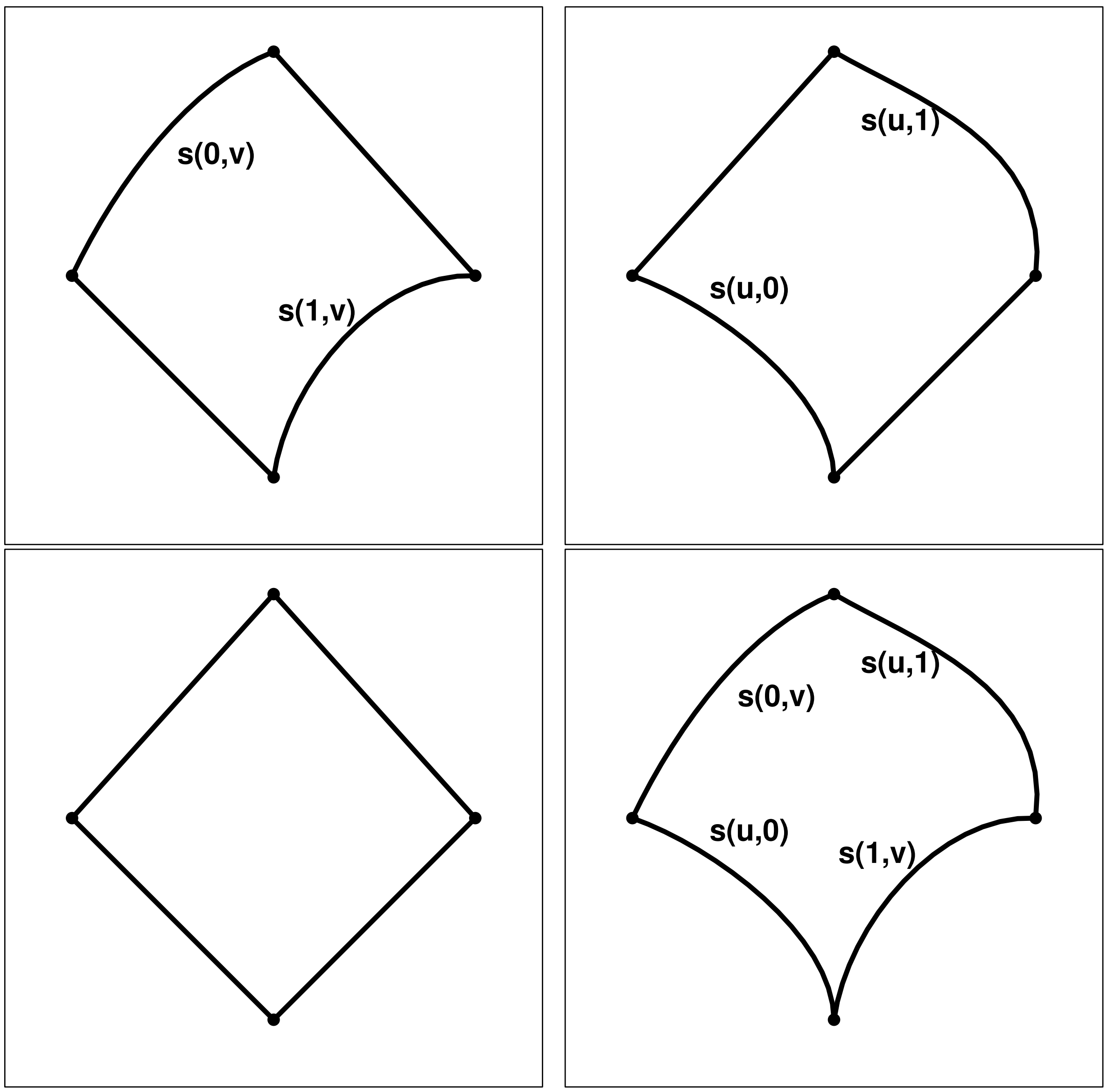
$$(P_1)\mathbf{f} = \mathbf{f}(0,v)H_0^3(u) + \mathbf{f}_u(0,v)H_1^3(u)$$

$$+\mathbf{f}_u(1,v)H_2^3(u) + \mathbf{f}(1,v)H_3^3(u)$$

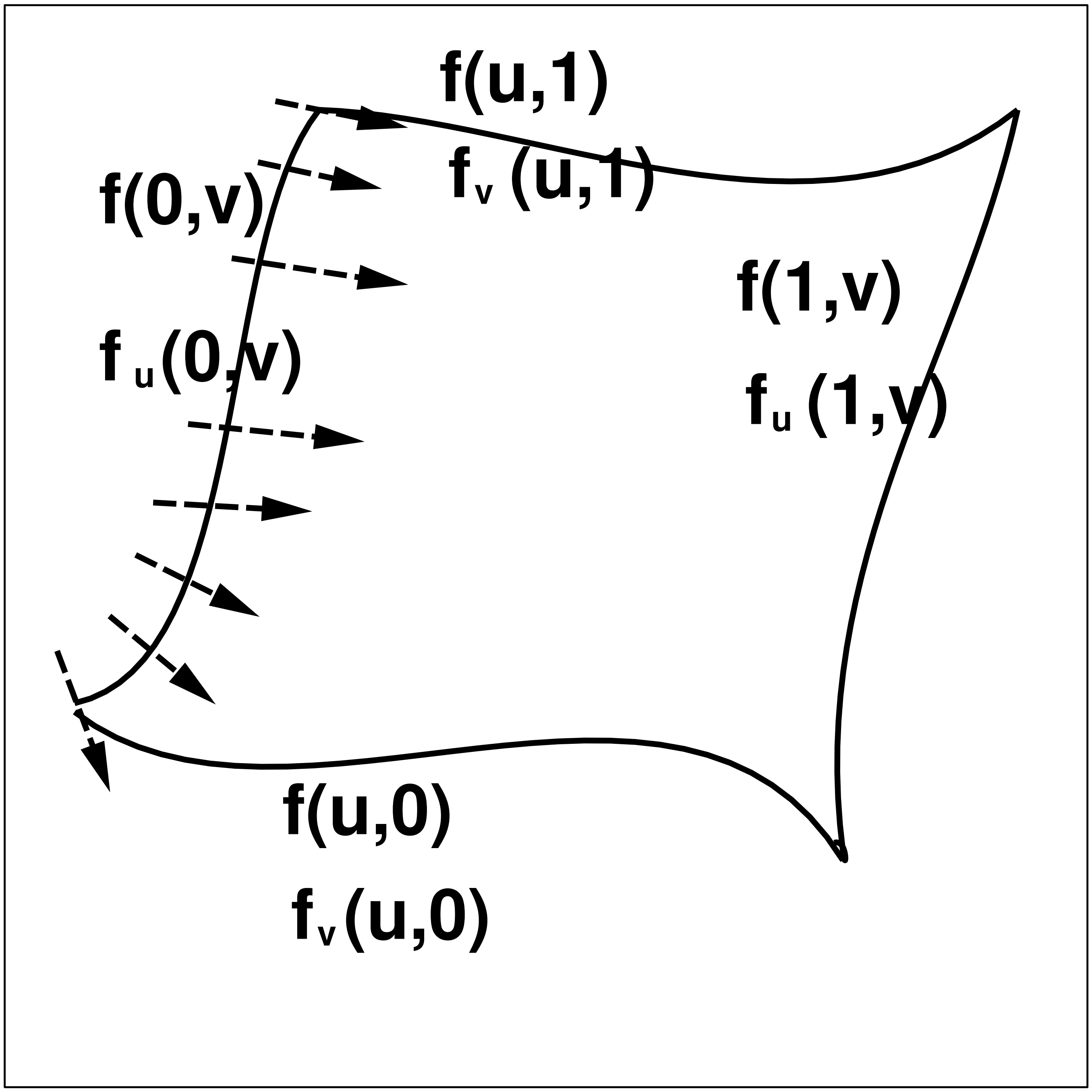
$$(P_2)\mathbf{f} = \mathbf{f}(u,0)H_0^3(v) + \mathbf{f}_v(u,1)H_1^3(v)$$

$$+\mathbf{f}_v(u,1)H_2^3(v) + \mathbf{f}(u,1)H_3^3(v)$$

Coons Patch



<u>Coons Patch</u>



Gordon Surfaces

- Generalization of Coons techniques
- A set of curves

$$\mathbf{f}(u_i, v), i = 0, \ldots, n$$

$$\mathbf{f}(u, v_j), j = 0, \ldots, m$$

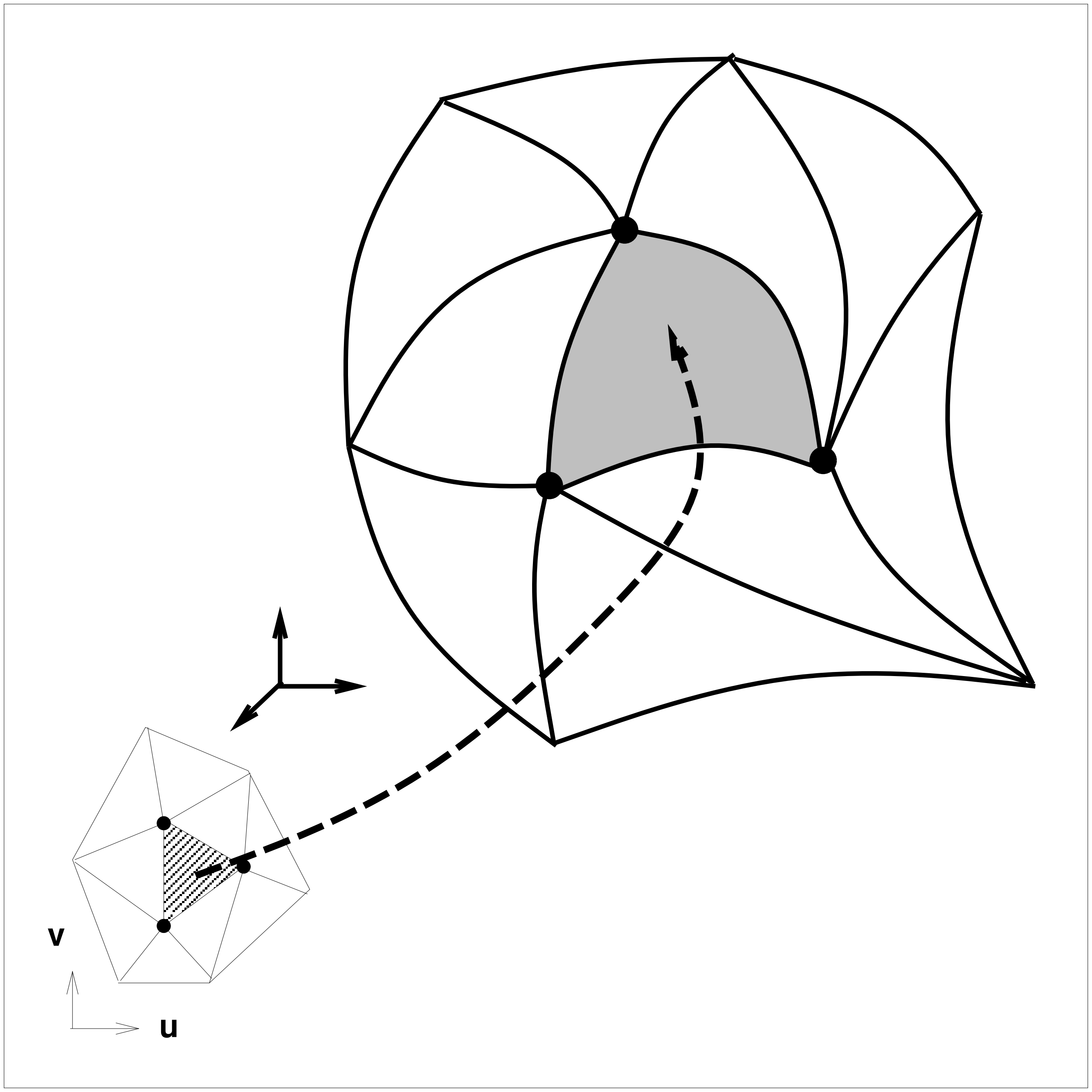
- Boolean sum using Lagrange polynomials

$$(P_1)\mathbf{f} = \sum_{i=0}^{n} \mathbf{f}(u_i, v) L_i^n(u)$$

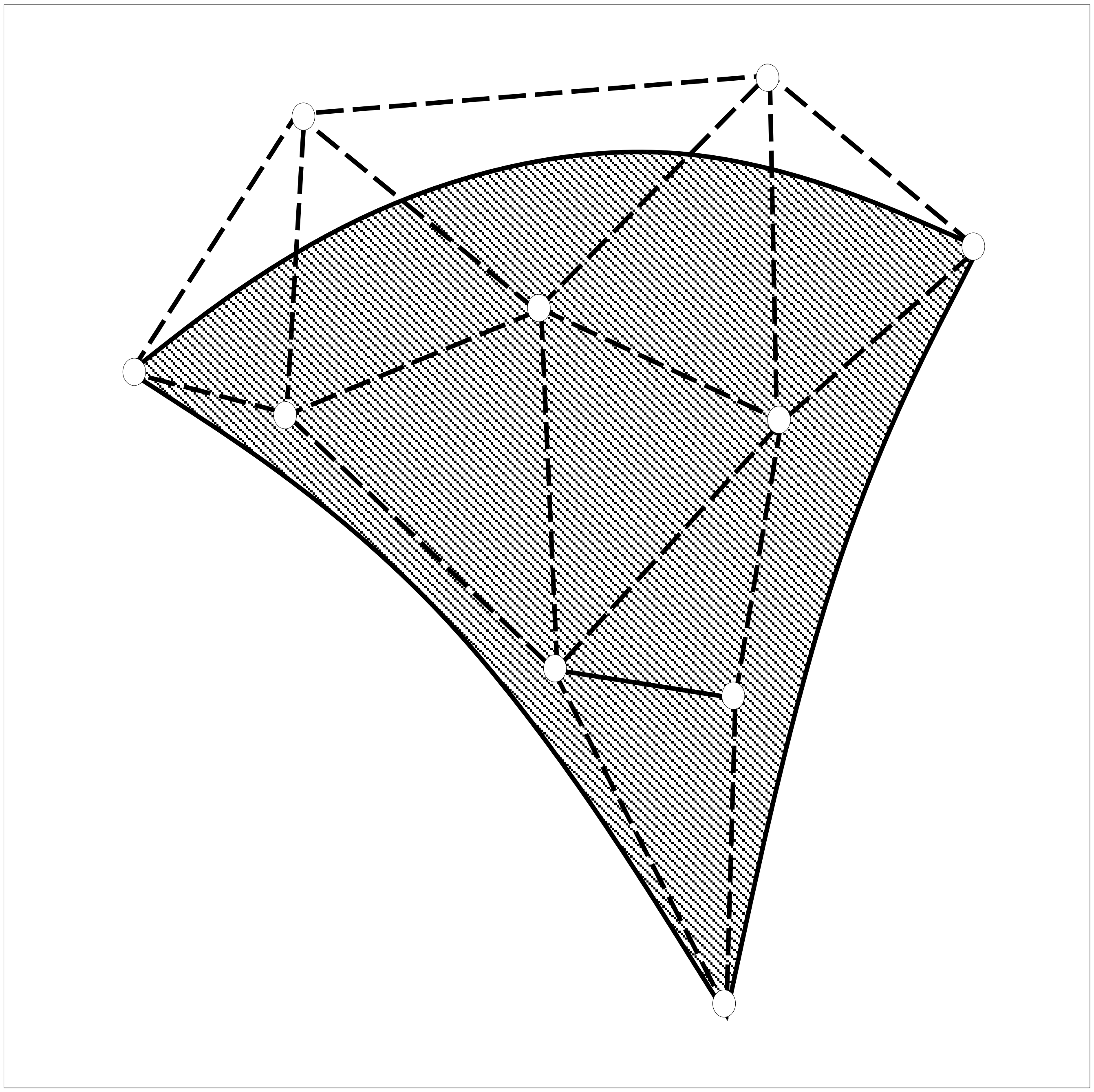
$$(P_2)\mathbf{f} = \sum_{j=0}^{m} \mathbf{f}(u, v_j) L_j^m(v)$$

$$(P)\mathbf{f} = (P_1 \oplus P_2)\mathbf{f} = (P_1 + P_2 - P_1 P_2)\mathbf{f}$$

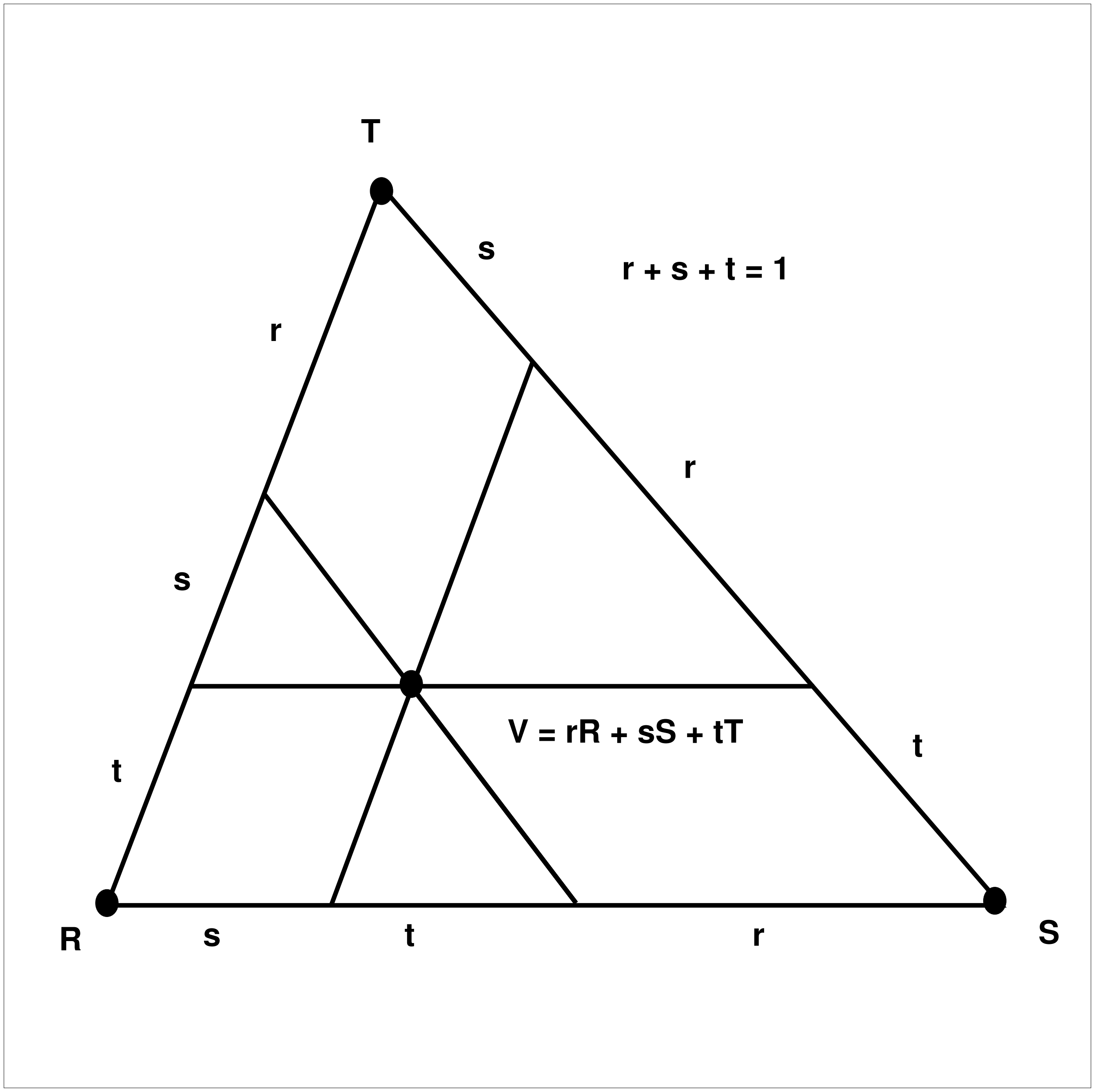
Triangular Surfaces



Triangular Bezier Surface



Barycentric Coordinates



Bezier Formulation

- **Triangular Bezier Surface**

$$\mathrm{s}(u,v) = \sum_{i,j,k>=0} \mathbf{p}_{i,j,k} B^n_{i,j,k}(r,s,t),$$

  where $i+j+k=n$, $r+s+t=1$, and they are local barycentric coordinates

- **Basis functions**

$$B^n_{i,j,k}(r,s,t) = \frac{n!}{i!j!k!} r^i s^j t^k$$

- **Properties**
  - **recursive evaluation**
  - **directional derivatives**
  - **degree elevation**
  - **subdivision**
  - **composite surfaces**

Triangular Interpolation

- $(P_1)\mathbf{f} = \mathbf{f}(r,0,t)L_0^1(\alpha) + \mathbf{f}(r,s,0)L_1^1(\alpha)$,
  where $\alpha = \frac{s}{s+t}$

- $(P_2)\mathbf{f} = \mathbf{f}(r,s,0)L_0^1(\beta) + \mathbf{f}(0,s,t)L_1^1(\beta)$,
  where $\beta = \frac{t}{r+t}$

- $(P_3)\mathbf{f} = \mathbf{f}(0,s,t)L_0^1(\gamma) + \mathbf{f}(r,0,t)L_1^1(\gamma)$,
  where $\gamma = \frac{r}{r+s}$

- **The Boolean sum of any two operators results the same!**

$$(P_{12})\mathbf{f} = (P_1 \oplus P_2)\mathbf{f}$$

$$(P_{13})\mathbf{f} = (P_1 \oplus P_3)\mathbf{f}$$

$$(P_{23})\mathbf{f} = (P_2 \oplus P_3)\mathbf{f}$$

- **Use cubic blending functions for C^1 interpolation!**

$$(Q_1)\mathbf{f} = \mathbf{f}(r,0,t)H_0^3(\alpha) + D_\alpha\mathbf{f}(r,0,t)H_1^3(\alpha) + D_\alpha\mathbf{f}(r,s,0)H_2^3(\alpha) + \mathbf{f}(r,s,0)H_3^3(\alpha)$$

$$(Q_2)\mathbf{f} = \ldots$$

$$(Q_3)\mathbf{f} = \dots$$

- **Convex combination (Gregory's method)**

$$(T_1)\mathbf{f} = \mathbf{f}(r, 0, t) + \alpha D_\alpha \mathbf{f}(r, 0, t)$$

$$(T_2)\mathbf{f} = \dots$$

$$(T_3)\mathbf{f} = \dots$$

$$(T_{12})\mathbf{f} = (T_1 \oplus T_2)\mathbf{f}$$

$$(T_{13})\mathbf{f} = \dots$$

$$(T_{23})\mathbf{f} = \dots$$

$$(T)\mathbf{f} = (a_1 T_{23} + a_2 T_{13} + a_3 T_{12})\mathbf{f}$$

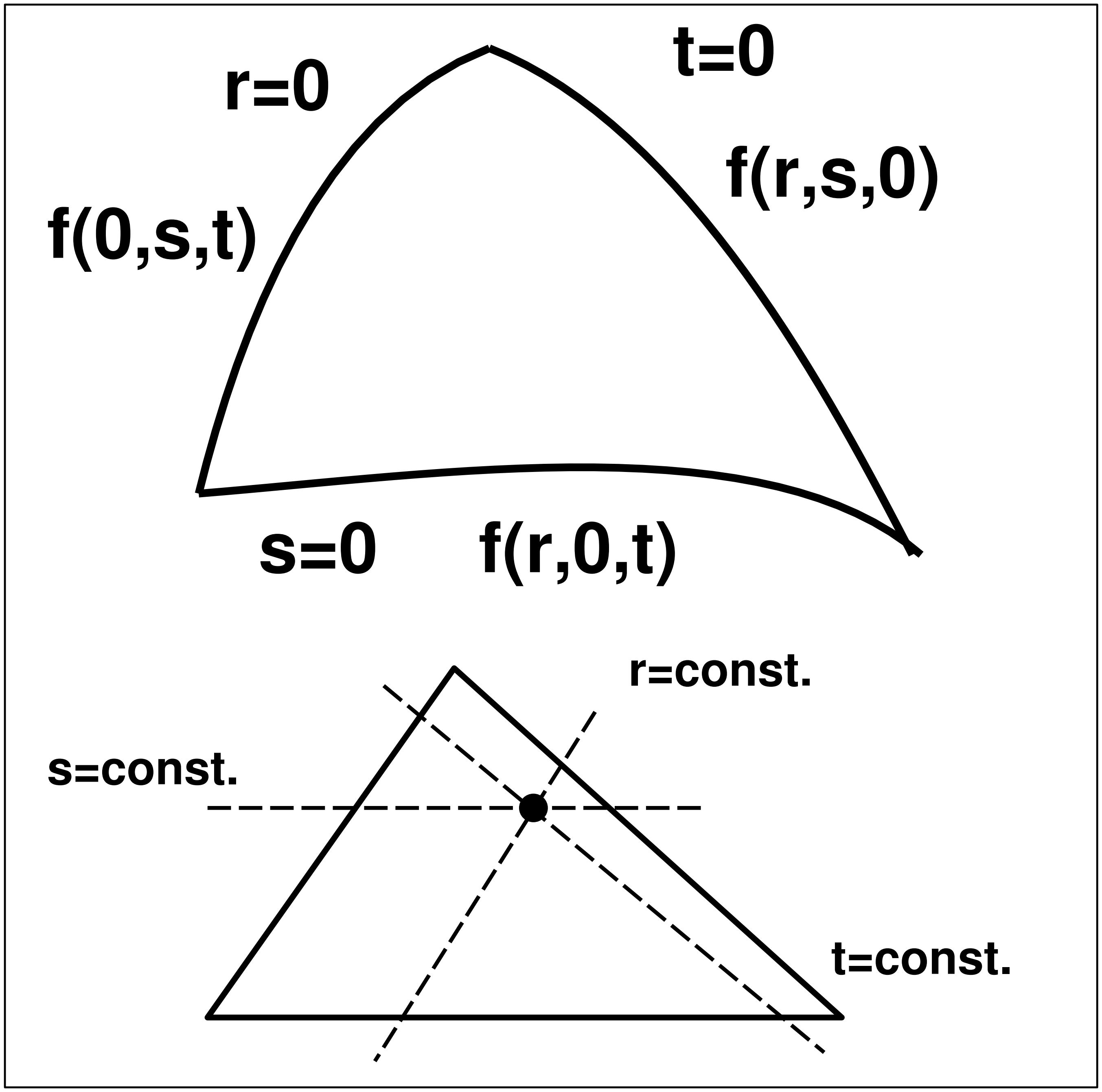
$$a_1 = \frac{s^2}{r^2 + s^2 + t^2}$$

$$a_2 = \dots$$

$$a_3 = \dots$$

- **Pentagonal patch!**

Triangular Coons-Gordon Surface



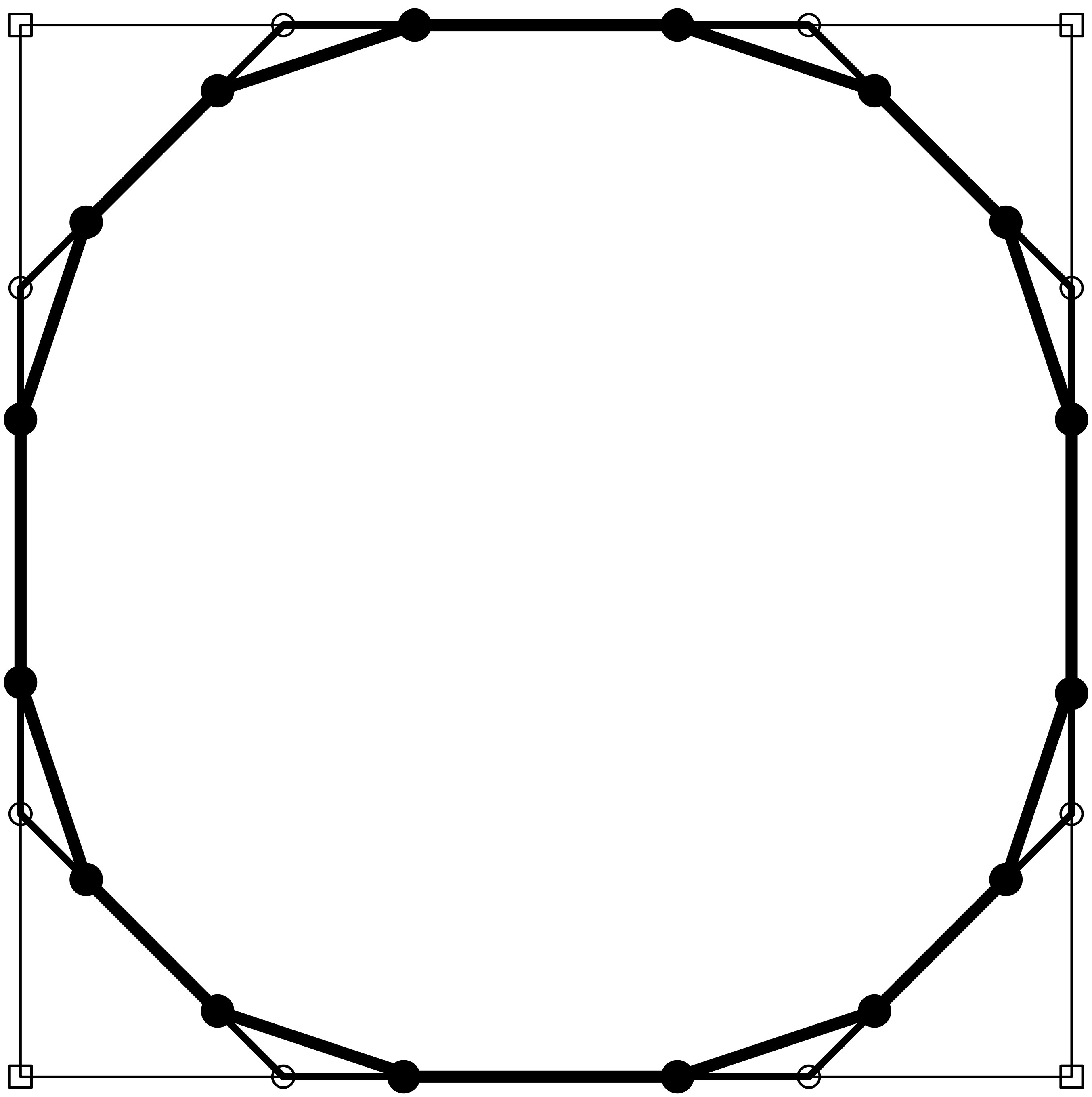
Motivation

- Why splines (instead of pixels or voxels)
- Much higher and easy managable level of abstraction
- We are dealing with features (tangent, orientation, curvature, corners, etc)
- Features are invariant under standard geometric transformation

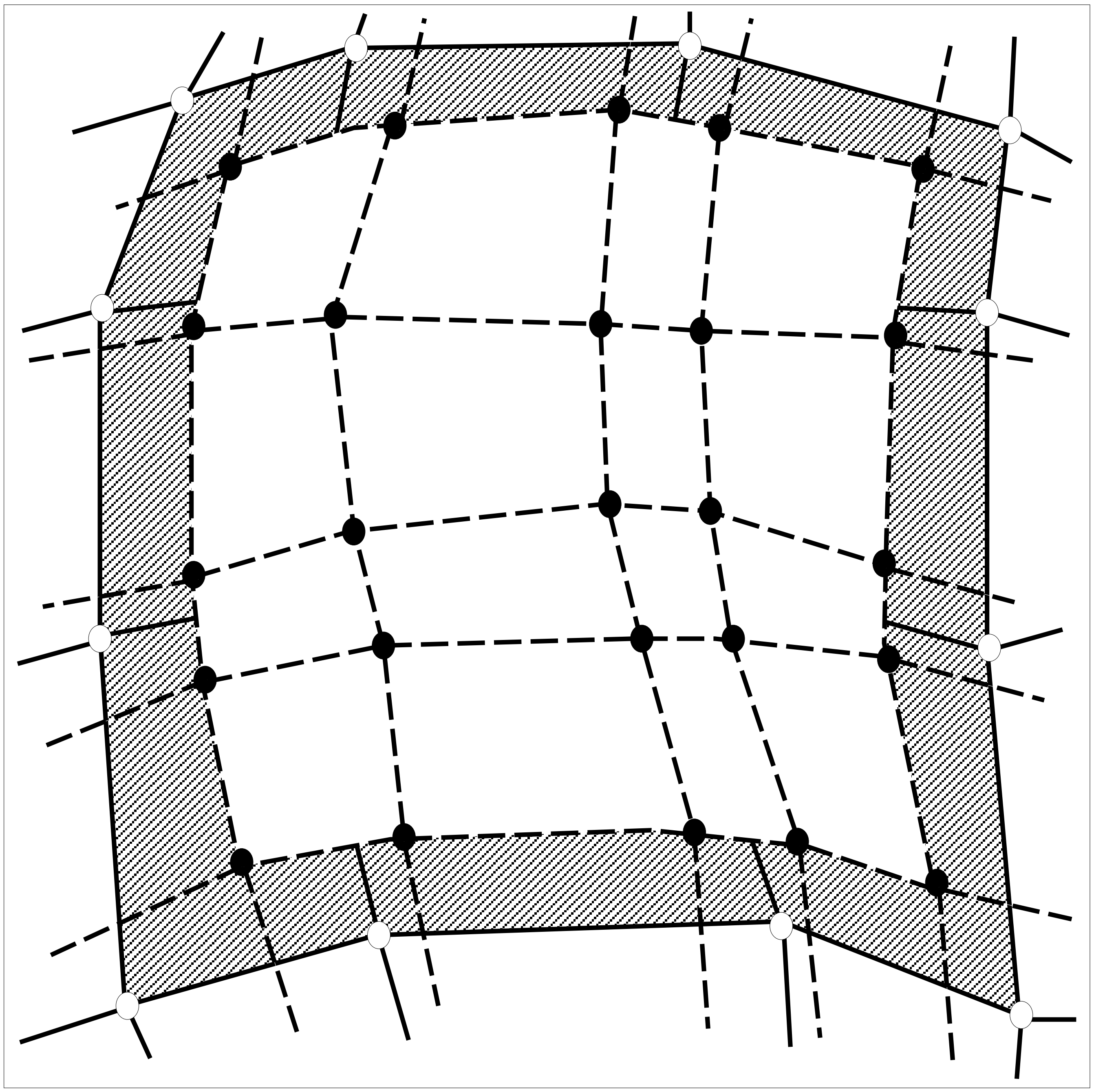
Surface Summary

- How many ways to construct surfaces???
- Multiple patches
- "Carpet method"
- A set of boundary curves
- Subdivision
- Others!

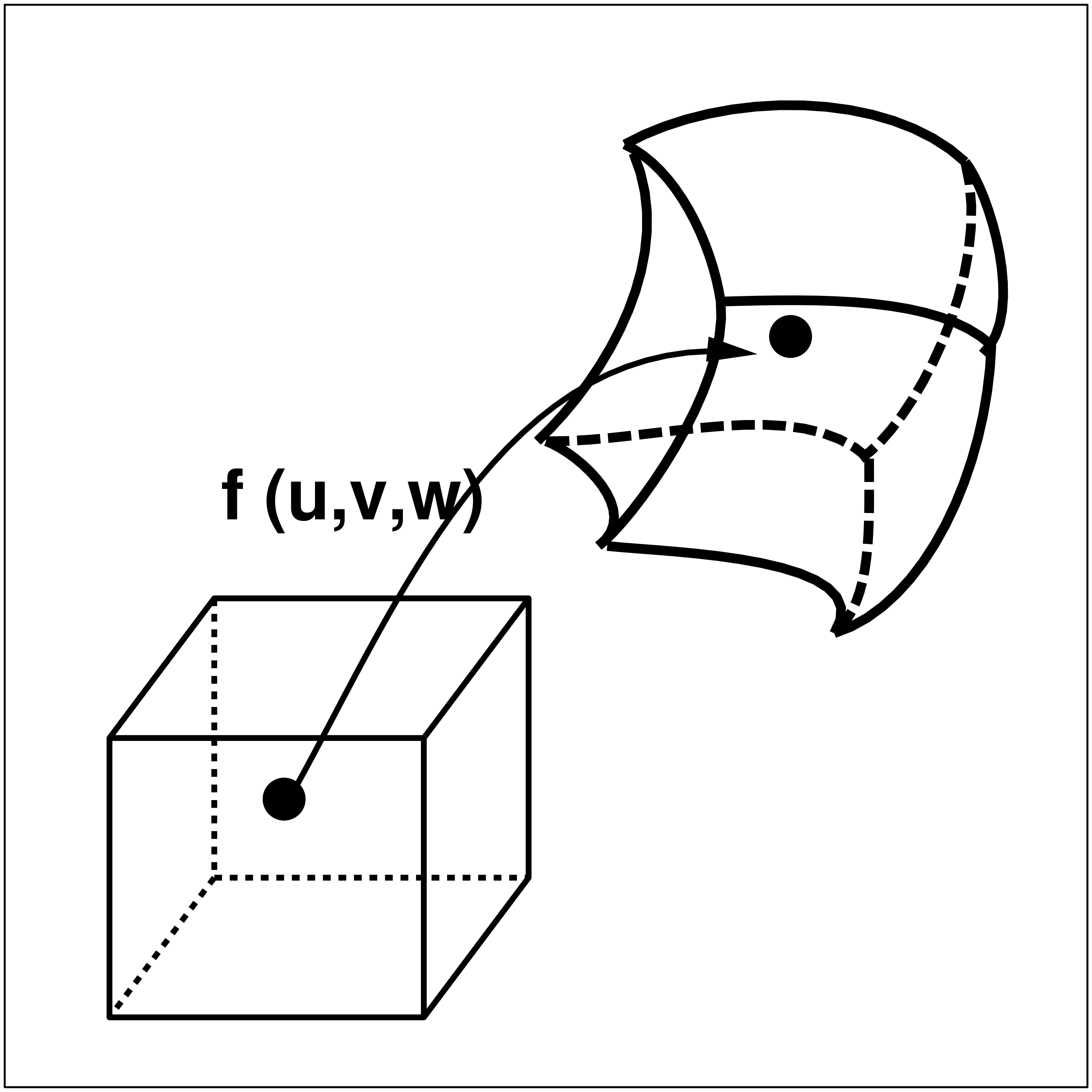
Subdivision Forms



Subdivision Surface



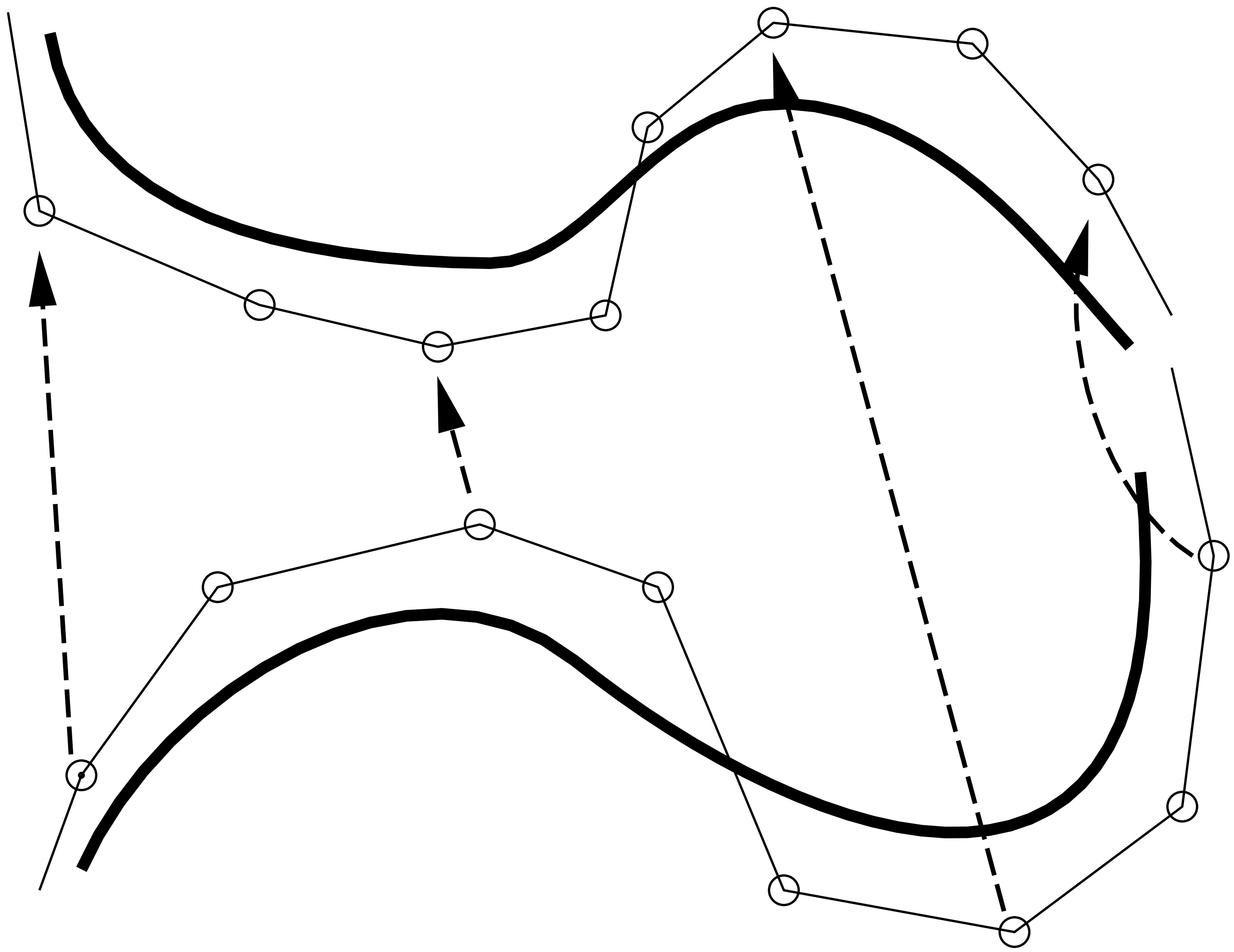
Solid



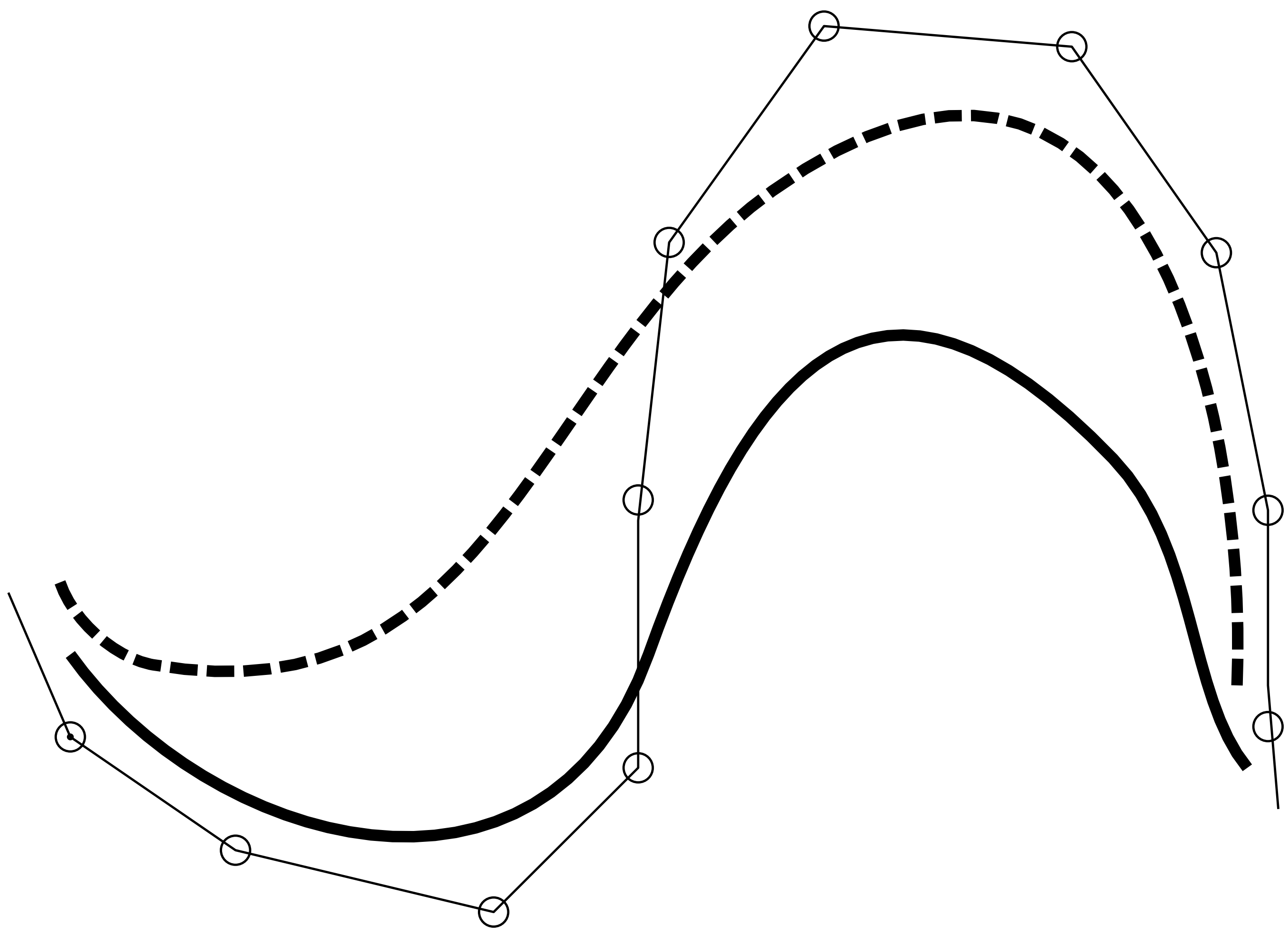
Geometric Modeling Techniques

- Interpolation/Approximation
  - regular dataset
  - scattered dataset
  - curve network
- Optimization
- Interaction
- Constraints
- Continuity
- Differential geometry
- Other advanced operations
  - blending, rounding, etc

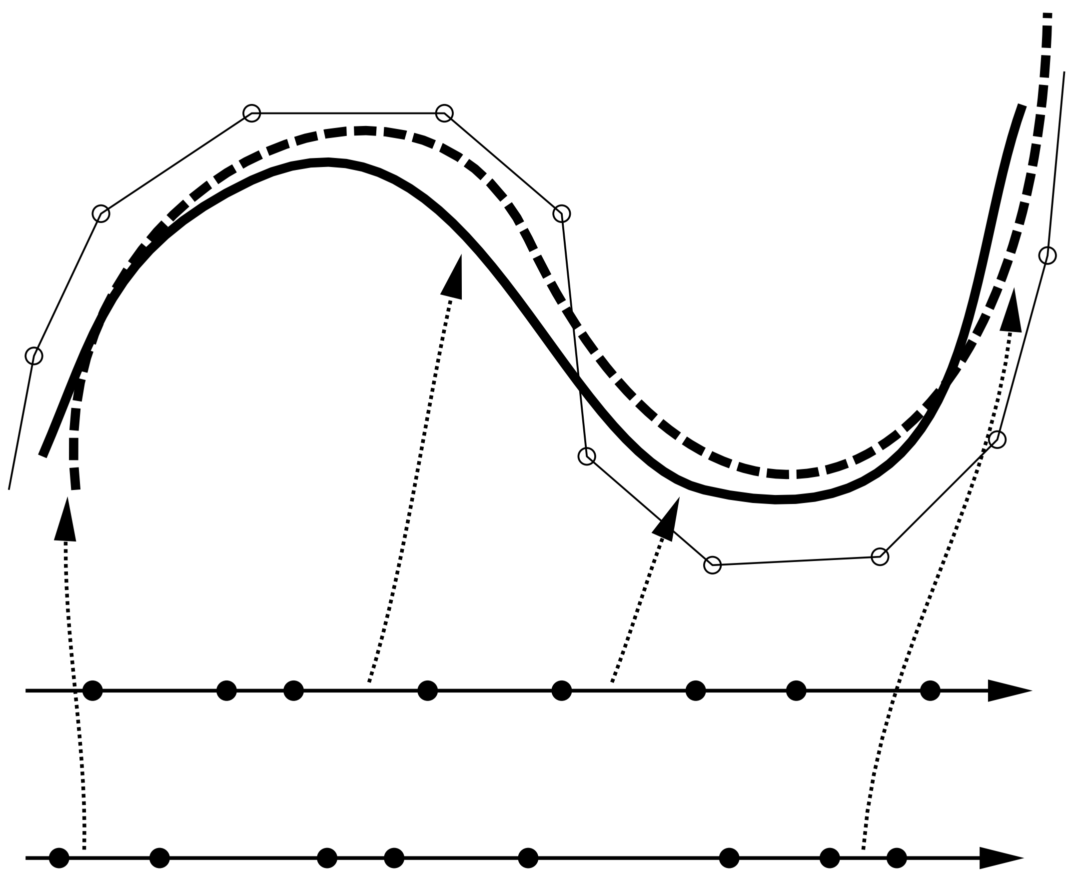
Control Point Manipulation



Weight Modification



Knot Vector Variation



Physics-Based Modeling