

Course outline

- Introduction
- Images and display techniques
 - Bases
 - Gamma correction
 - Aliasing and techniques to remedy
 - Storage

Course outline

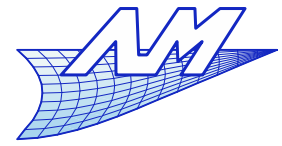
- 3D Perspective & 2D / 3D transformations
 - Go from a 3D space to a 2D display device
- Two paradigms for image synthesis
- Representation of curves and surfaces
 - Splines & co.
 - Meshes
- Realistic rendering by ray tracing
 - Concepts and theoretical bases

Course outline

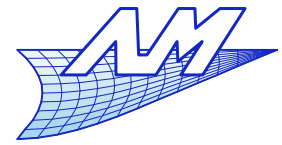
- Lighting
 - Law of reflexion, Textures
- Colorimetry
 - Color space
 - Metamerism
- Graphic pipeline and OpenGL
 - Primitives
 - Discretization (*Rasterization*)
 - Hidden faces
- Animations ?

Course outline

- Parametric surfaces
 - Coons patches
 - Tensor product surfaces
 - (Bézier triangle)
- Non parametric surfaces
 - Subdivision surfaces



Parametric Surfaces



Parametric Surfaces

- Coons patches
- Tensor-Product Surfaces
- Bézier Triangle

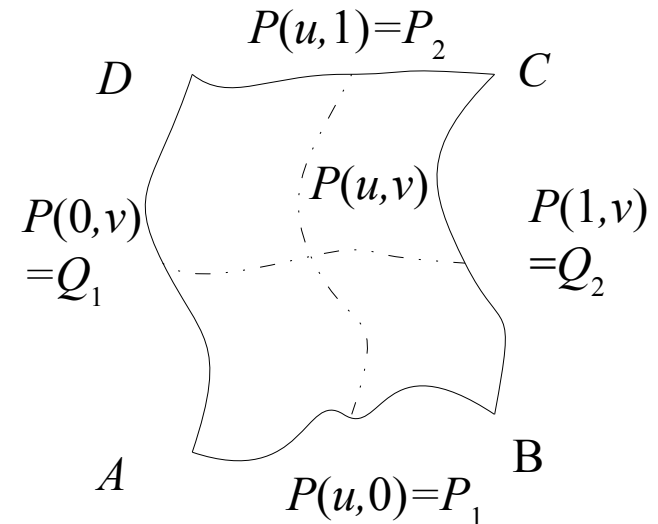
Coons patches

- Bilinear Coons patch

- Steven Anson Coons – (published in 1967 but came from research done during WWII in aeronautics)
- Let 4 parametric curves (Bézier or B-Splines or other) passing through 4 points (A, B, C, D) :

$$\begin{aligned}
 P_1(u) &= P(u, 0), & P_2(u) &= P(u, 1), \\
 Q_1(v) &= P(0, v), & Q_2(v) &= P(1, v) \text{ such as} \\
 P(0, 0) &= A & P(1, 0) &= B \\
 P(1, 1) &= C & P(0, 1) &= D
 \end{aligned}$$

- The surface $P(u, v)$ is carried by these 4 curves



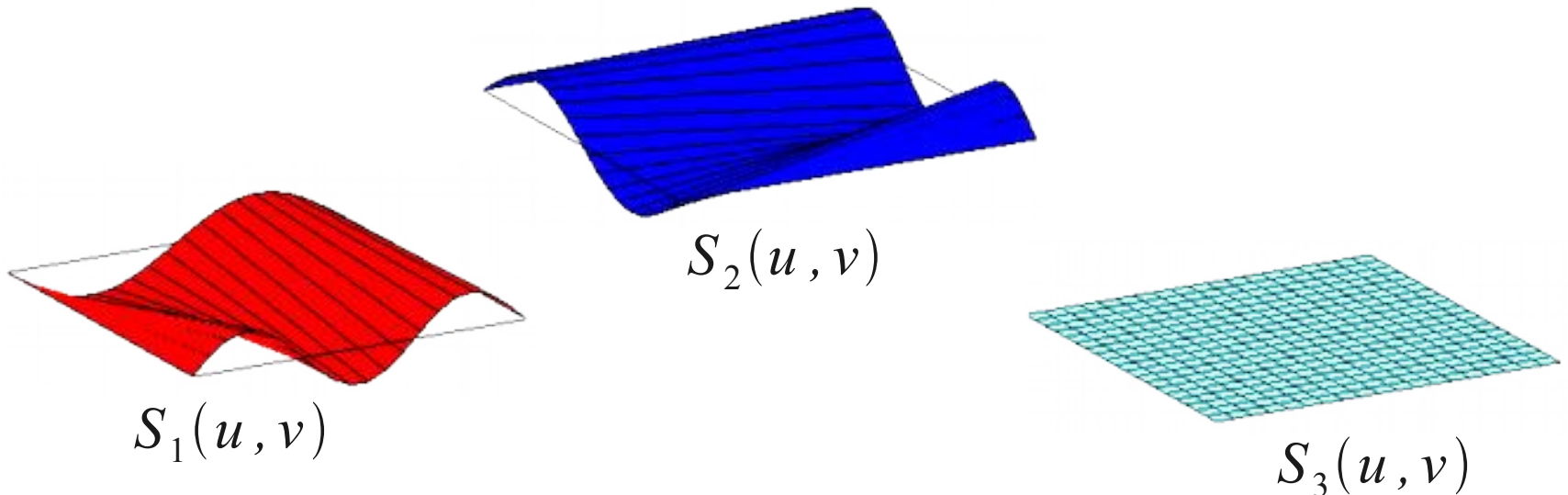
Bilinear Coons patches

- We define 3 surfaces by linear interpolation:

$$S_1(u, v) = (1-v)P(u, 0) + vP(u, 1)$$

$$S_2(u, v) = (1-u)P(0, v) + uP(1, v)$$

$$S_3(u, v) = (1-u)(1-v)P(0, 0) + u(1-v)P(0, 1) + v(1-u)P(1, 0) + uvP(1, 1)$$



Bilinear Coons patches

- The Bilinear Coons patch is defined by :

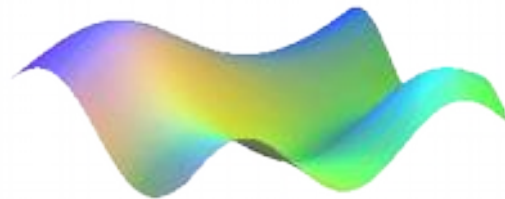
$$P(u, v) = S_1(u, v) + S_2(u, v) - S_3(u, v)$$

- Why ?

S_1 interpolates A, B, C, D

S_2 too

$S_1 + S_2$ can just interpolate A, B, C, D if we remove a term depending on A, B, C, D and linear in u and v .



$$P(u, v)$$

Bilinear Coons patches

- Systematic notation

The surface may be expressed as :

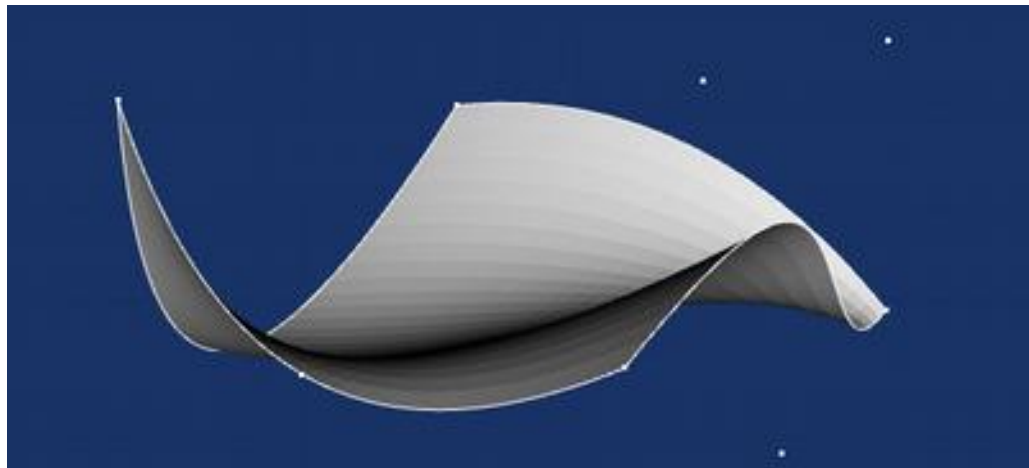
$$P(u, v) = \begin{pmatrix} 1 & F_1(u) & F_2(u) \end{pmatrix} \cdot \begin{pmatrix} 0 & P(u,0) & P(u,1) \\ P(0,v) & -P(0,0) & -P(0,1) \\ P(1,v) & -P(1,0) & -P(1,1) \end{pmatrix} \cdot \begin{pmatrix} 1 \\ F_1(v) \\ F_2(v) \end{pmatrix}$$

- In this notation, $F_1(x)=1-x$ and $F_2(x)=x$ are **blending functions**, and x is either u or v . They can be replaced by whatever function to achieve e.g. a better continuity (see later)
- In the matrix, the lower right 4-by-4 square corresponds to surface S_3 ; the upper line to S_1 and left column to S_2 .

Bilinear Coons patches

- Characteristics of a bilinear Coons patch
 - Easy to build
 - Based on any set of 4 boundary curves
 - However, there is no precise control of the shape of the surface “inside” the patch

e.g. it is impossible to impose a C^1 continuity between two neighbouring patches without constraints on the network of curves



Hermite blending

$$P(u, v) = (1 \quad F_1(u) \quad F_2(u)) \cdot \begin{pmatrix} 0 & P(u, 0) & P(u, 1) \\ P(0, v) & -P(0, 0) & -P(0, 1) \\ P(1, v) & -P(1, 0) & -P(1, 1) \end{pmatrix} \cdot \begin{pmatrix} 1 \\ F_1(v) \\ F_2(v) \end{pmatrix}$$

- It is however possible to carefully select F_1 and F_2 so that they are C^1 – continuous, like e.g. Hermite polynomials :

$$F_1(x) = 2x^3 - 3x^2 + 1 \quad F_2(x) = -2x^3 + 3x^2$$

- Then, as $\left. \frac{\partial F_1(x)}{\partial x} \right|_{x=0} = 0$, the derivatives are continuous over patch boundaries.

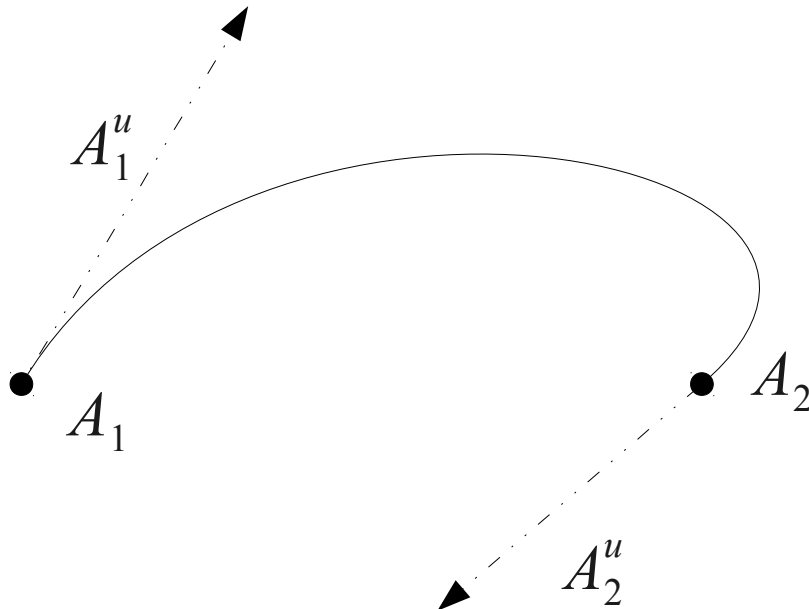
However, this is usually not sufficient : there are too many constraints on the derivatives (they vanish) and it yields a surface with flat areas around the edges.

Hermite Blending

- Hermite interpolation

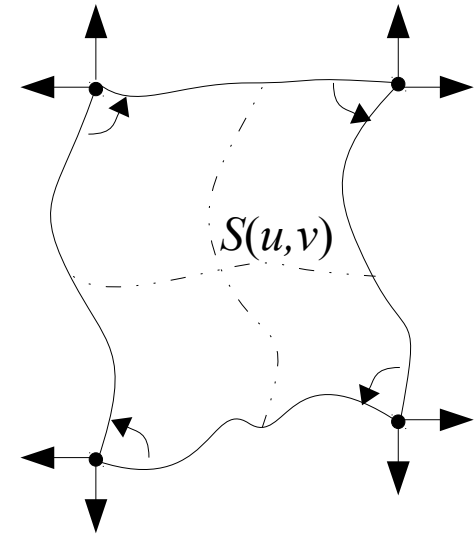
H is Hermite's matrix

$$C(u) = (A_1, A_2, A_1^u, A_2^u) \begin{pmatrix} 2 & -3 & 0 & 1 \\ -2 & 3 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 1 & -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} u^3 \\ u^2 \\ u \\ 1 \end{pmatrix}$$



Hermite Blending

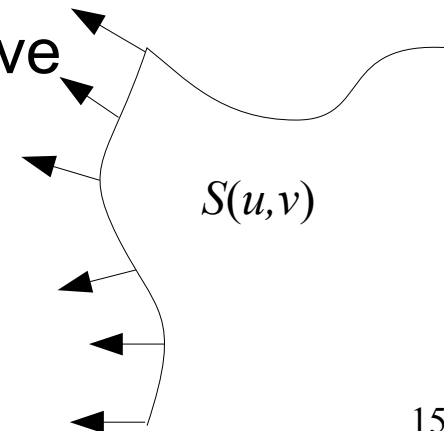
- Bicubic Hermite patch
 - 2D Hermite's interpolation
 - 4 positions at corners
 - 8 normal derivatives
 - 4 torsion vectors at corners



$$S_3(u, v) = (u^3, u^2, u, 1) \mathbf{H}^T \begin{pmatrix} A_{00} & A_{01} & A_{00}^v & A_{01}^v \\ A_{10} & A_{11} & A_{10}^v & A_{11}^v \\ A_{00}^u & A_{01}^u & A_{00}^{uv} & A_{01}^{uv} \\ A_{10}^u & A_{11}^u & A_{10}^{uv} & A_{11}^{uv} \end{pmatrix} \mathbf{H} \begin{pmatrix} v^3 \\ v^2 \\ v \\ 1 \end{pmatrix}$$

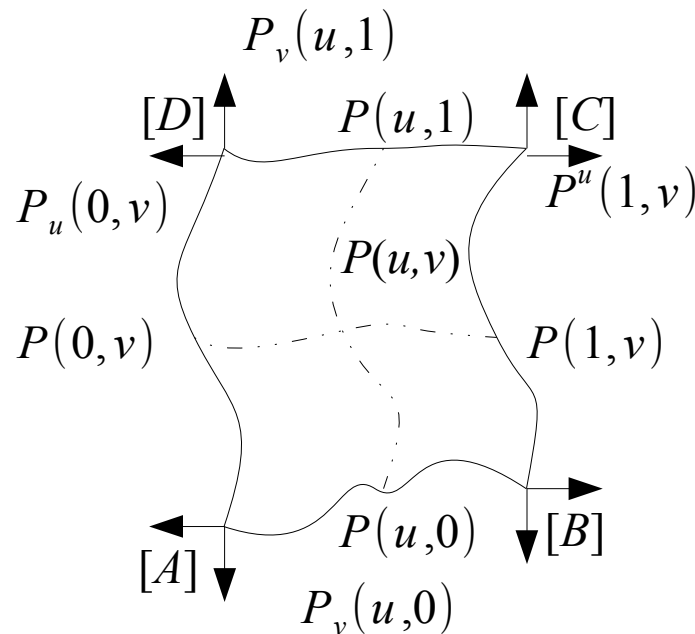
Bicubic Coons patches

- Bicubic Coons patch
 - One can build a surface that is based on any boundary curves as for the bilinear patch
 - 8 curves are necessary : 4 positional curves + 4 “curves” depicting normal derivatives
 - There are constraints between the derivative curve on one side and the positional curve on an incident side.
 - There are also constraints on the derivative curves on each corner (cross derivatives must be equal)



Bicubic Coons patches

- As for the bilinear case, we need information on boundary curves : position + derivatives, and corresponding info at the corners.



$$[X] = [P(i,j), P(i,j)_u, P(i,j)_v, P(i,j)_{uv}], (i,j) = \{0,1\}^2$$

Bicubic Coons patches

- Systematic notation for the bicubic Coons patch
 - We have now :

$$P(u, v) = (1 \quad F_1(u) \quad F_2(u) \quad F_3(u) \quad F_4(u)) \cdot \begin{pmatrix} 0 & P(u,0) & P(u,1) & P_v(u,0) & P_v(u,1) \\ P(0,v) & -P(0,0) & -P(0,1) & -P_v(0,0) & -P_v(0,1) \\ P(1,v) & -P(1,0) & -P(1,1) & -P_v(1,0) & -P_v(1,1) \\ P_u(0,v) & -P_u(0,0) & -P_u(0,1) & -P_{uv}(0,0) & -P_{uv}(0,1) \\ P_u(1,v) & -P_u(1,0) & -P_u(1,1) & -P_{uv}(1,0) & -P_{uv}(1,1) \end{pmatrix} \cdot \begin{pmatrix} 1 \\ F_1(v) \\ F_2(v) \\ F_3(v) \\ F_4(v) \end{pmatrix}$$

Bicubic Coons patches

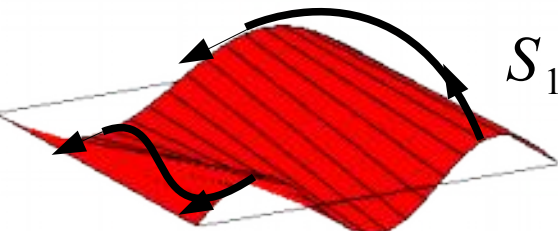
$$= (1 \quad u^3 \quad u^2 \quad u \quad 1) \cdot (\mathbf{H}^*)^T$$

$$\cdot \begin{pmatrix} 0 & P(u,0) & P(u,1) & P_v(u,0) & P_v(u,1) \\ P(0,v) & -P(0,0) & -P(0,1) & -P_v(0,0) & -P_v(0,1) \\ P(1,v) & -P(1,0) & -P(1,1) & -P_v(1,0) & -P_v(1,1) \\ P_u(0,v) & -P_u(0,0) & -P_u(0,1) & -P_{uv}(0,0) & -P_{uv}(0,1) \\ P_u(1,v) & -P_u(1,0) & -P_u(1,1) & -P_{uv}(1,0) & -P_{uv}(1,1) \end{pmatrix} \cdot \mathbf{H}^* \cdot \begin{pmatrix} 1 \\ v^3 \\ v^2 \\ v \\ 1 \end{pmatrix}$$

... with $\mathbf{H}^* = \begin{pmatrix} 1 & \dots & 0 & \dots \\ \vdots & & & \\ 0 & \mathbf{H} & & \\ \vdots & & & \end{pmatrix}$ and $\mathbf{H} = \begin{pmatrix} 2 & -3 & 0 & 1 \\ -2 & 3 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 1 & -1 & 0 & 0 \end{pmatrix}$

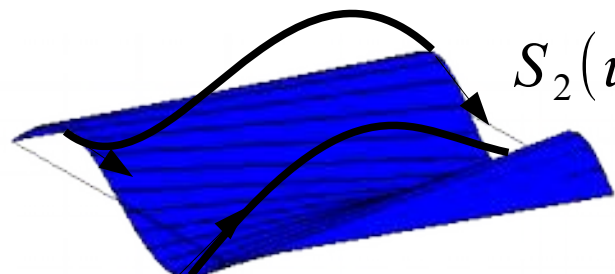
Bicubic Coons patches

- As for bilinear patches, it can be decomposed into three surfaces



$$S_1(u, v) = (P(u, 0), P(u, 1), P_v(u, 0), P_v(u, 1)) \mathbf{H} \begin{pmatrix} v^3 \\ v^2 \\ v \\ 1 \end{pmatrix}$$

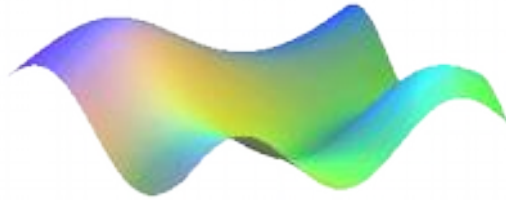
$S_1(u, v)$



$$S_2(u, v) = (P(0, v), P(1, v), P_u(0, v), P_u(1, v)) \mathbf{H} \begin{pmatrix} u^3 \\ u^2 \\ u \\ 1 \end{pmatrix}$$

$S_2(u, v)$

Bicubic Coons patches



$$S_3(u, v) = (u^3, u^2, u, 1) \cdot \mathbf{H}^T \cdot \begin{pmatrix} P(0,0) & P(0,1) & P_v(0,0) & P_v(0,1) \\ P(1,0) & P(1,1) & P_v(1,0) & P_v(1,1) \\ P_u(0,0) & P_u(0,1) & P_{uv}(0,0) & P_{uv}(0,1) \\ P_u(1,0) & P_u(1,1) & P_{uv}(1,0) & P_{uv}(1,1) \end{pmatrix} \cdot \mathbf{H} \cdot \begin{pmatrix} v^3 \\ v^2 \\ v \\ 1 \end{pmatrix}$$

$$P(u, v) = S_1(u, v) + S_2(u, v) - S_3(u, v)$$

Bicubic Coons patches

- Bicubic Coons patch

$$P(u, v) = S_1(u, v) + S_2(u, v) - S_3(u, v)$$

$$S_3(u, v) = (u^3, u^2, u, 1) \mathbf{H}^T \begin{pmatrix} A_{00} & A_{01} & A_{00}^v & A_{01}^v \\ A_{10} & A_{11} & A_{10}^v & A_{11}^v \\ A_{00}^u & A_{01}^u & A_{00}^{uv} & A_{01}^{uv} \\ A_{10}^u & A_{11}^u & A_{10}^{uv} & A_{11}^{uv} \end{pmatrix} \mathbf{H} \begin{pmatrix} v^3 \\ v^2 \\ v \\ 1 \end{pmatrix}$$

The terms of the matrix are computed with the help of border curves and verify the following conditions :

$$\begin{array}{cccc} A_{00} = P(0,0) & A_{01} = P(0,1) & A_{00}^v = P_v(0,0) & A_{01}^v = P_v(0,1) \\ A_{10} = P(1,0) & A_{11} = P(1,1) & A_{10}^v = P_v(1,0) & A_{11}^v = P_v(1,1) \\ A_{00}^u = P_u(0,0) & A_{01}^u = P_u(0,1) & A_{00}^{uv} = P_{uv}(0,0) & A_{01}^{uv} = P_{uv}(0,1) \\ A_{10}^u = P_u(1,0) & A_{11}^u = P_u(1,1) & A_{10}^{uv} = P_{uv}(1,0) & A_{11}^{uv} = P_{uv}(1,1) \end{array}$$

Bicubic Ferguson patch

- Ferguson patch

$$P(u, v) = S_1(u, v) + S_2(u, v) - S_3(u, v)$$

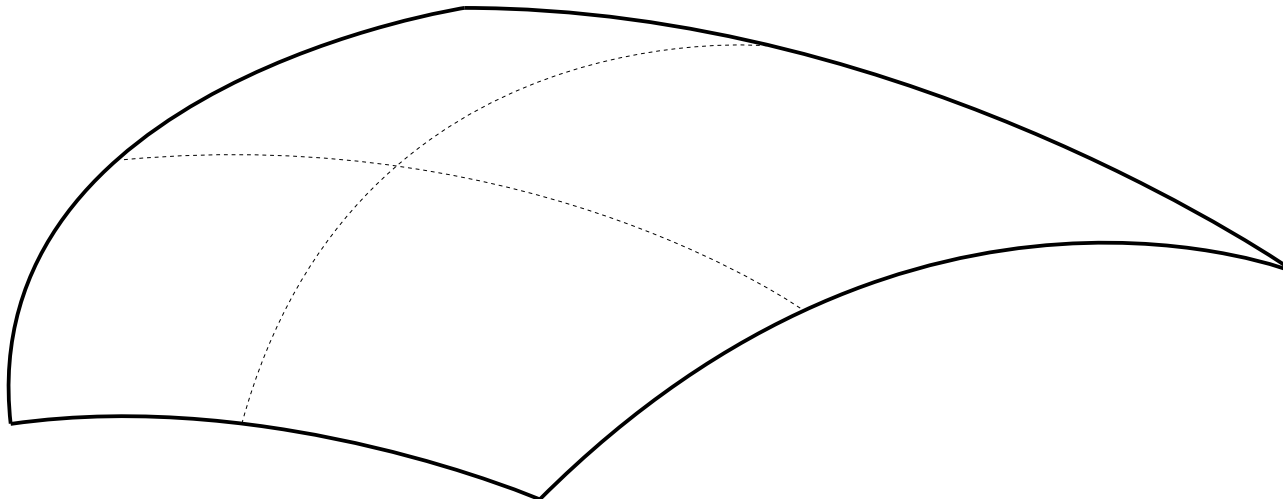
$$S_3(u, v) = (u^3, u^2, u, 1) \mathbf{H}^T \begin{pmatrix} A_{00} & A_{01} & A_{00}^v & A_{01}^v \\ A_{10} & A_{11} & A_{10}^v & A_{11}^v \\ A_{00}^u & A_{01}^u & A_{00}^{uv} & A_{01}^{uv} \\ A_{10}^u & A_{11}^u & A_{10}^{uv} & A_{11}^{uv} \end{pmatrix} \mathbf{H} \begin{pmatrix} v^3 \\ v^2 \\ v \\ 1 \end{pmatrix}$$

The terms of the matrix are computed with the help of border curves and verify the following conditions :

$A_{00} = P(0,0)$	$A_{01} = P(0,1)$	$A_{00}^v = P_v(0,0)$	$A_{01}^v = P_v(0,1)$
$A_{10} = P(1,0)$	$A_{11} = P(1,1)$	$A_{10}^v = P_v(1,0)$	$A_{11}^v = P_v(1,1)$
$A_{00}^u = P_u(0,0)$	$A_{01}^u = P_u(0,1)$	A_{00}^{uv} such that $\frac{\partial^2 P}{\partial u \partial v}(0,0) = 0$	A_{01}^{uv} such that $\frac{\partial^2 P}{\partial u \partial v}(0,1) = 0$
$A_{10}^u = P_u(1,0)$	$A_{11}^u = P_u(1,1)$	A_{10}^{uv} such that $\frac{\partial^2 P}{\partial u \partial v}(1,0) = 0$	A_{11}^{uv} such that $\frac{\partial^2 P}{\partial u \partial v}(1,1) = 0$

Bicubic Coons patches

- We can impose the position and the normal tangent along the boundaries
- Remain the problem of the continuity at every corner
 - We usually impose that cross derivatives vanish \rightarrow Ferguson patch
 - Other constraints may be found in the literature...



Tensor product surfaces

- Parametric surfaces as a polar form

$$S(u, v) = \sum_k N_k(u, v) P_k$$

- One shape function per control point
- « Tensor product » surface if N_k is separable :
 - Combination of elementary curves/shape functions independently defined on u and v .

$$S(u, v) = \sum_i \sum_j G_i(u) H_j(v) P_{ij}$$

Usually built upon Bézier and B-Splines curves/SFs

- The “unique” shape function is $N_k(u, v) = G_i(u) H_j(v)$

B-Spline surfaces

- B-Splines surfaces uses 1D B-Spline shape fns.

- Definition as tensor product :

$$S(u, v) = \sum_{i=0}^n \sum_{j=0}^m N_i^p(u) N_j^q(v) P_{ij}$$

- Every variable u and v has a degree (p and q) and a nodal sequence U and V :

$$U = \left\{ \underbrace{0, \dots, 0}_{p+1}, u_{p+1}, \dots, u_{r-p-1}, \underbrace{1, \dots, 1}_{p+1} \right\} \quad (r+1 \text{ nodes})$$

$$V = \left\{ \underbrace{0, \dots, 0}_{q+1}, v_{q+1}, \dots, v_{s-q-1}, \underbrace{1, \dots, 1}_{q+1} \right\} \quad (s+1 \text{ nodes})$$

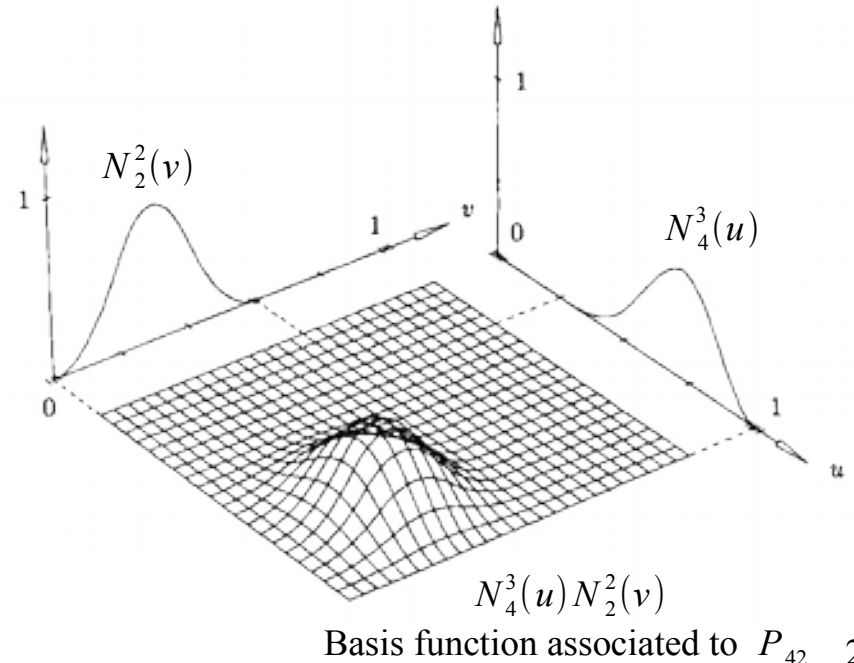
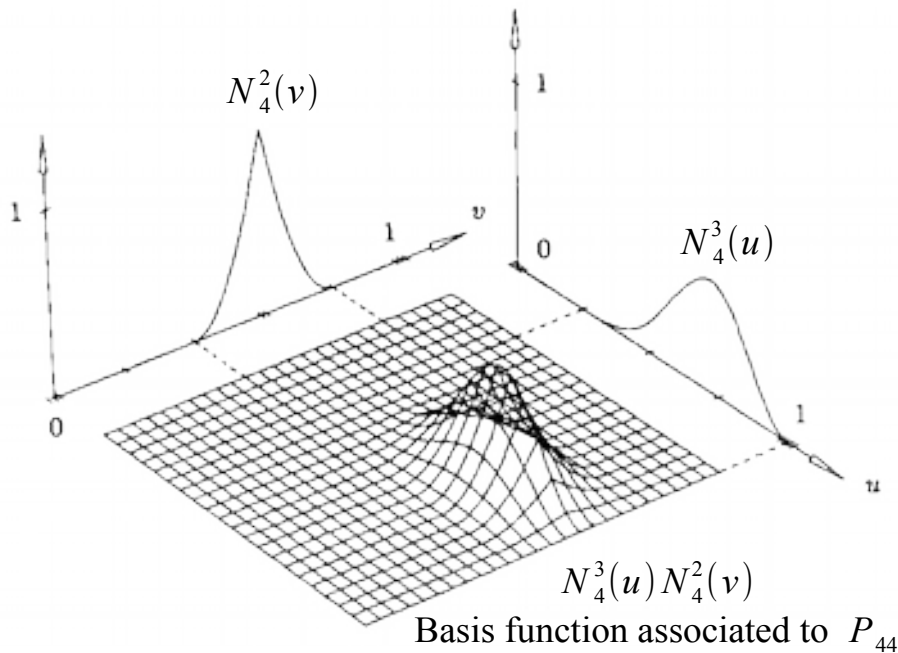
- The control points forms a regular net P_{ij} ($n+1$ times $m+1$) values.
- We have the following relations : $r = n + p + 1$ $s = m + q + 1$

B-Spline surfaces

- Example of basis functions

$$U = \{0, 0, 0, 0, 1/4, 1/2, 3/4, 1, 1, 1, 1\} \quad p=3$$

$$V = \{0, 0, 0, 1/5, 2/5, 3/5, 3/5, 4/5, 1, 1, 1\} \quad q=2$$



B-Spline surfaces

- Properties of surface basis functions
 - Extrema
If $p > 0$ and $q > 0$, $N_i^p(u) N_j^q(v)$ has a unique maximum.
 - Continuity
Inside rectangles formed by the nodes u_i and v_j , the SF are infinitely differentiable.
At a node u_i (resp. v_j), $N_i^p(u) N_j^q(v)$ is $(p-k)$ (resp. $(q-k)$) times differentiable, k being the node multiplicity u_i (resp. v_j)
The continuity with respect to u (resp. v) depends solely on the nodal sequence U (resp. V).

B-Spline surfaces

- Properties of surface basis functions

- A consequence of properties of the 1D shape functions

- Non-negativity

$$N_i^p(u) N_j^q(v) \geq 0 \quad \forall i, j, p, q, u, v$$

- Partition of unity

$$\sum_{i=0}^n \sum_{j=0}^m N_i^p(u) N_j^q(v) = 1 \quad \forall (u, v) \in [u_{min}, u_{max}] \times [v_{min}, v_{max}]$$

- Compact support

$$N_i^p(u) N_j^q(v) = 0 \quad \text{outside } (u, v) \in [u_i, u_{i+p+1}] \times [v_j, v_{j+q+1}]$$

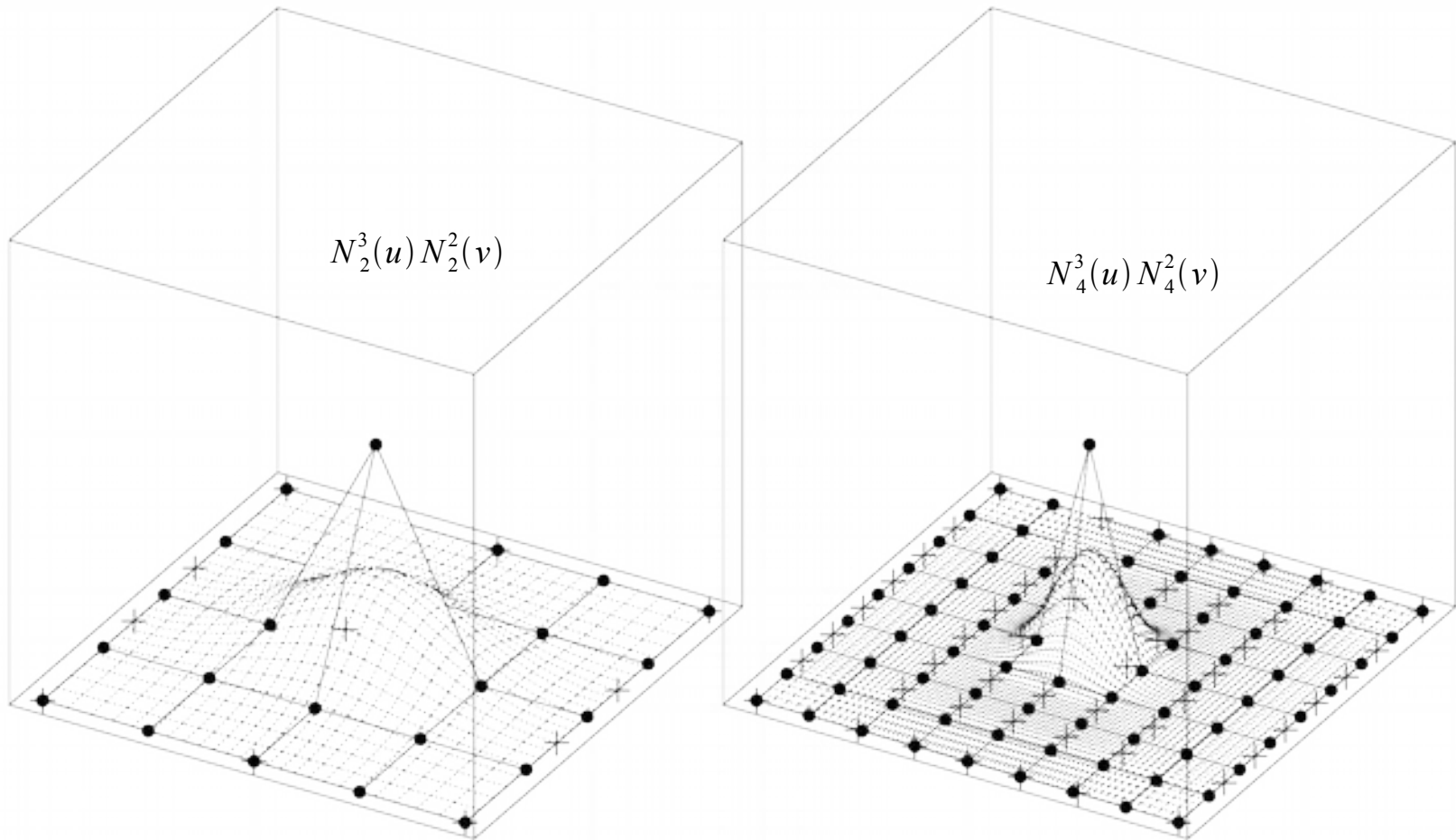
There are at most $(p+1)(q+1)$ non zero SF in a given interval $[u_{i_0}, u_{i_0+1}] \times [v_{j_0}, v_{j_0+1}]$.

In particular $N_i^p(u) N_j^q(v) \neq 0 \quad i_0 - p \leq i \leq i_0 \quad j_0 - q \leq j \leq j_0$

B-Spline surfaces

- Properties of surface basis functions
 - Extrema
If $p > 0$ and $q > 0$, $N_i^p(u) N_j^q(v)$ has a unique maximum.
 - Continuity
Inside rectangles formed by the nodes u_i and v_j , the SF are infinitely differentiable.
At a node u_i (resp. v_j), $N_i^p(u) N_j^q(v)$ is $(p-k)$ (resp. $(q-k)$) times differentiable, k being the node multiplicity u_i (resp. v_j)
The continuity with respect to u (resp. v) depends solely on the nodal sequence U (resp. V).

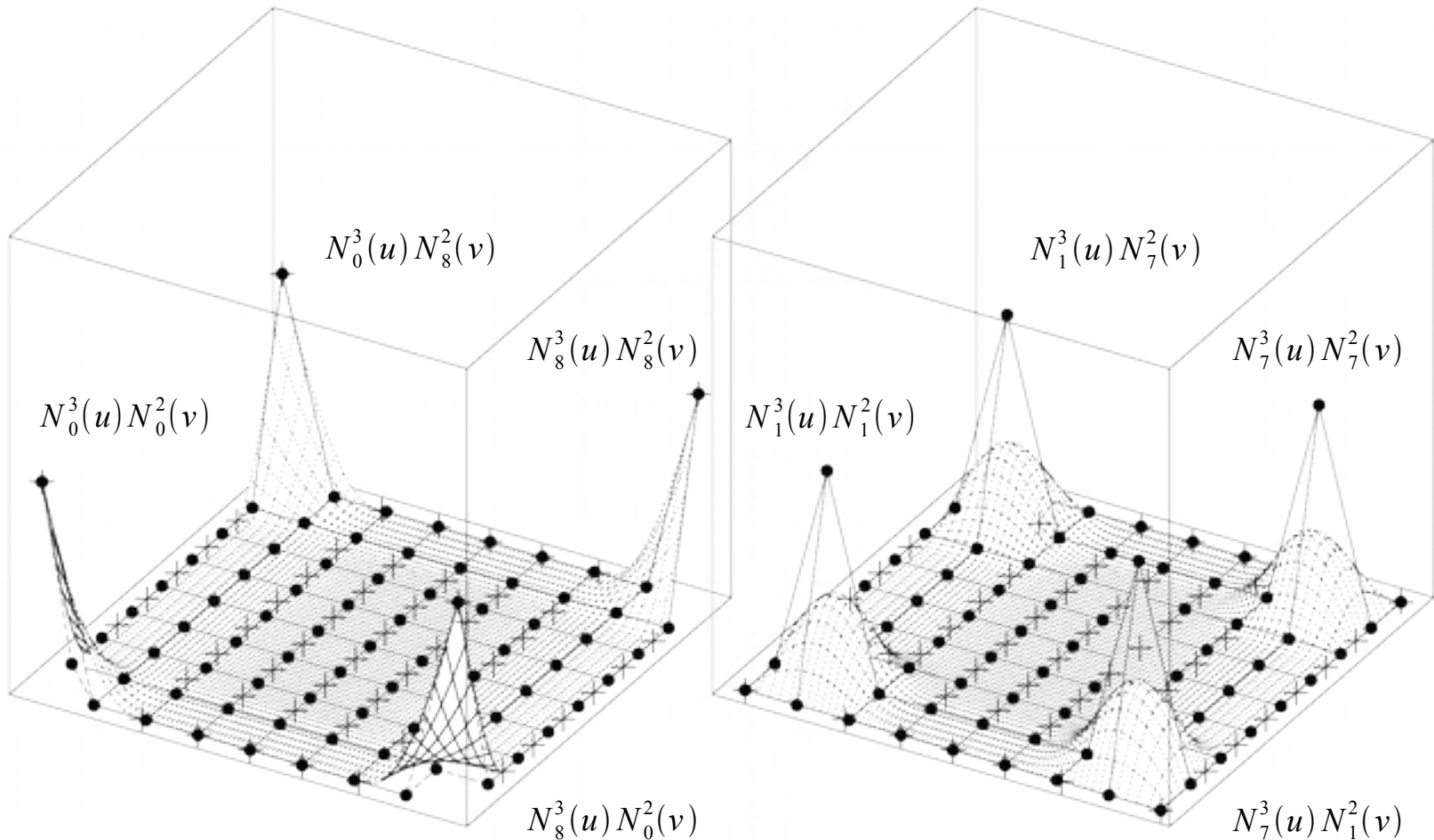
B-Spline surfaces



$$U = \{0, 0, 0, 0, 1, 2, 2, 2, 2\} \quad p=3 \quad U = \{0, 0, 0, 0, 1, 2, 3, 4, 5, 6, 6, 6, 6\}$$

$$V = \{0, 0, 0, 1, 2, 3, 3, 3\} \quad q=2 \quad V = \{0, 0, 0, 1, 2, 3, 4, 5, 6, 7, 7, 7\}$$

B-Spline surfaces



$$U = \{0, 0, 0, 0, 1, 2, 3, 4, 5, 6, 6, 6, 6\} \quad p = 3$$

$$V = \{0, 0, 0, 1, 2, 3, 4, 5, 6, 7, 7, 7\} \quad q = 2$$

B-Spline surfaces

- Computation of a point on the surface

- 1 – Find the nodal interval in which u is located

$$u \in [u_i, u_{i+1}[$$

- 2 – Compute the non vanishing 1D shape functions

$$N_{i-p}^p(u), \dots, N_i^p(u)$$

- 3 – Find the nodal interval in which v is located

$$v \in [v_j, v_{j+1}[$$

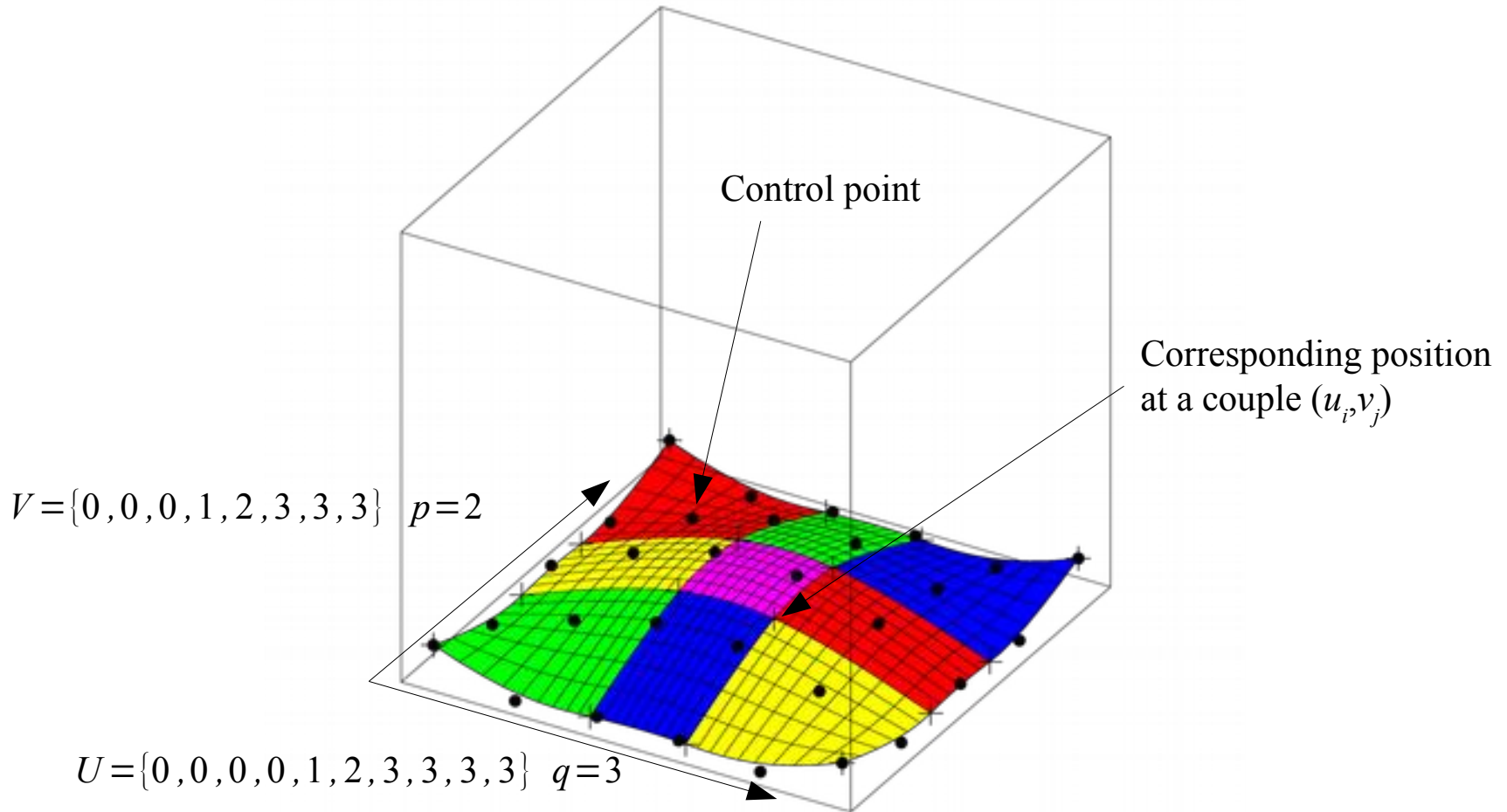
- 4 – Compute the non vanishing 1D shape functions

$$N_{j-q}^q(v), \dots, N_j^q(v)$$

- 5 – Multiply the SFs with the adequate control points

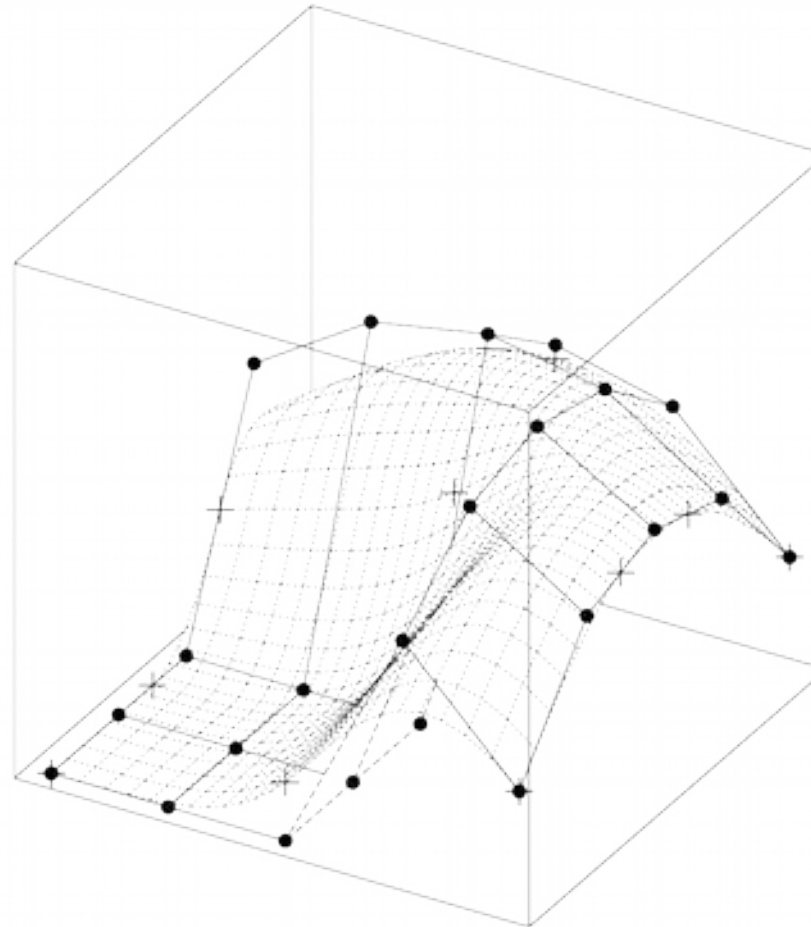
$$S(u, v) = \sum_k \sum_l N_k^p(u) P_{kl} N_l^q(v) \quad i-p \leq k \leq i \quad , \quad j-q \leq l \leq j$$

B-Spline surfaces



- Each coloured square has an independent polynomial expression

B-Spline surfaces

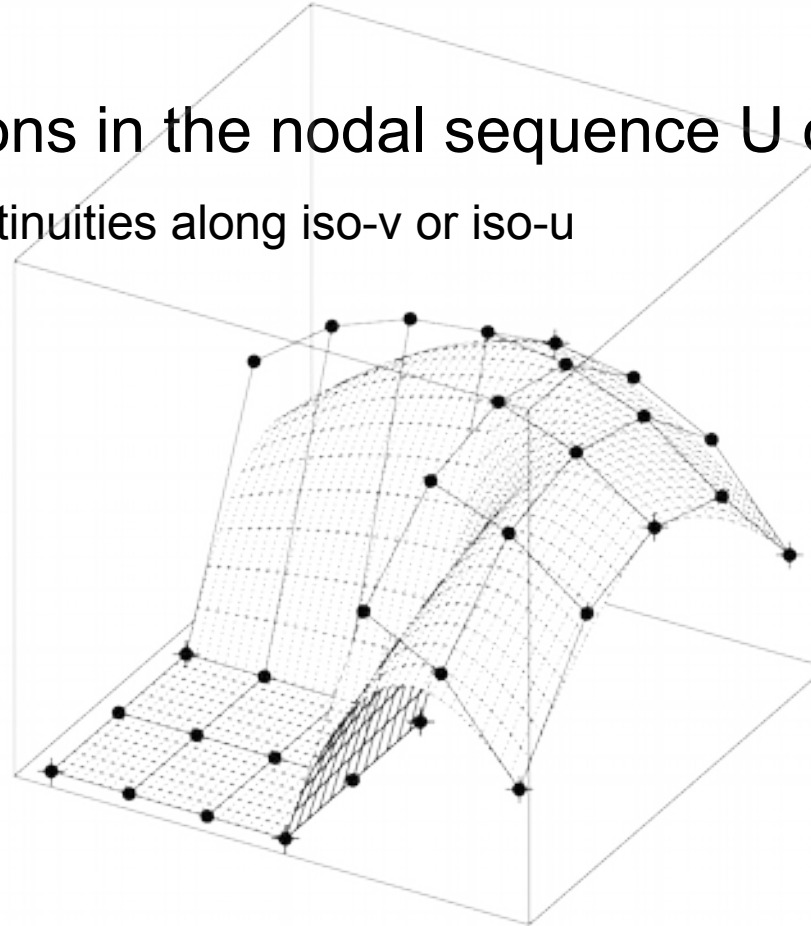


$$U = \{0, 0, 0, 0, 1, 2, 2, 2, 2\} \quad p = 3$$

$$V = \{0, 0, 0, 1, 2, 3, 3, 3\} \quad q = 2$$

B-Spline surfaces

- Repetitions in the nodal sequence U or V
 - Discontinuities along iso-v or iso-u



$$U = \{0, 0, 0, 0, 2, 2, 2, 4, 4, 4, 4\} \quad p=3$$

$$V = \{0, 0, 0, 1.5, 1.5, 3, 3, 3\} \quad q=2$$

B-Spline surfaces

- Properties of the B-Spline surface
 - Interpolate the 4 corners if the nodal sequences are of the form

$$U = \left\{ \underbrace{0, \dots, 0}_{p+1}, u_{p+1}, \dots, u_{r-p-1}, \underbrace{1, \dots, 1}_{p+1} \right\}$$

$$V = \left\{ \underbrace{0, \dots, 0}_{q+1}, v_{q+1}, \dots, v_{s-q-1}, \underbrace{1, \dots, 1}_{q+1} \right\}$$

- If the nodal sequences correspond to Bézier curves :

$$U = \left\{ \underbrace{0, \dots, 0}_{p+1}, \underbrace{1, \dots, 1}_{p+1} \right\} \quad V = \left\{ \underbrace{0, \dots, 0}_{q+1}, \underbrace{1, \dots, 1}_{q+1} \right\}$$

then the surface is called a Bézier patch.

B-Spline surfaces

- Properties of the B-Spline surface
 - The surface has the property of affine invariance (invariance by translation in particular)
 - The convex hull of the control points contains the surface.
 - In every interval $(u, v) \in [u_{i_0}, u_{i_0+1}[\times [v_{j_0}, v_{j_0+1}[$, the portion of the surface is in the convex hull of the control points P_{ij} , (i, j) such that $i_0 - p \leq i \leq i_0$ $j_0 - q \leq j \leq j_0$
 - Control points may have a local control
 - There is **no** variation diminishing property (on the contrary to B-Spline/Bézier curves)

B-Spline surfaces

- Isoparametrics

- Computation of isoparametrics is easy :

Set $u=u_0$

$$\begin{aligned}
 C_{u_0}(v) = S(u_0, v) &= \sum_{i=0}^n \sum_{j=0}^m N_i^p(u_0) N_j^q(v) P_{ij} \\
 &= \sum_{j=0}^m N_j^q(v) \sum_{i=0}^n N_i^p(u_0) P_{ij} = \sum_{j=0}^m N_j^q(v) Q_j(u_0)
 \end{aligned}$$

$$\text{with } Q_j(u_0) = \sum_{i=0}^n N_i^p(u_0) P_{ij}$$

- same with $v=v_0$

$$C_{v_0}(u) = S(u, v_0) = \sum_{i=0}^n N_i^p(u) Q_i(v_0)$$

$$\text{with } Q_i(v_0) = \sum_{j=0}^m N_j^q(v_0) P_{ij} \quad 38$$

B-Spline surfaces

- Derivatives of a B-Spline surface

- We want to compute $\frac{\partial^{k+l}}{\partial u^k \partial v^l} S(u, v)$

- Differentiation of basis functions :

$$\begin{aligned} \frac{\partial^{k+l}}{\partial u^k \partial v^l} S(u, v) &= \sum_{i=0}^n \sum_{j=0}^m \frac{\partial^{k+l}}{\partial u^k \partial v^l} N_i^p(u) N_j^q(v) P_{ij} \\ &= \sum_{i=0}^n \sum_{j=0}^m \frac{\partial^k}{\partial u^k} N_i^p(u) \frac{\partial^l}{\partial v^l} N_j^q(v) P_{ij} \\ &= \sum_{i=0}^n \sum_{j=0}^m N_i^{p^{(k)}}(u) N_j^{q^{(l)}}(v) P_{ij} \end{aligned}$$

B-Spline surfaces

- Derivatives expressed as B-Spline surfaces
 - Let's derive formally S with respect to u :

$$\begin{aligned}\frac{\partial S(u, v)}{\partial u} &= \sum_{j=0}^m N_j^q(v) \left(\frac{\partial}{\partial u} \sum_{i=0}^n N_i^p(u) P_{ij} \right) \\ &= \sum_{j=0}^m N_j^q(v) \left(\frac{\partial}{\partial u} C_j(u) \right)\end{aligned}$$

$$\text{with } C_j(u) = \sum_{i=0}^n N_i^p(u) P_{ij}$$

- We want to apply equations seen for curves :

B-Spline surfaces

- Derivatives of the curve $C_j(u) = \sum_{i=0}^n N_i^p(u) P_{ij}$

$$U = \left\{ \underbrace{u_0, \dots, u_p}_{p+1 \text{ times}}, \dots, \underbrace{u_{m-p}, \dots, u_m}_{p+1 \text{ times}} \right\}$$

$$U' = \left\{ \underbrace{u'_0, \dots, u'_{p-1}}_{p \text{ times}}, \dots, \underbrace{u'_{m-p-1}, \dots, u'_{m-2}}_{p \text{ times}} \right\} \text{ with } u'_i = u_{i+1}$$

$$P'(u) = \sum_{i=0}^{n-1} N_i^{p-1}(u) Q_i \quad \text{with } N_i^{p-1} \text{ defined on } U'$$

$$Q_i = p \frac{P_{i+1j} - P_{ij}}{u_{i+d+1} - u_{i+1}}$$

B-Spline surfaces

- We obtain :

$$\frac{\partial S(u, v)}{\partial u} = \sum_{i=0}^{n-1} \sum_{j=0}^m N_i^{p-1}(u) N_j^q(v) P_{ij}^{(1,0)}$$

$$\text{with } P_{ij}^{(1,0)} = p \frac{P_{i+1j} - P_{ij}}{u_{i+p+1} - u_{i+1}}$$

$$U = \left\{ \underbrace{u_0, \dots, u_p}_{p+1 \text{ times}}, \dots, \underbrace{u_{m-p}, \dots, u_m}_{p+1 \text{ times}} \right\}$$

$$U^{(1)} = \left\{ \underbrace{u_0^{(1)}, \dots, u_{p-1}^{(1)}}_{p \text{ times}}, \dots, \underbrace{u_{m-p-1}^{(1)}, \dots, u_{m-2}^{(1)}}_{p \text{ times}} \right\} \text{ with } u_i^{(1)} = u_{i+1}, 0 \leq i \leq m-2$$

B-Spline surfaces

- Let's derive formally S with respect to v :

$$\frac{\partial S(u, v)}{\partial v} = \sum_{i=0}^n \sum_{j=0}^{m-1} N_i^p(u) N_j^{q-1}(v) P_{ij}^{(0,1)}$$

$$\text{with } P_{ij}^{(0,1)} = q \frac{P_{ij+1} - P_{ij}}{v_{j+q+1} - v_{j+1}}$$

$$V = \left\{ \underbrace{v_0, \dots, v_q}_{q+1 \text{ times}}, \dots, \underbrace{v_{n-q}, \dots, v_n}_{q+1 \text{ times}} \right\}$$

$$V^{(1)} = \left\{ \underbrace{v_0^{(1)}, \dots, v_{q-1}^{(1)}}_{q \text{ times}}, \dots, \underbrace{v_{n-q-1}^{(1)}, \dots, v_{n-2}^{(1)}}_{q \text{ times}} \right\} \text{ with } v_j^{(1)} = v_{j+1}, 0 \leq j \leq n-2$$

B-Spline surfaces

- Let's derive formally S with respect to u , then v :

$$\frac{\partial^2 S(u, v)}{\partial u \partial v} = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} N_i^{p-1}(u) N_j^{q-1}(v) P_{ij}^{(1,1)}$$

$$\text{with } P_{ij}^{(1,1)} = q \frac{P_{ij+1}^{(1,0)} - P_{ij}^{(1,0)}}{v_{j+q+1} - v_{j+1}}$$

$$U^{(1)} = \left\{ \underbrace{u_0^{(1)}, \dots, u_{p-1}^{(1)}}_{p \text{ times}}, \dots, \underbrace{u_{m-p-1}^{(1)}, \dots, u_{m-2}^{(1)}}_{p \text{ times}} \right\} \text{ with } u_i^{(1)} = u_{i+1}, 0 \leq i \leq m-2$$

$$V^{(1)} = \left\{ \underbrace{v_0^{(1)}, \dots, v_{q-1}^{(1)}}_{q \text{ times}}, \dots, \underbrace{v_{n-q-1}^{(1)}, \dots, v_{n-2}^{(1)}}_{q \text{ times}} \right\} \text{ with } v_j^{(1)} = v_{j+1}, 0 \leq j \leq n-2$$

B-Spline surfaces

- General case :

$$\frac{\partial^{k+l} S(u, v)}{\partial u^k \partial v^l} = \sum_{i=0}^{n-k} \sum_{j=0}^{m-l} N_i^{p-k}(u) N_j^{q-l}(v) P_{ij}^{(k,l)}$$

$$\text{with } P_{ij}^{(k,l)} = (q-l+1) \frac{P_{ij+1}^{(k,l-1)} - P_{ij}^{(k,l-1)}}{v_{j+q+1} - v_{j+l}}$$

$$U^{(k)} = \left\{ \underbrace{u_0^{(k)}, \dots, u_{p-k}^{(k)}}_{p+1-k \text{ times}}, \dots, \underbrace{u_{m-p-k}^{(k)}, \dots, u_{m-2k}^{(k)}}_{p+1-k \text{ times}} \right\} \text{ with } u_i^{(k)} = u_{i+k}, 0 \leq i \leq m-2k$$

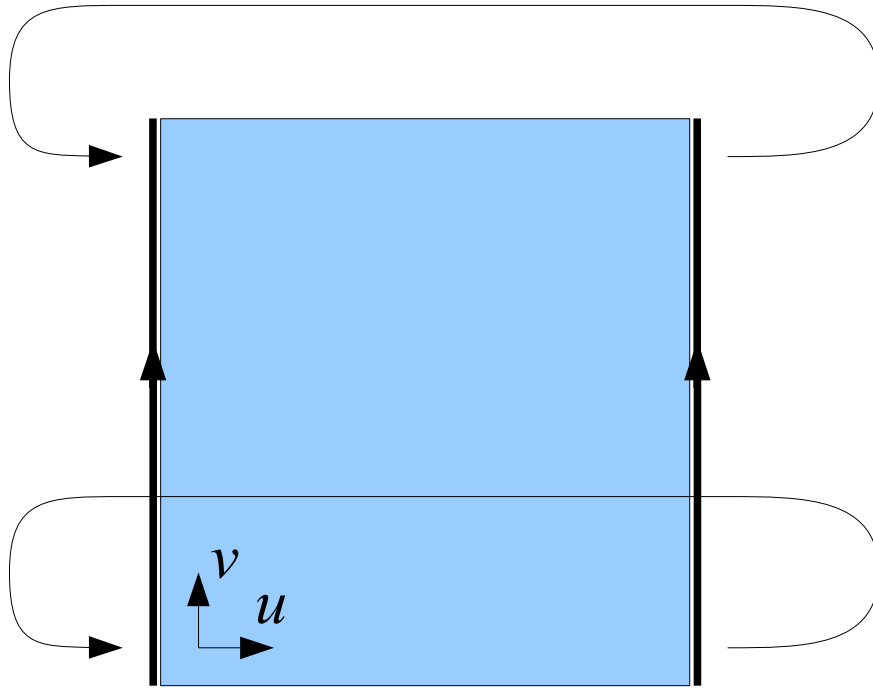
$$V^{(l)} = \left\{ \underbrace{v_0^{(l)}, \dots, v_{q-l}^{(l)}}_{q+1-l \text{ times}}, \dots, \underbrace{v_{n-q-l}^{(l)}, \dots, v_{n-2l}^{(l)}}_{q+1-l \text{ times}} \right\} \text{ with } v_j^{(l)} = v_{j+l}, 0 \leq j \leq n-2l$$

- The derivative vector of a B-Spline surface also is a B-Spline surface...

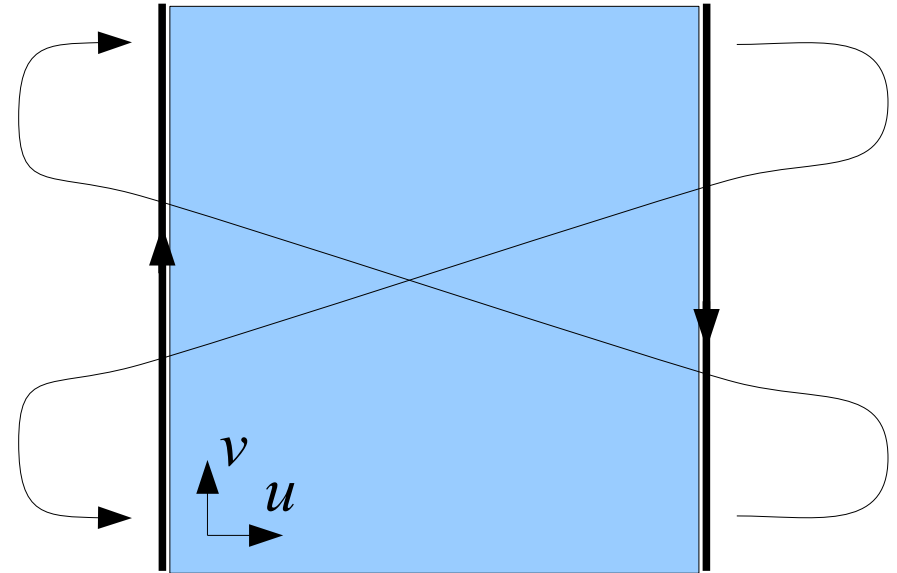
B-Spline surfaces

- Periodic surfaces
 - Like for the curves, possibility to “close” a B-Spline surface by transforming the nodal sequence
 - According to one parameter (u or v)
Cylindrical surfaces
 - A single periodic nodal sequence
 - Control points on both sides of the seam are doubled
 - According to both parameters (u and v)
Toroidal surfaces
 - Two periodic nodal sequences
 - Some control points are repeated 4 times !

B-Spline surfaces

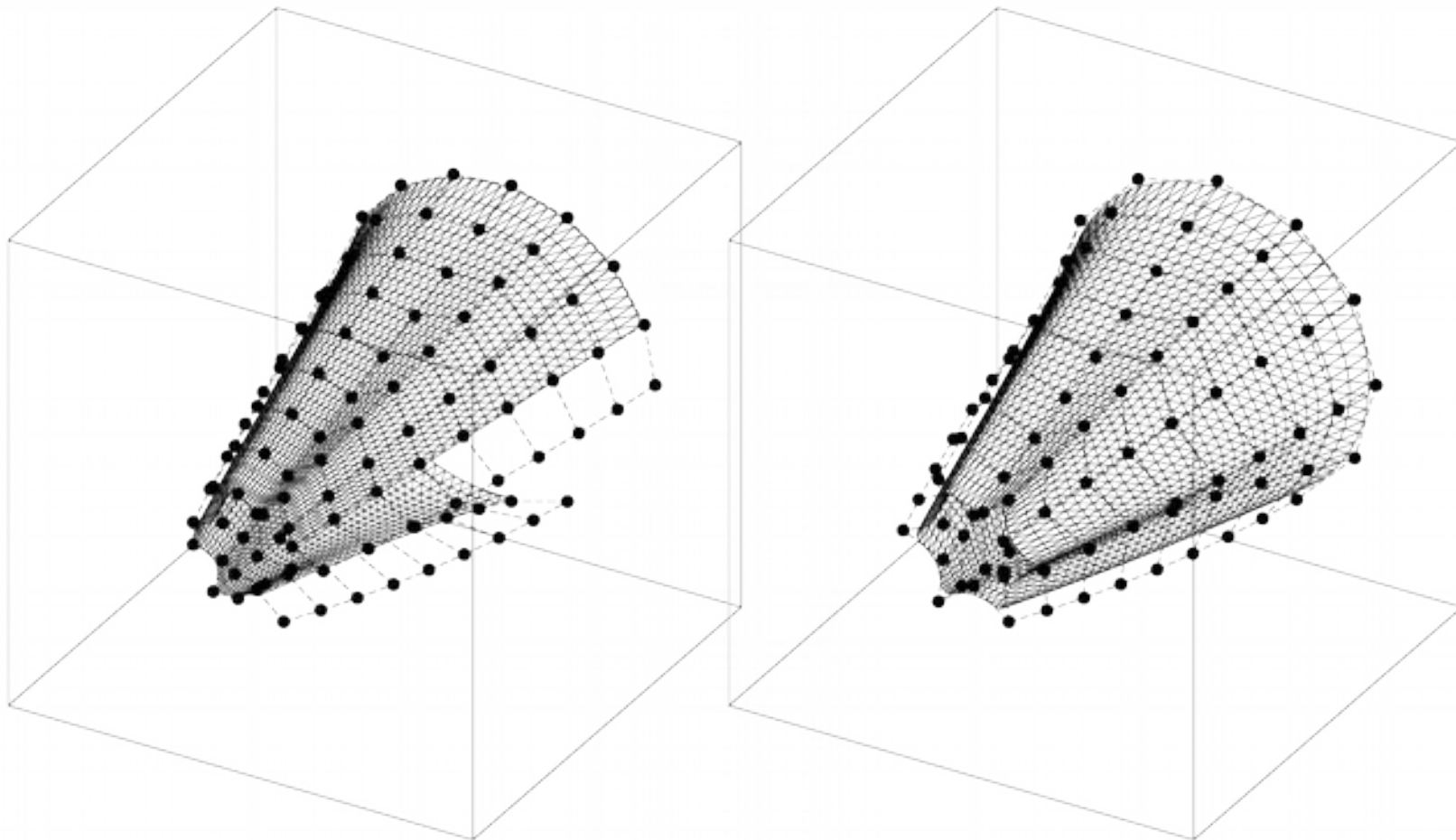


Pipe



Moebius's ribbon

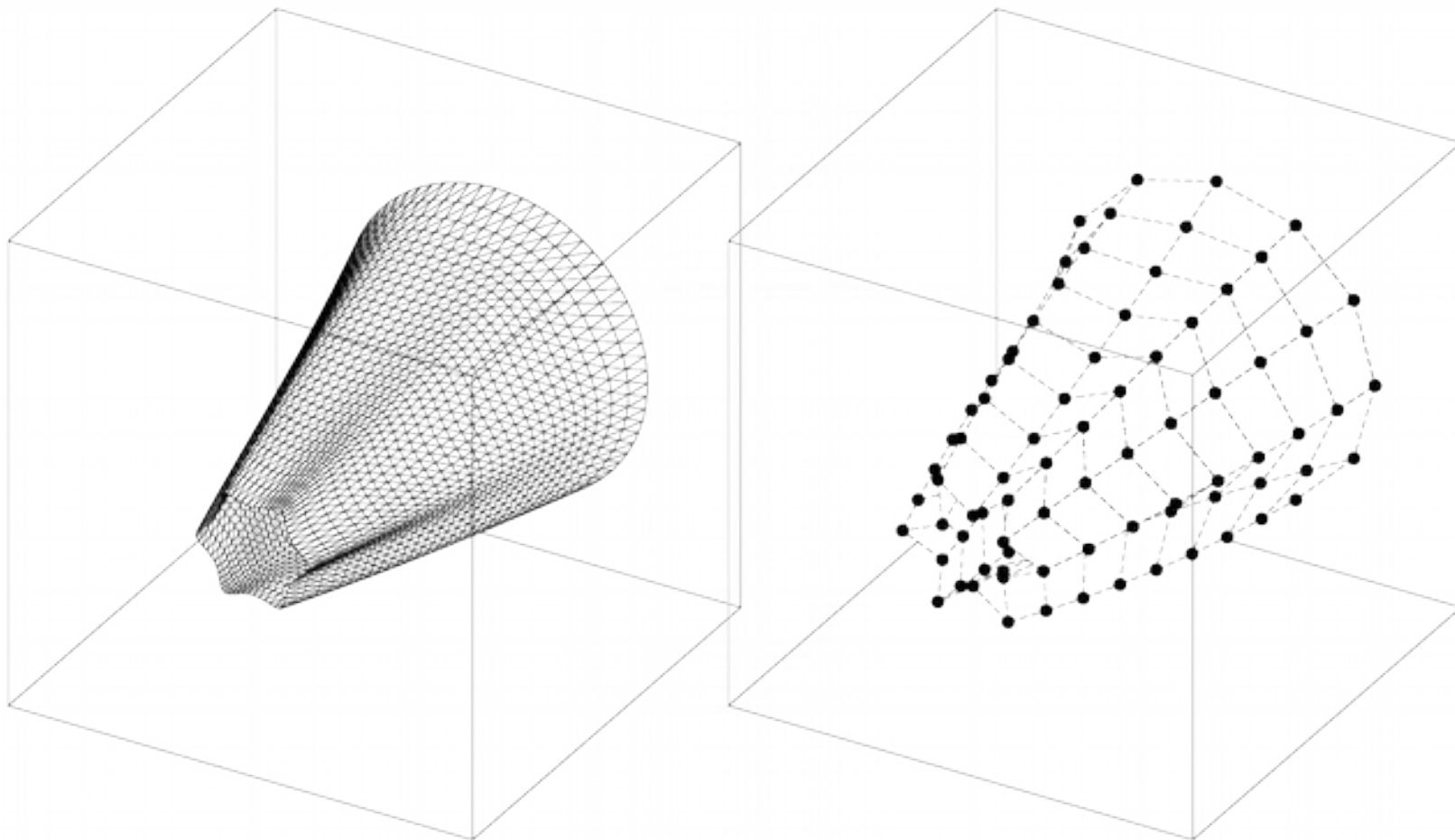
B-Spline surfaces



$$U = \{-3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\} \quad p=3$$

$$V = \{0, 0, 0, 1, 2, 3, 4, 5, 6, 7, 7, 7\} \quad p=2$$

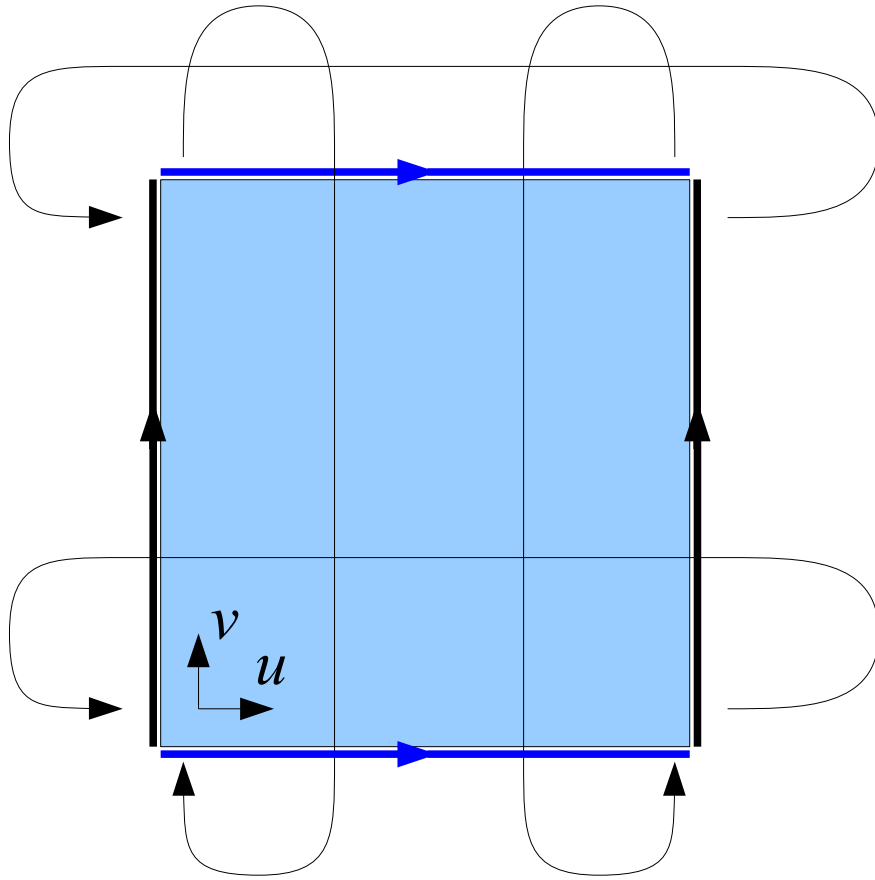
B-Spline surfaces



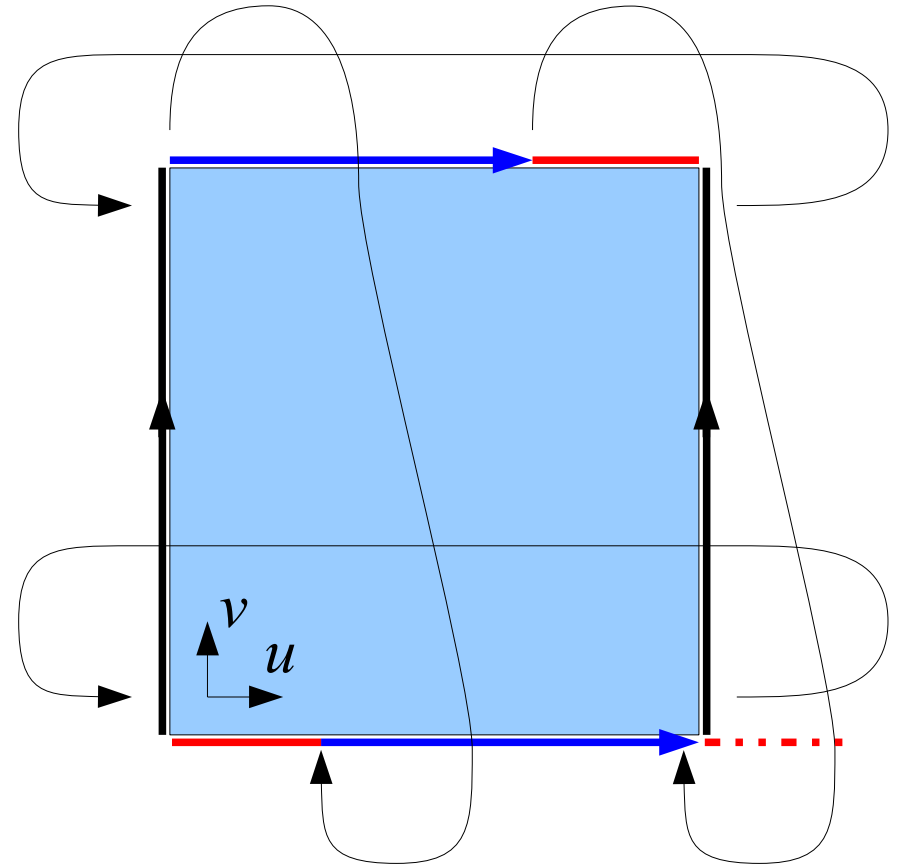
$$U = \{-3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\} \quad p=3$$

$$V = \{0, 0, 0, 1, 2, 3, 4, 5, 6, 7, 7, 7\} \quad p=2$$

B-Spline surfaces

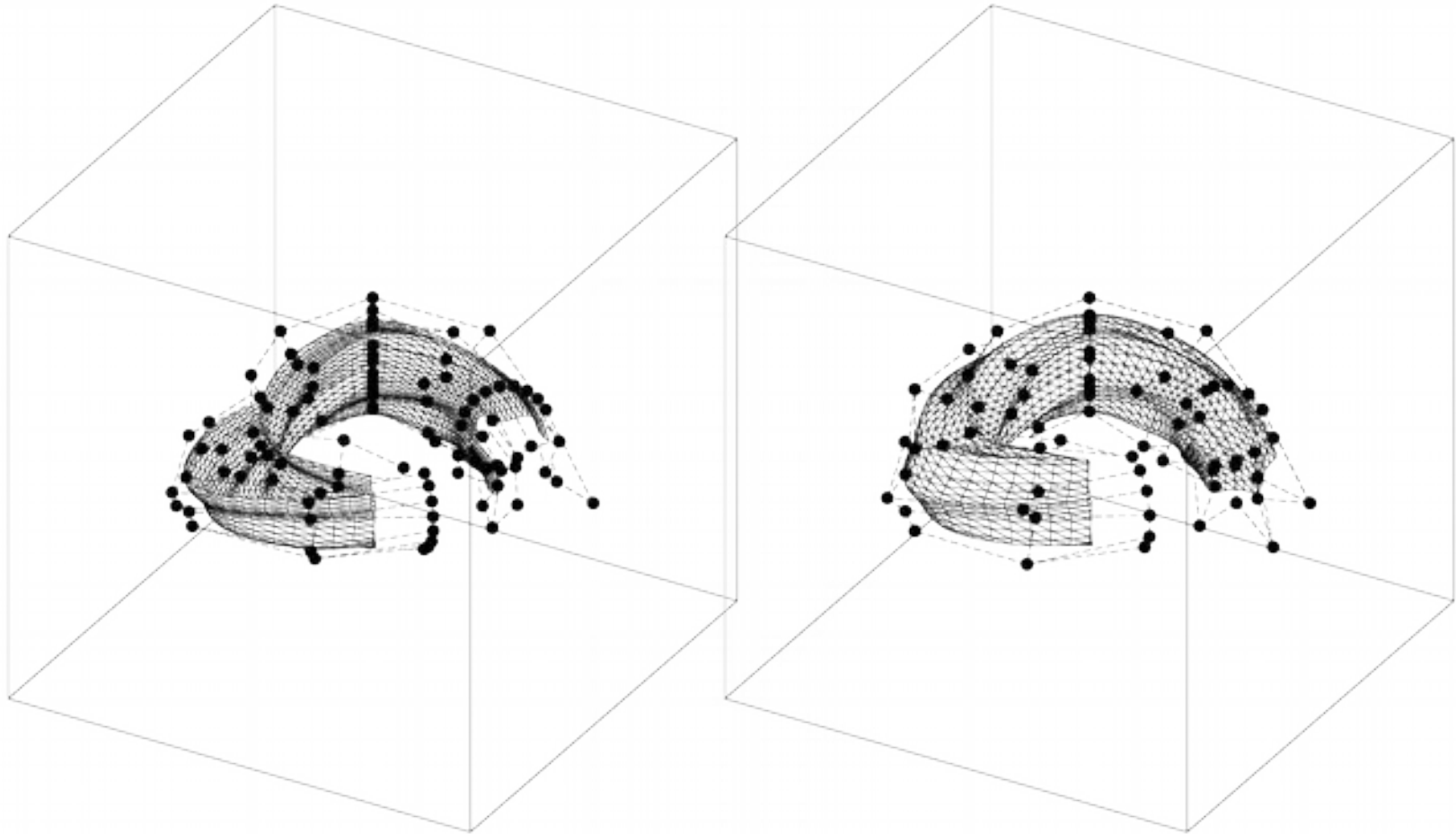


Torus

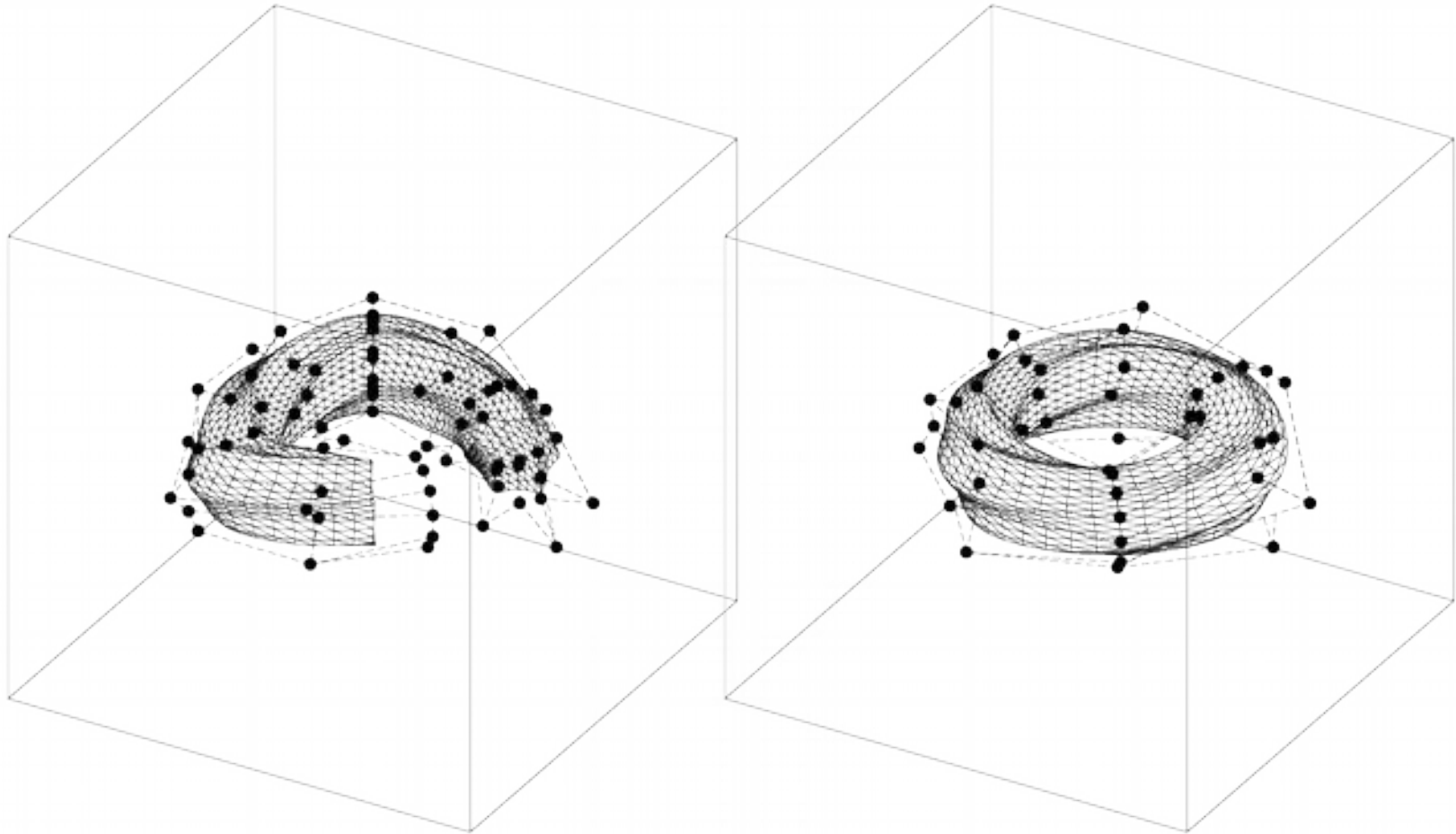


Twisted Torus...

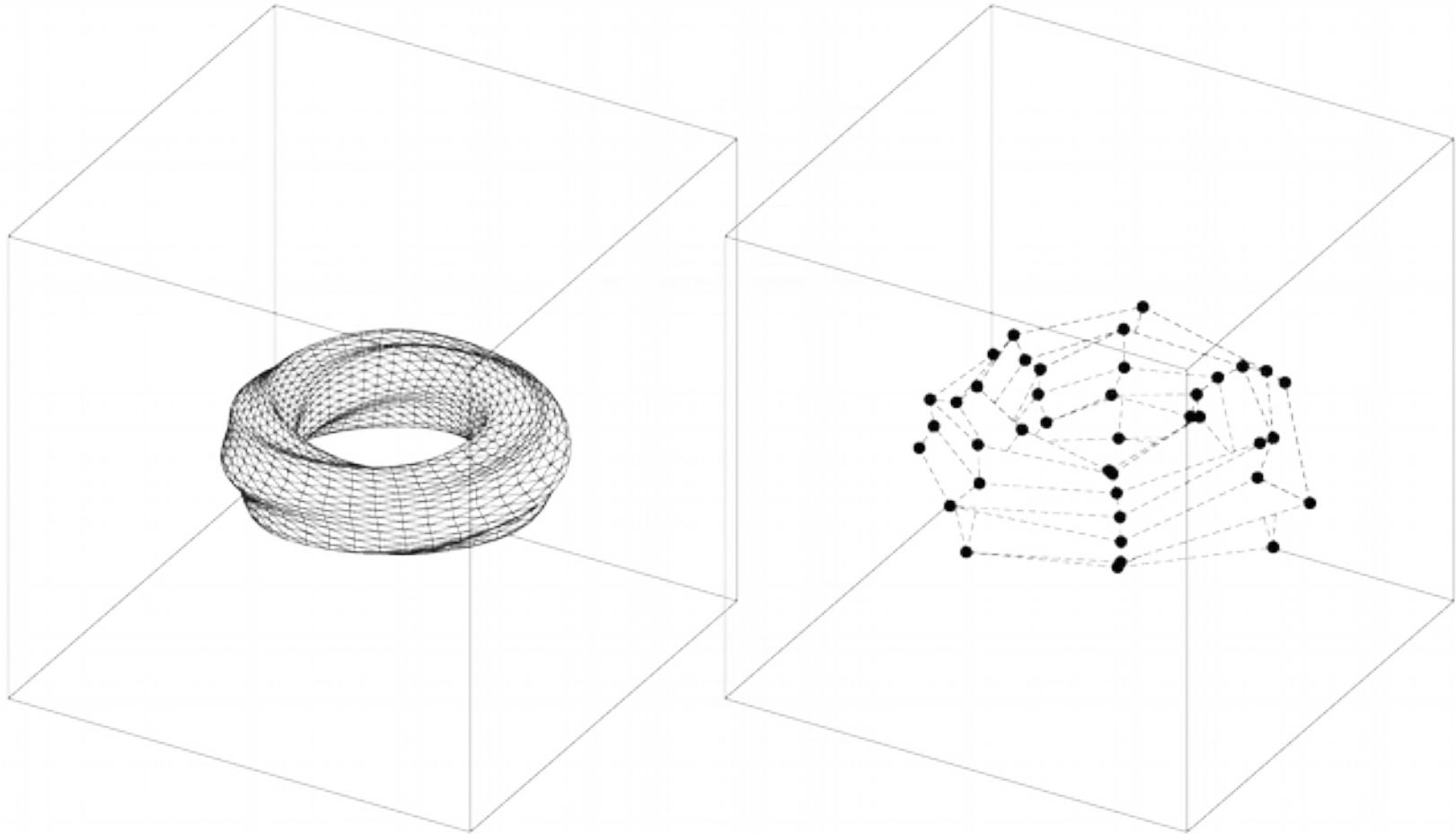
B-Spline surfaces



B-Spline surfaces



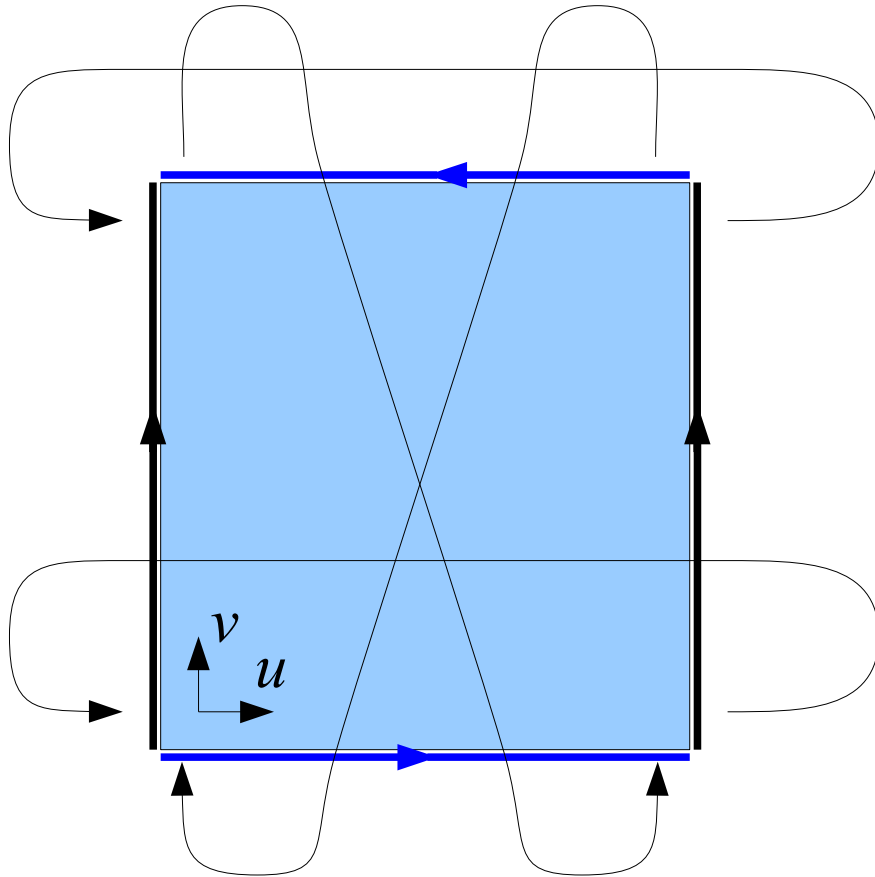
B-Spline surfaces



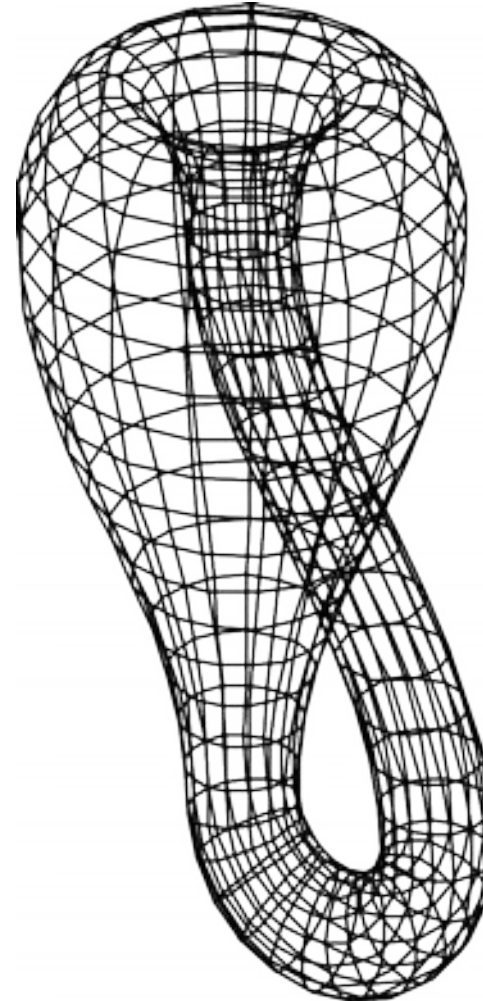
$$U = \{-3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\} \quad p=3$$

$$V = \{-2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \quad p=2$$

B-Spline surfaces



Klein's bottle



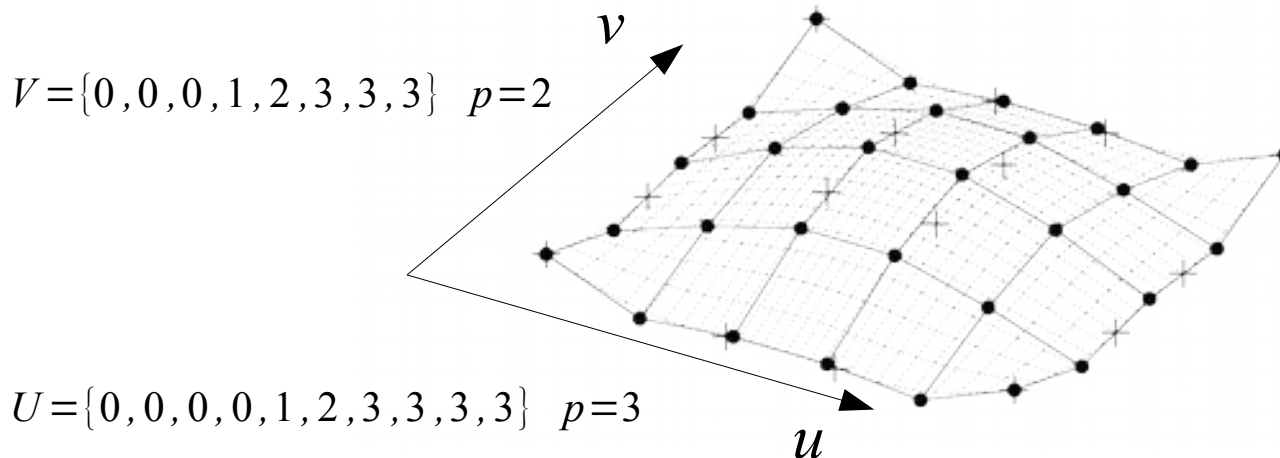
B-Spline surfaces

- Some manipulations
 - Insertion of nodes
 - Extraction of iso-parametrics
 - Calculation of the position of a point on the surface
 - Subdivision of the surface
 - Transformation into Bézier patches

B-Spline surfaces

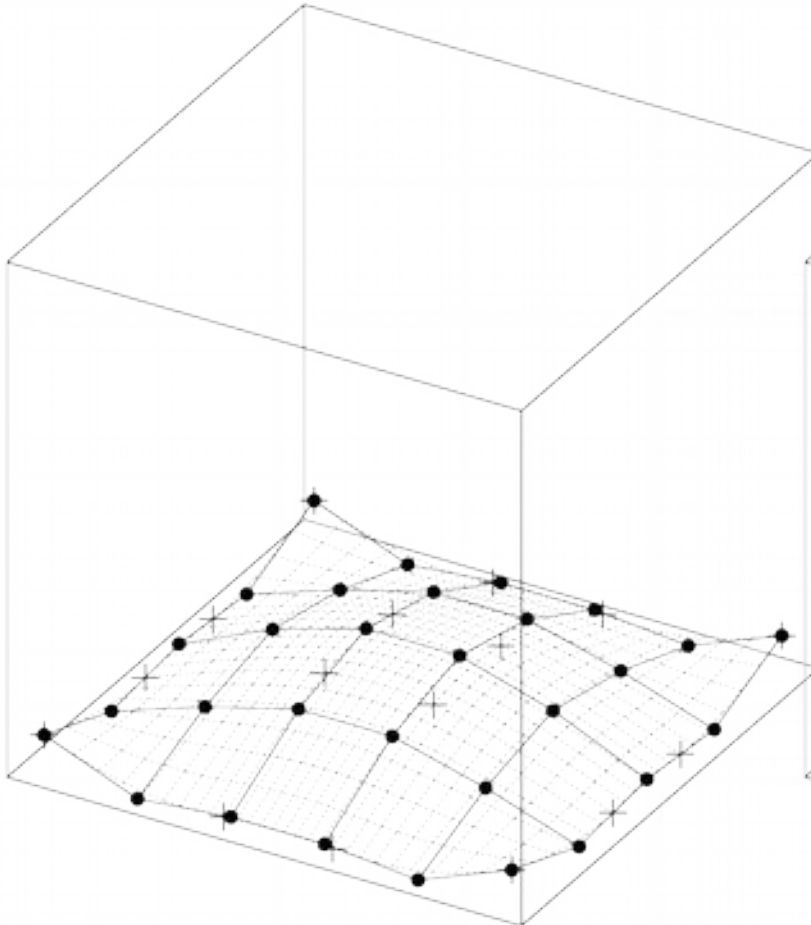
- Insertion of nodes

- We insert nodes in a nodal sequence (U ou V)
- The new nodal sequence replace the old one
- The control points are modified
 - If U is modified, every series of control points corresponding to $v=cst$ is independently modified
 - If V is modified, every series of control points corresponding to $u=cst$ is independently modified
 - We use Boehm's algorithm as for curves



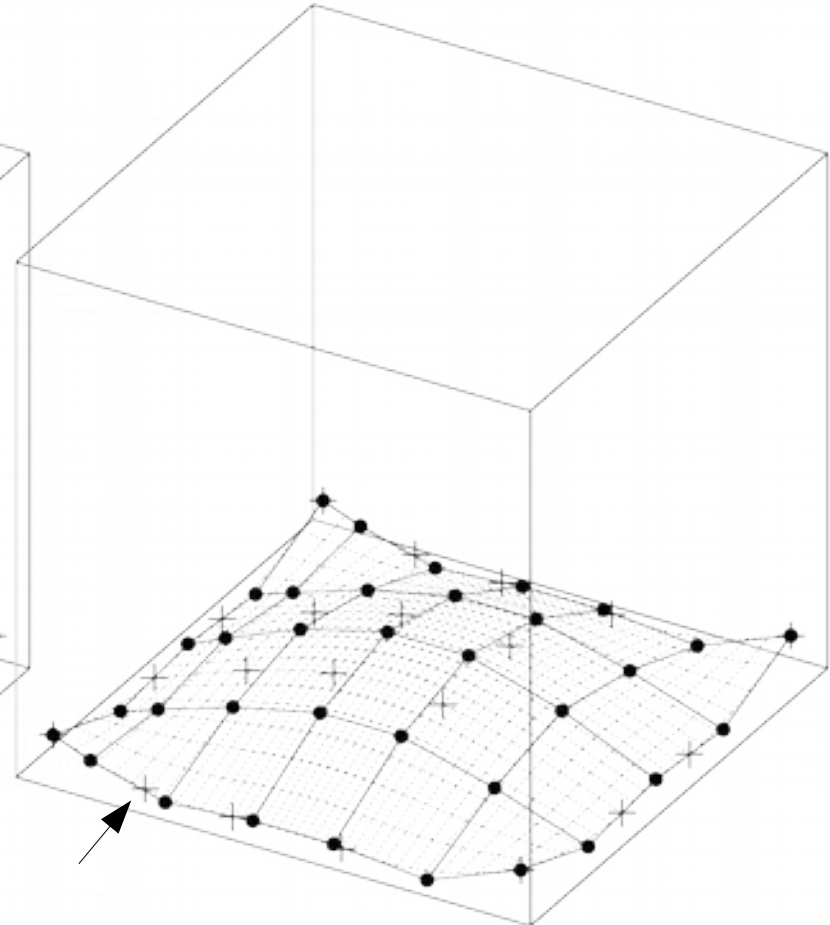
B-Spline surfaces

- Insertion of nodes in u



$$U = \{0, 0, 0, 0, 1, 2, 3, 3, 3, 3\} \quad p=3$$

$$V = \{0, 0, 0, 1, 2, 3, 3, 3\} \quad q=2$$

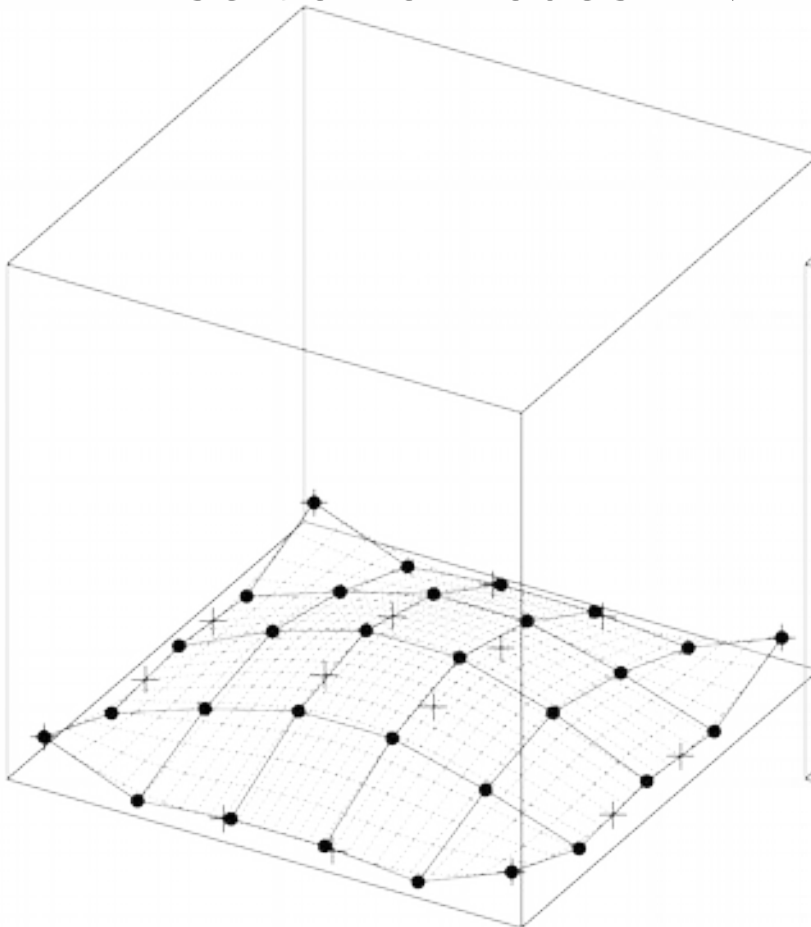


$$U = \{0, 0, 0, 0, 0.4, 1, 2, 3, 3, 3, 3\} \quad p=3$$

$$V = \{0, 0, 0, 1, 2, 3, 3, 3\} \quad q=2$$

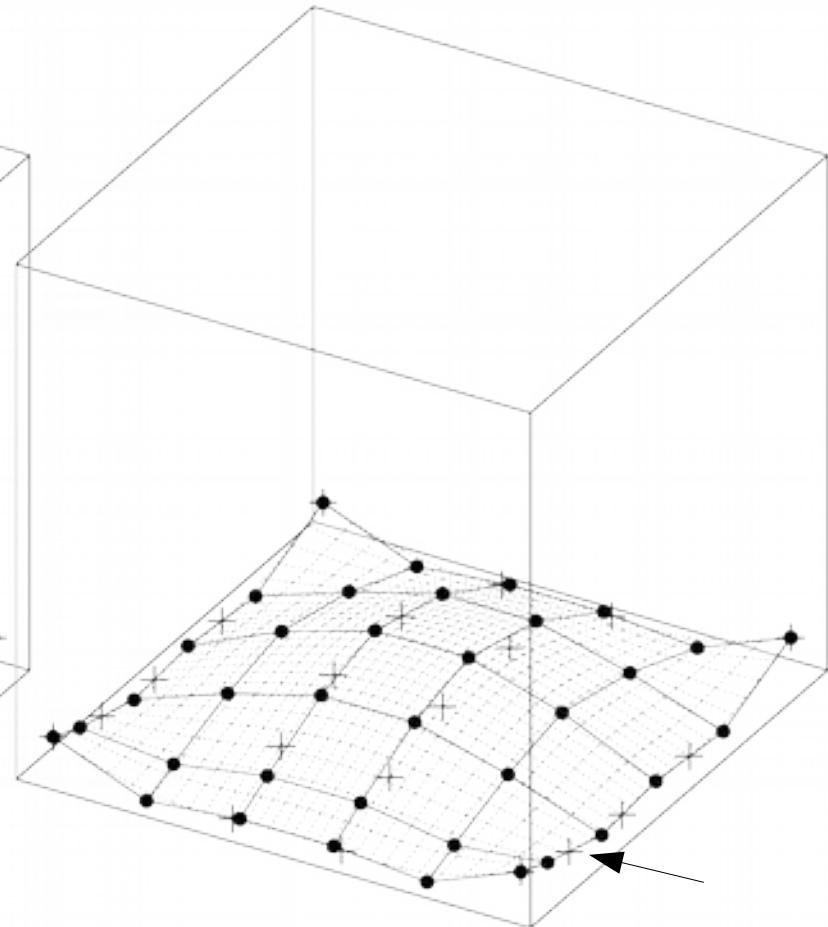
B-Spline surfaces

- Insertion of nodes in v



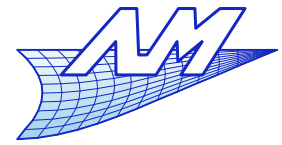
$$U = \{0, 0, 0, 0, 1, 2, 3, 3, 3, 3\} \quad p=3$$

$$V = \{0, 0, 0, 1, 2, 3, 3, 3\} \quad q=2$$



$$U = \{0, 0, 0, 0, 1, 2, 3, 3, 3, 3\} \quad p=3$$

$$V = \{0, 0, 0, 0.4, 1, 2, 3, 3, 3\} \quad q=2$$

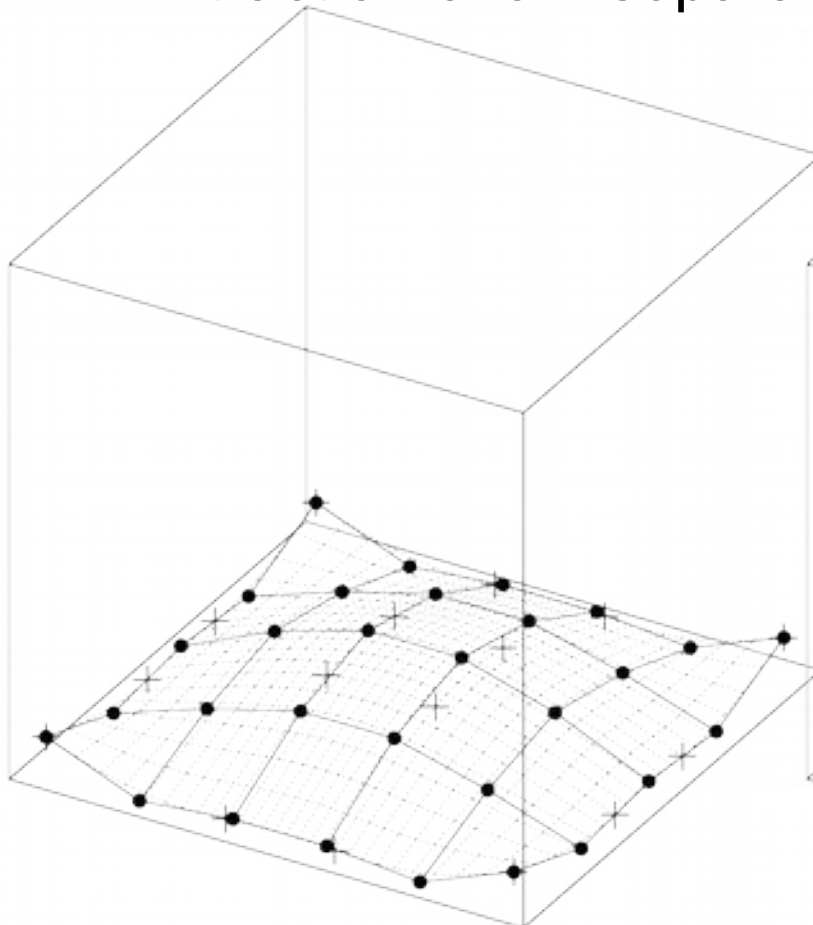


B-Spline surfaces

- Extraction of iso-parametrics using node insertion
 - We must saturate one node in $u=u_{iso}$ (resp. in $v=v_{iso}$).
 - The new control points obtained by Boehm's algorithm do form the control polygon of the iso-parametric curve.
 - The nodal sequence of this curve is V (resp. U).

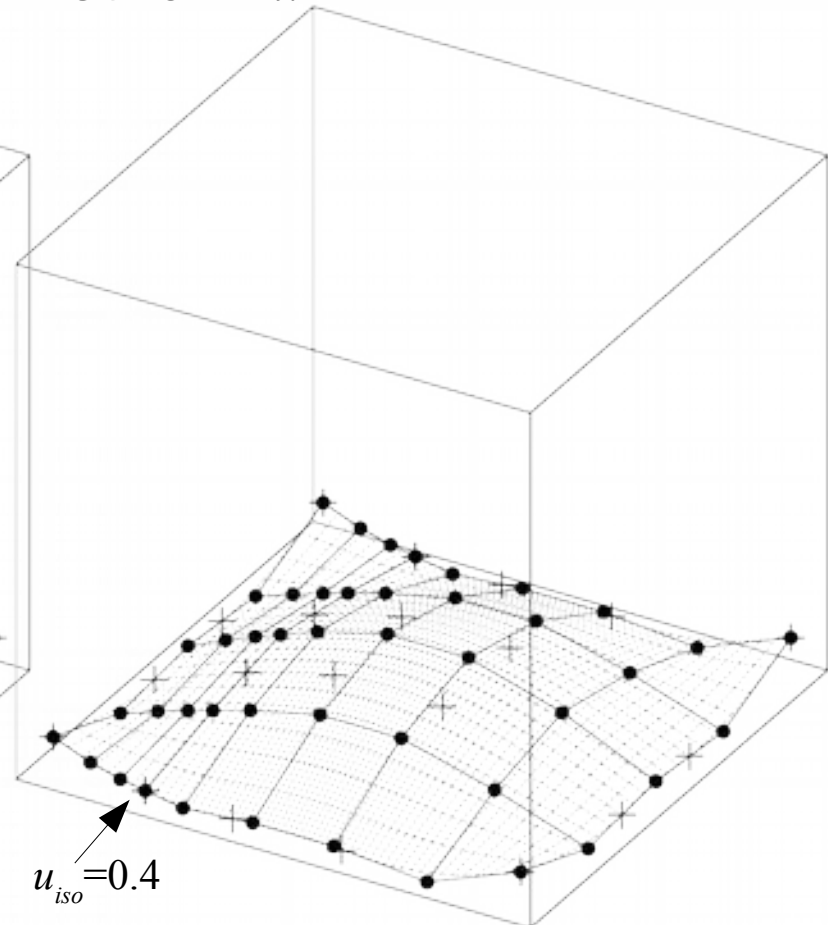
B-Spline surfaces

- Extraction of an isoparametric in u



$$U = \{0, 0, 0, 0, 1, 2, 3, 3, 3, 3\} \quad p=3$$

$$V = \{0, 0, 0, 1, 2, 3, 3, 3\} \quad q=2$$

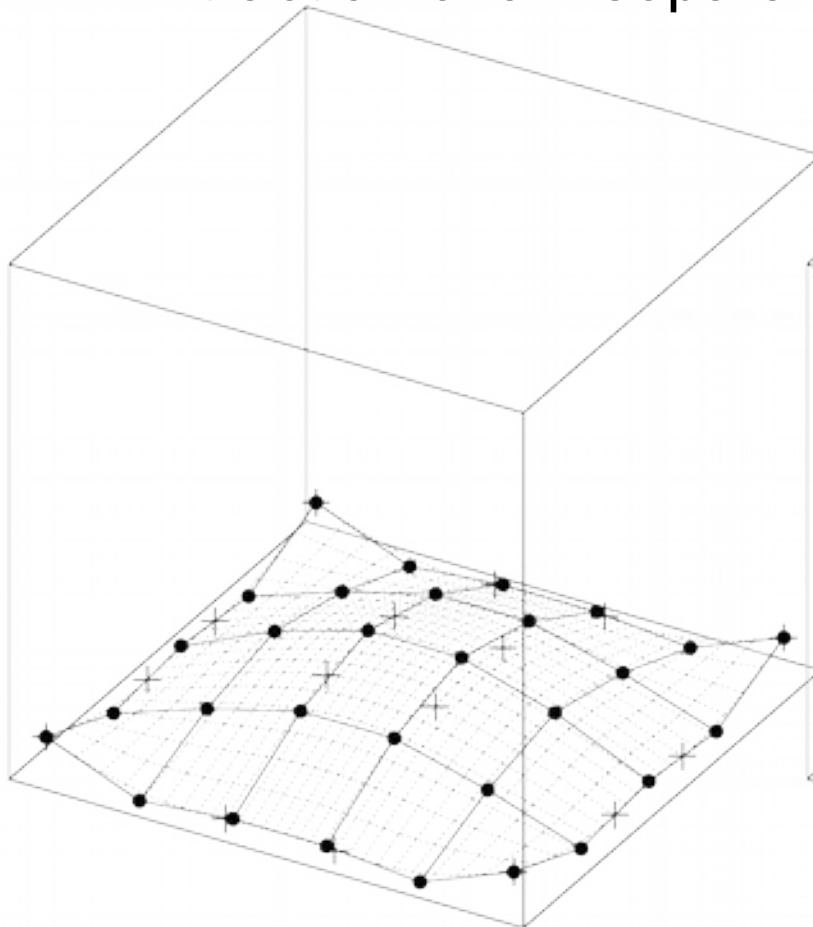


$$U = \{0, 0, 0, 0, 0.4, 0.4, 0.4, 1, 2, 3, 3, 3, 3\} \quad p=3$$

$$V = \{0, 0, 0, 1, 2, 3, 3, 3\} \quad q=2$$

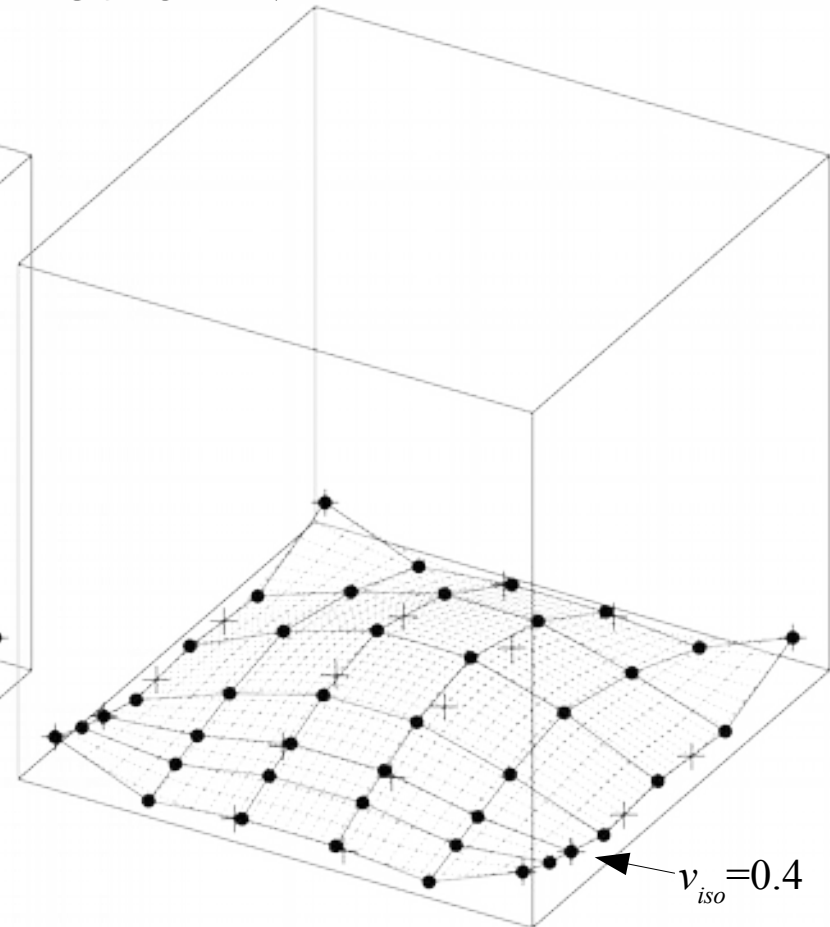
B-Spline surfaces

- Extraction of an isoparametric in v



$$U = \{0, 0, 0, 0, 1, 2, 3, 3, 3, 3\} \quad p=3$$

$$V = \{0, 0, 0, 1, 2, 3, 3, 3\} \quad q=2$$

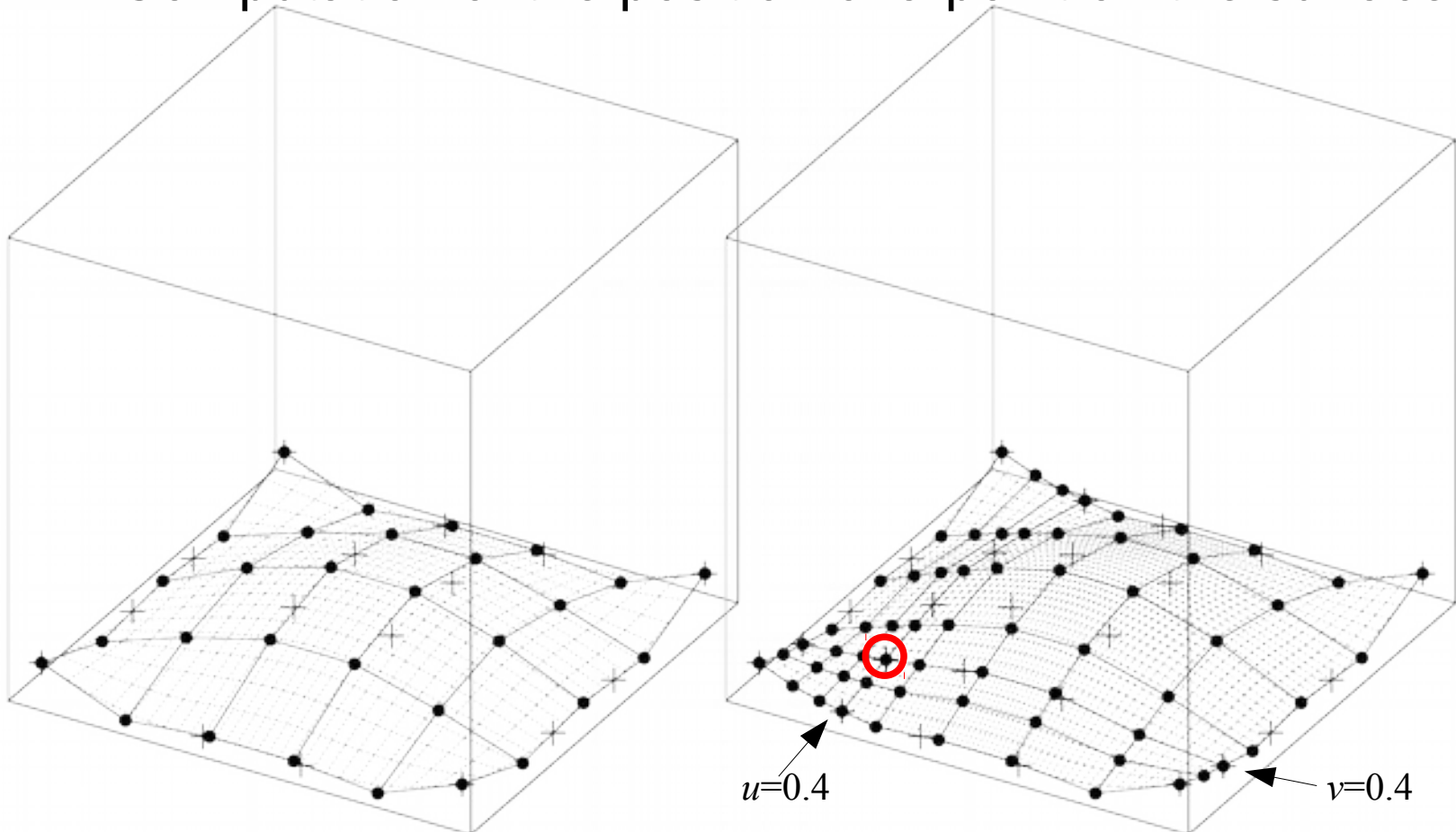


$$U = \{0, 0, 0, 0, 1, 2, 3, 3, 3, 3\} \quad p=3$$

$$V = \{0, 0, 0, 0.4, 0.4, 1, 2, 3, 3, 3\} \quad q=2$$

B-Spline surfaces

- Computation of the position of a point on the surface



$$U = \{0, 0, 0, 0, 1, 2, 3, 3, 3, 3\} \quad p=3$$

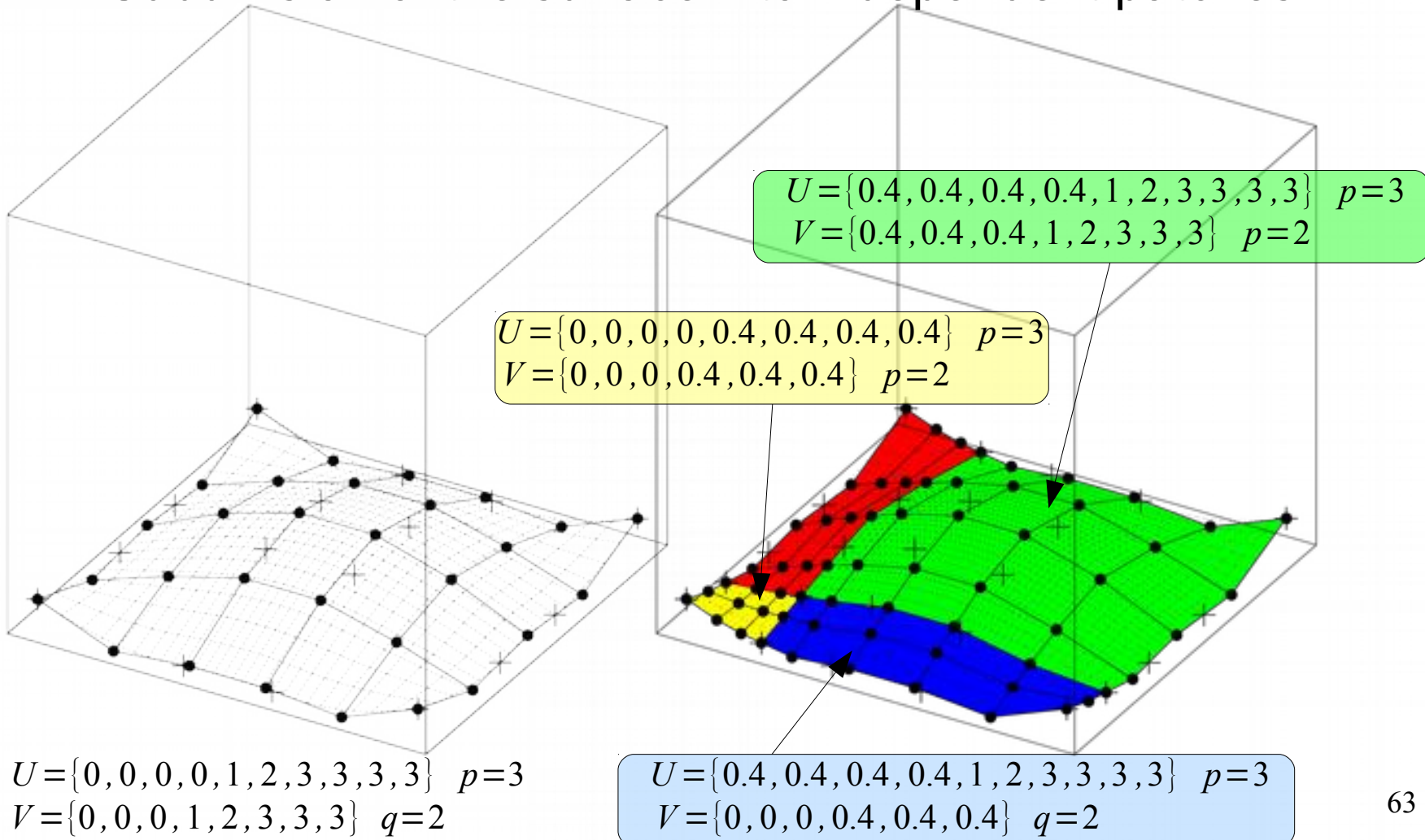
$$V = \{0, 0, 0, 1, 2, 3, 3, 3\} \quad q=2$$

$$U = \{0, 0, 0, 0, 0.4, 0.4, 0.4, 1, 2, 3, 3, 3, 3\} \quad p=3$$

$$V = \{0, 0, 0, 0.4, 0.4, 1, 2, 3, 3, 3\} \quad q=2$$

B-Spline surfaces

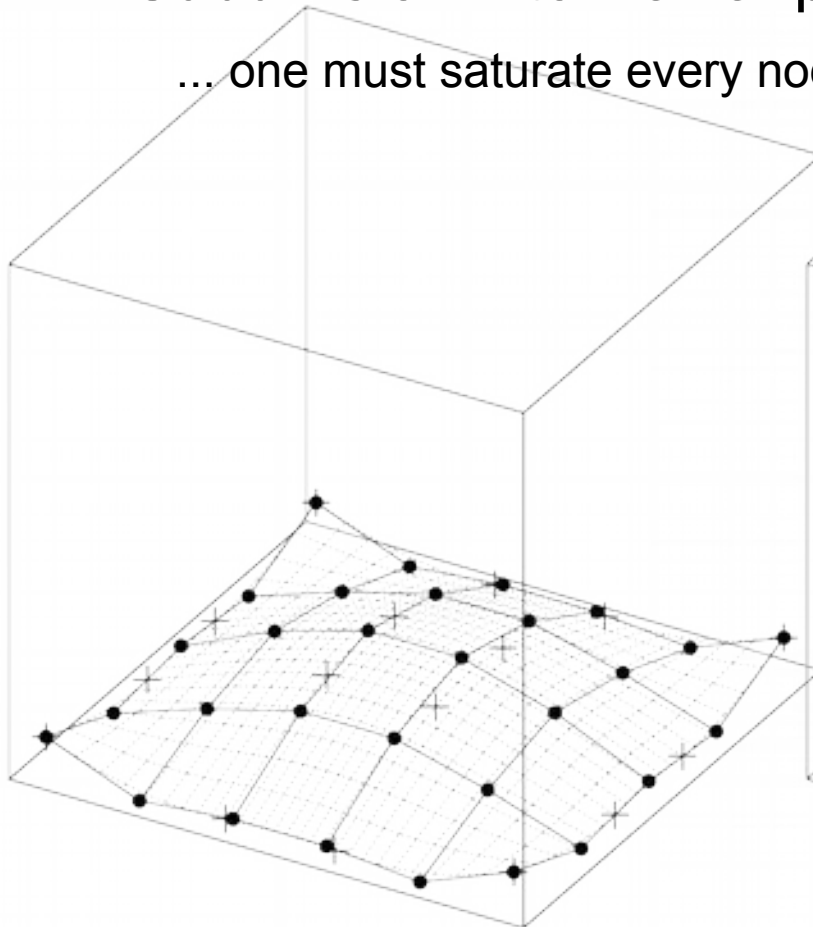
- Subdivision of the surface into independent patches



B-Spline surfaces

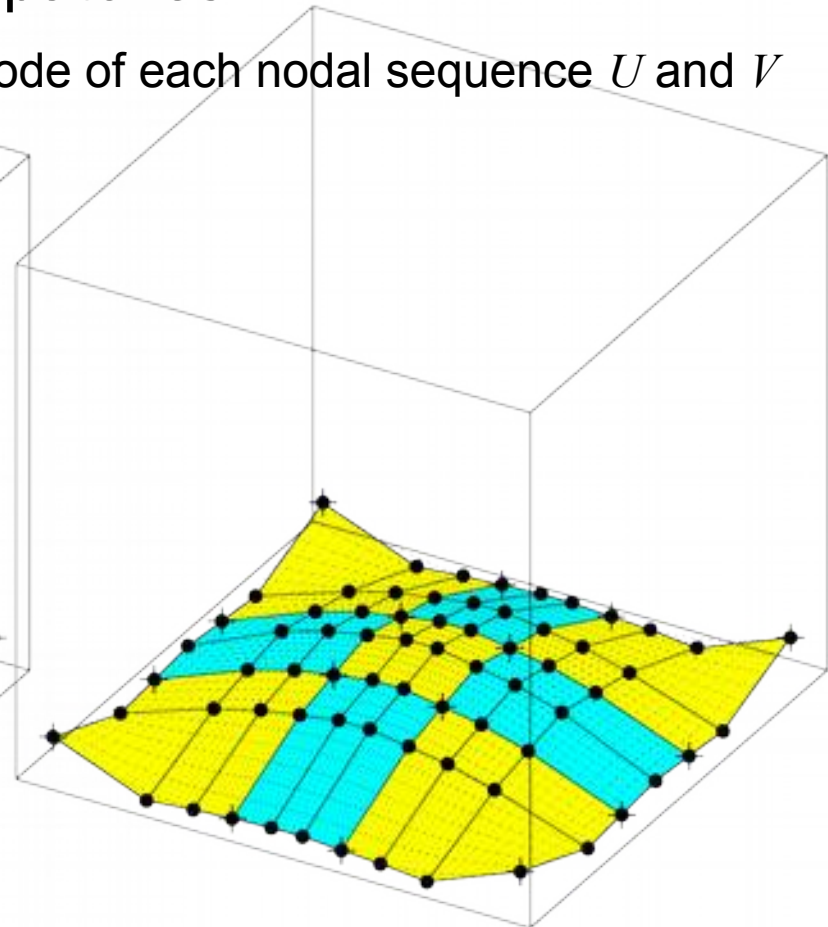
- Subdivision into Bézier patches

... one must saturate every node of each nodal sequence U and V



$$U = \{0, 0, 0, 0, 1, 2, 3, 3, 3, 3\} \quad p=3$$

$$V = \{0, 0, 0, 1, 2, 3, 3, 3\} \quad q=2$$

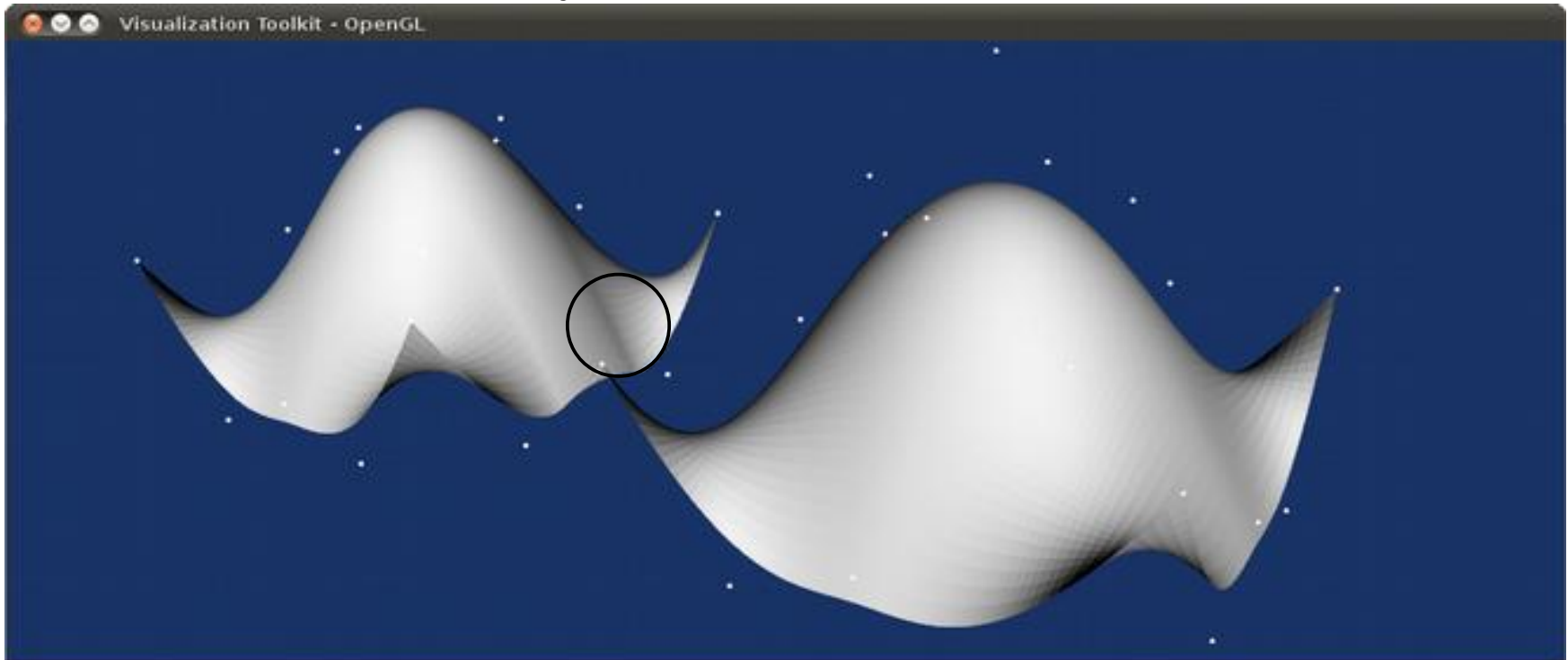


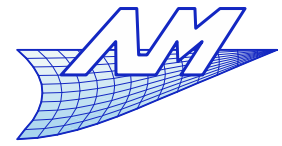
$$U = \{0, 0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 3, 3\} \quad p=3$$

$$V = \{0, 0, 0, 1, 1, 2, 2, 3, 3, 3\} \quad q=2$$

B-Spline surfaces

- Continuity requirements for surfaces
 - C^1 vs C^2 – becomes visible when light interaction comes into play

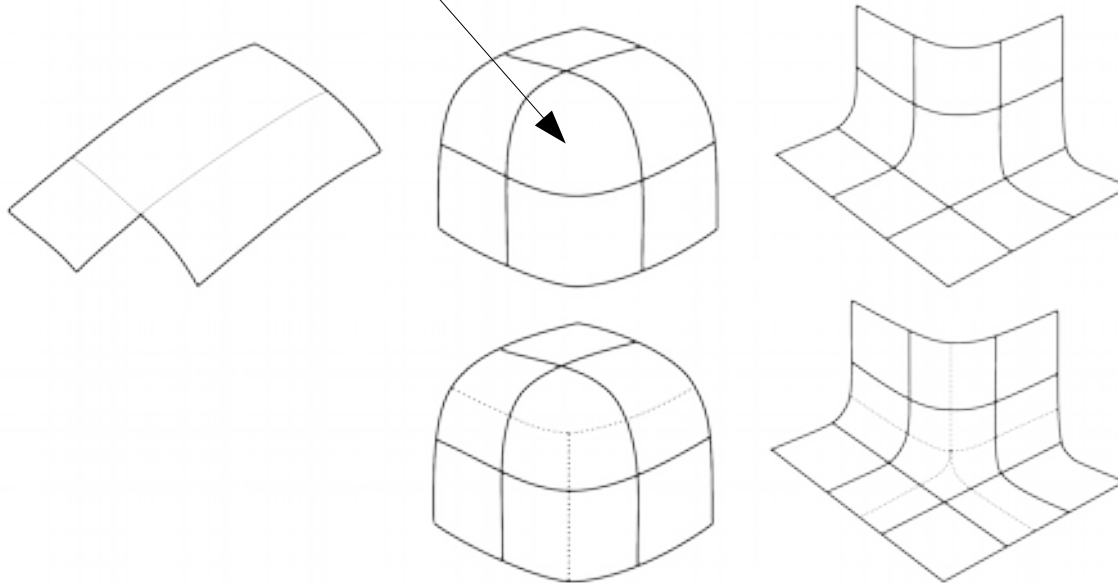




Bézier triangle

Bézier triangle

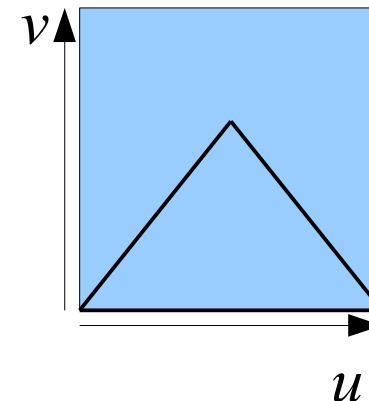
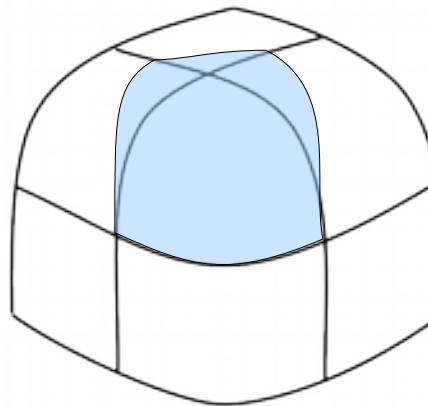
- Need for specific topology
 - Box corner aka « coin de valise » (in French)



S. Hahmann

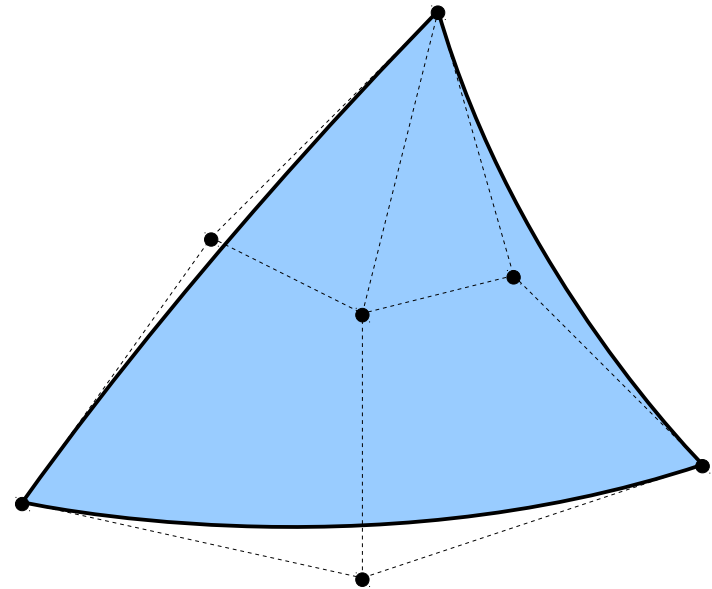
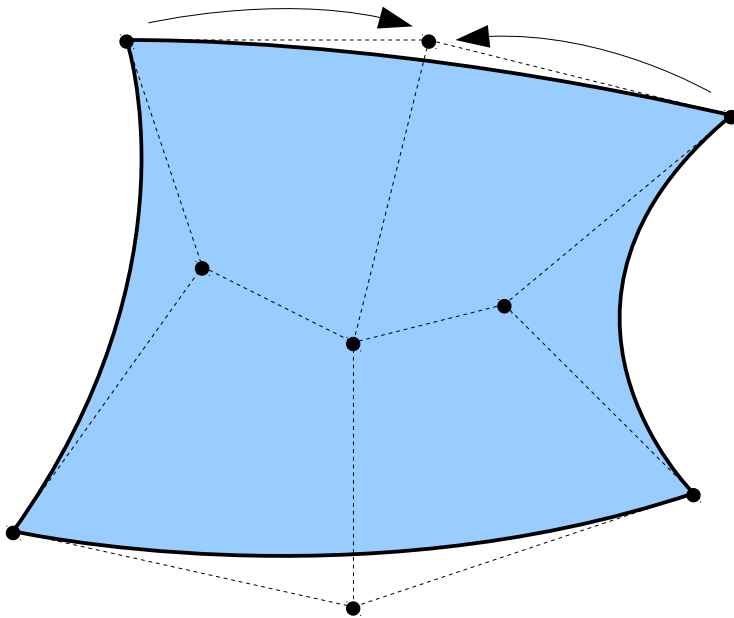
Bézier triangle

- There are several techniques to model the corner
 - It's more difficult than one thinks ...
 - On may take a regular patch with 4 sides and « limit » it by a triangle in the parametric space
 - Problem : The surface in question is not built with the control points of the other surfaces, so any continuity is difficult to enforce (minimization of a non linear functional)

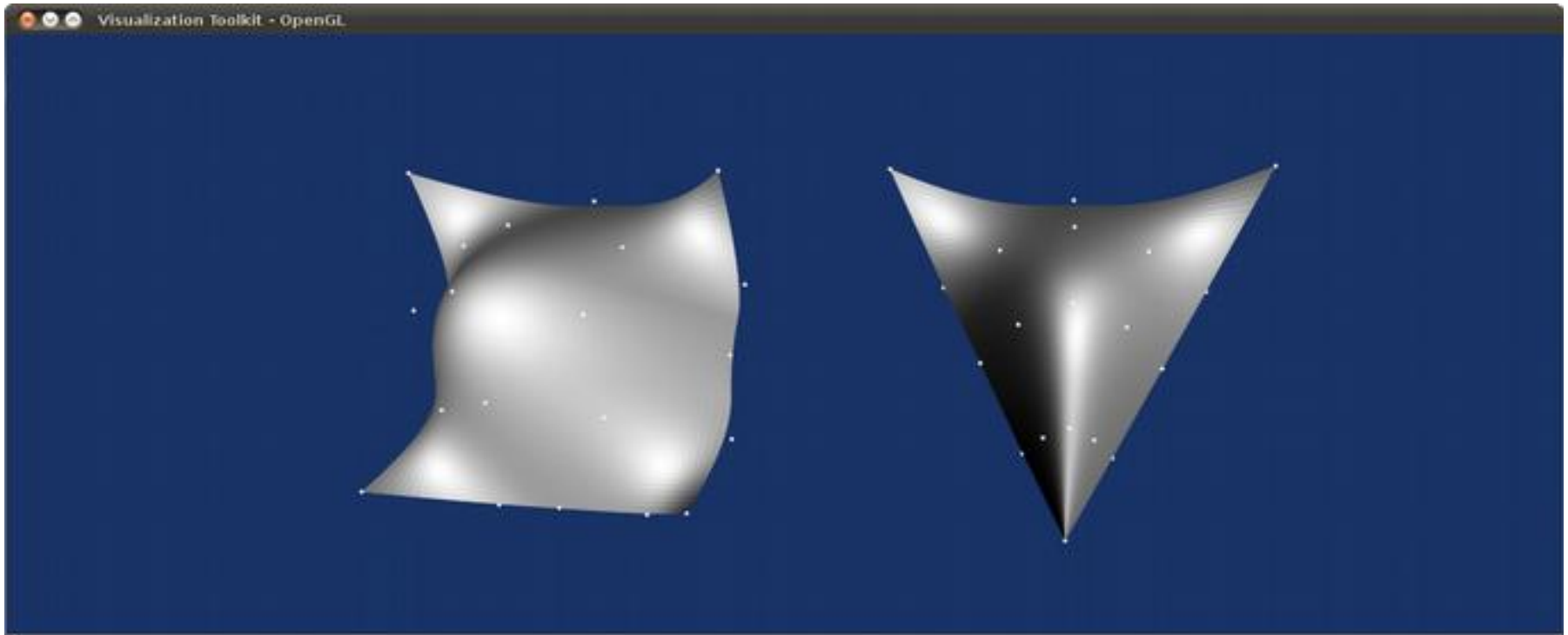


Bézier triangle

- Degenerated quadrangular patch
 - Normals and derivatives are undefined at the singular point



Bézier triangle



Bézier triangle

- Triangular B-splines

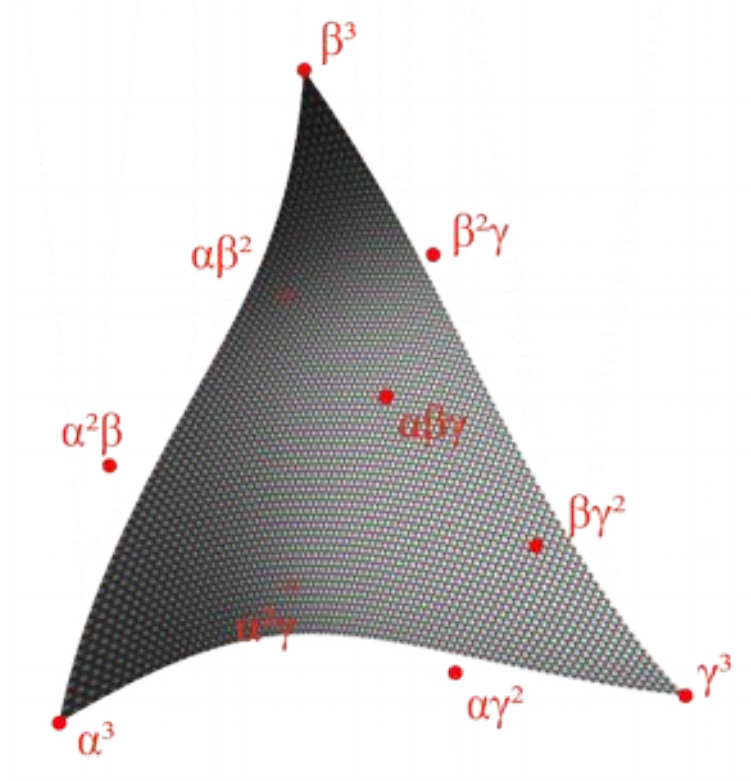
- 1992 : works of Dahmen, Micchelli et Seidel

W. Dahmen, C.A. Micchelli and H.P. Seidel, Blossoming begets B-Splines built better by B-patches, *Mathematics of Computation*, 59 (199), pp. 97-115, 1992

- Extension of the definition of B-Splines on triangular surfaces of any topology
 - Network of control points
 - « Mesh » of non structured topology instead of a structured network as for B-splines surfaces
 - Complex and not usually not implemented in current CAD software, therefore not a “standard” tool.

Bézier triangle

- Triangular Bézier Surfaces
 - Example : surface of order 3



Bézier triangle

- Barycentric coordinates

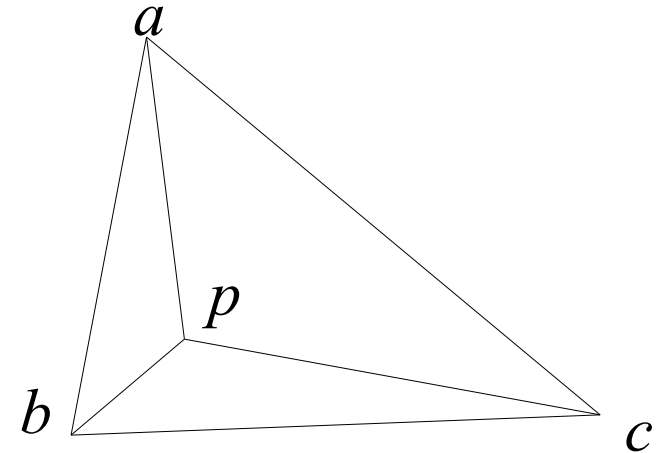
- $p = u \cdot a + v \cdot b + w \cdot c$

- $u + v + w = 1$

- Affine invariance

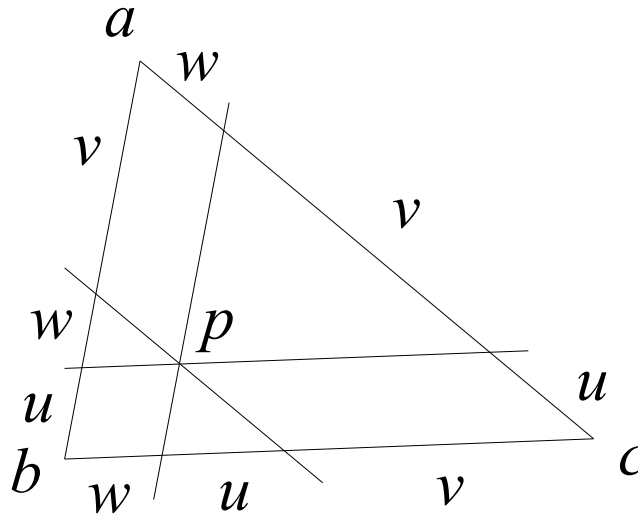
- $0 \leq u, v, w \leq 1 \Leftrightarrow p$ is in the triangle

- $u = \frac{\text{area}(p, b, c)}{\text{area}(a, b, c)}$ $v = \frac{\text{area}(p, c, a)}{\text{area}(a, b, c)}$ $w = \frac{\text{area}(p, a, b)}{\text{area}(a, b, c)}$



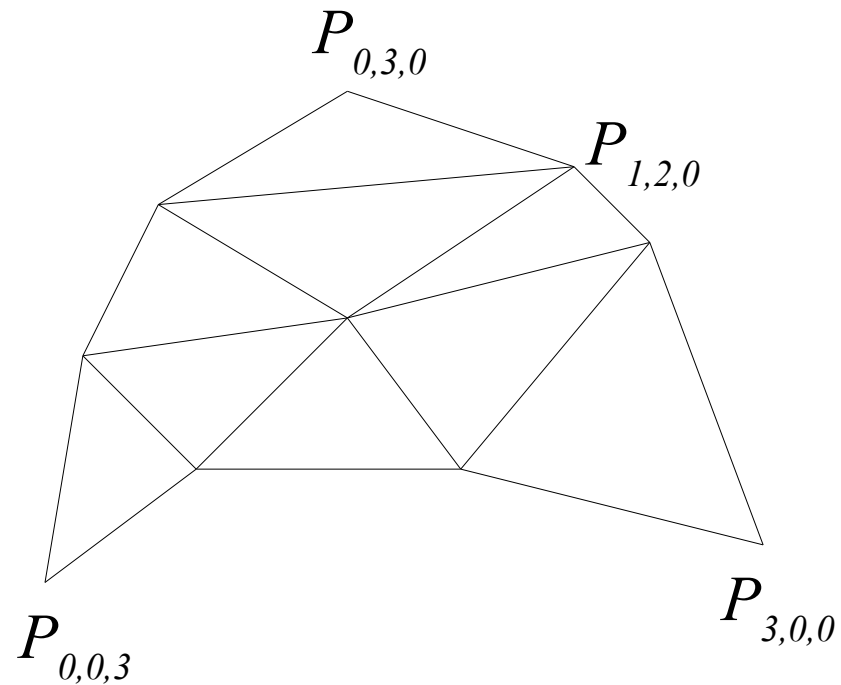
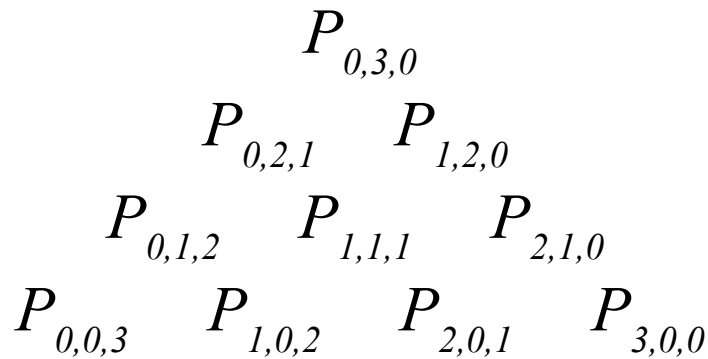
Bézier triangle

- Barycentric coordinates



Bézier triangle

- Decomposition of the Bézier triangle
 - Defined by the control points $P_{i,j,k}$
 - Degree $d : i+j+k=d$
 - Example with $d=3$



- Overall, $\frac{(d+2)(d+1)}{2}$ control points.

Bézier triangle

- De Casteljau's algorithm on the Bézier triangle

- The $P_{i,j,k}$ are given
- We want to compute $P(u,v,w)$ with $u+v+w=1$
- We follow the next algorithm :
initialize $P_{i,j,k}^0(u,v,w) = P_{i,j,k}$

for r from 1 to d and for every triplet (i,j,k) s.t. $i+j+k=d-r$

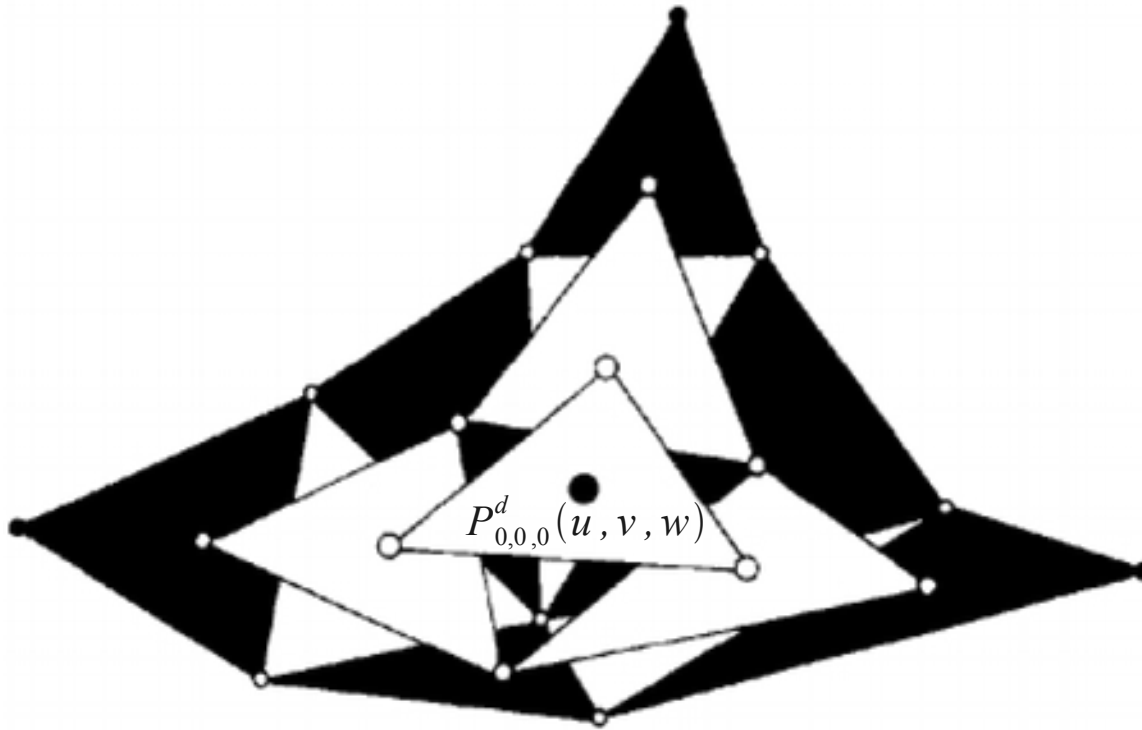
$$P_{i,j,k}^r(u,v,w) = u P_{i+1,j,k}^{r-1}(\dots) + v P_{i,j+1,k}^{r-1}(\dots) + w P_{i,j,k+1}^{r-1}(\dots)$$

The point on the surface is the last point :

$$P(u,v,w) = P_{0,0,0}^d(u,v,w)$$

Bézier triangle

- De Casteljau's algorithm on the Bézier triangle



Bézier triangle

- Characteristics of the Bézier triangle
 - Affine invariance
 - Contained in the convex hull of the control points
 - Interpolation of extremal vertices
 - Edges of the Bézier triangle are in fact Bézier curves
 - Algebraic form : another form of Bernstein polynomials

$$P(u, v, w) = \sum_{i+j+k=d} B_{i,j,k}^d(u, v, w) P_{i,j,k}$$

with (recurrence) $B_{i,j,k}^d(u, v, w) = \frac{d!}{i!j!k!} u^i v^j w^k$

$$B_{i,j,k}^d(u, v, w) = uB_{i-1,j,k}^{d-1}(\dots) + vB_{i,j-1,k}^{d-1}(\dots) + wB_{i,j,k-1}^{d-1}(\dots)$$

$$B_{0,0,0}^0(u, v, w) = 1$$

Bézier triangle

- Characteristics (following)
 - On the contrary to tensor product surfaces, the Bézier triangle is **variation diminishing**.

Bézier triangle

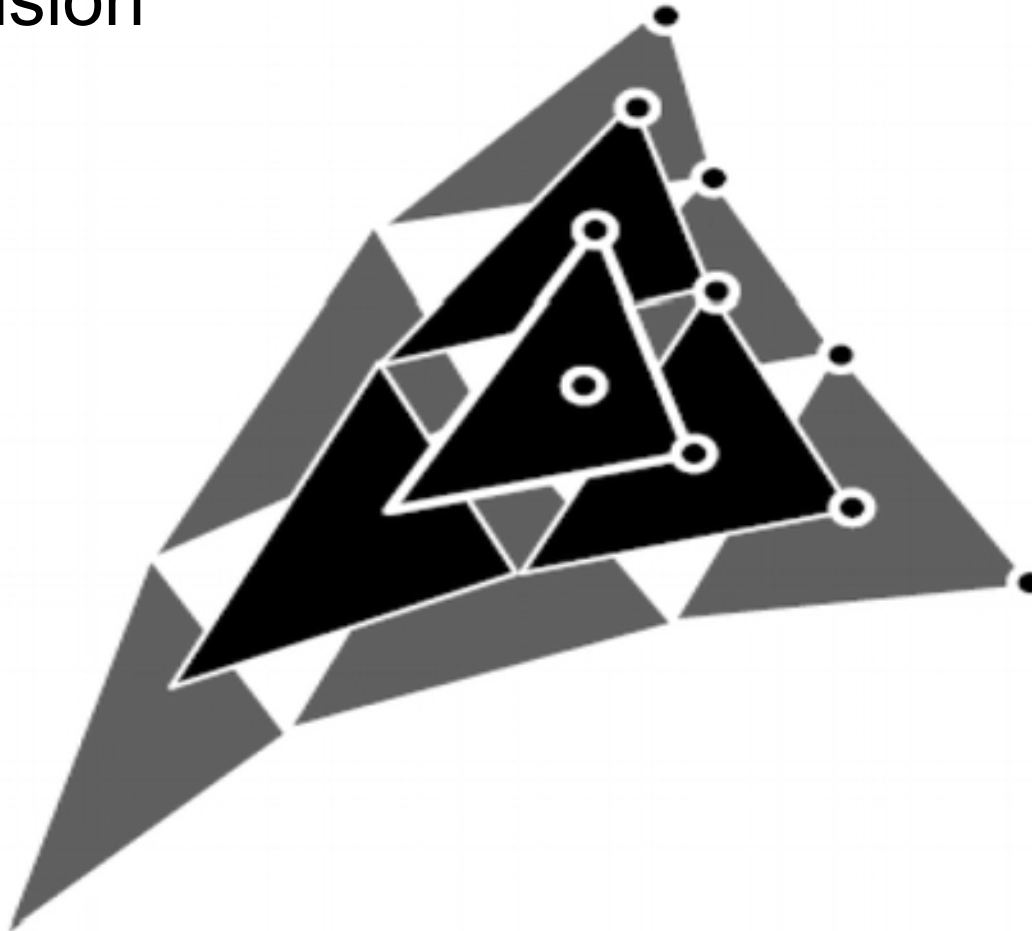
- Degree elevation
 - The $P_{i,j,k}$ are given, we search the $P'_{i,j,k}$ corresponding to the same surface of degree $d+1$
 - Forrest's relations for the Bézier triangle

$$P'_{i,j,k} = \frac{1}{d+1} (i P_{i-1,j,k} + j P_{i,j-1,k} + k P_{i,j,k-1})$$

$$\begin{aligned} P(u, v, w) &= \sum_{i+j+k=d+1} B_{i,j,k}^{d+1}(u, v, w) P'_{i,j,k} \\ &= \sum_{i+j+k=d} B_{i,j,k}^d(u, v, w) P_{i,j,k} \end{aligned}$$

Bézier triangle

- Subdivision



Bézier triangle

- Derivatives

- For tensor product surfaces, partial derivatives are computed along $u=\text{const}$ or $v=\text{const}$
- Here, we express directional derivatives for $u=\text{const}$; $v=\text{const}$ or $w=\text{const}$. - these are not partial derivatives !

$$D_u(P(u, v, w)) = \lim_{t \rightarrow 0} \frac{P(u, v + tdv, w + tdw) - P(u, v, w)}{t}$$

Bézier triangle

- Case of a surfaces of degree 3

$$\begin{array}{cccc}
 & & & P(0,1,0) = P_{0,3,0} \\
 & & P_{0,3,0} & \\
 & P_{0,2,1} & P_{1,2,0} & \\
 P_{0,1,2} & P_{1,1,1} & P_{2,1,0} & \\
 P_{0,0,3} & P_{1,0,2} & P_{2,0,1} & P_{3,0,0}
 \end{array}$$

$$\begin{aligned}
 D_u(P(u, v, w))(0,1,0) &= 3(P_{0,2,1} - P_{0,3,0}) \\
 D_v(P(u, v, w))(0,1,0) &= 3(P_{1,2,0} - P_{0,2,1}) \\
 D_w(P(u, v, w))(0,1,0) &= 3(P_{1,2,0} - P_{0,3,0})
 \end{aligned}$$

$$D_{uu}(P(u, v, w))(0,1,0) = 6(P_{0,1,2} - 2P_{0,2,1} + P_{0,3,0})$$

$$D_{vv}(P(u, v, w))(0,1,0) = 6(P_{2,1,0} - 2P_{1,1,1} + P_{0,1,2})$$

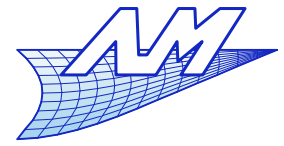
$$D_{ww}(P(u, v, w))(0,1,0) = 6(P_{2,1,0} - 2P_{1,2,0} + P_{0,3,0})$$

Same at
other corners

$$D_{uv}(P(u, v, w))(0,1,0) = 6(P_{1,1,1} + P_{0,2,1} - P_{0,1,2} - P_{1,2,0})$$

$$D_{uw}(P(u, v, w))(0,1,0) = 6(P_{1,1,1} + P_{0,3,0} - P_{0,2,1} - P_{1,2,0})$$

$$D_{vw}(P(u, v, w))(0,1,0) = 6(P_{2,1,0} + P_{0,2,1} - P_{1,2,0} - P_{1,1,1})$$



Subdivision surfaces

Subdivision surfaces

- Parametric surfaces: an explicit representation
 - Lightweight
 - Discretization algorithms are non trivial... but it is necessary for display purposes and in computer graphics
 - Generally, these surfaces are used in cases where the geometric accuracy is essential, as in the computation of intersections and other precise geometric primitives
 - Modelling operators are non trivial
 - In computer graphics, such accuracy is generally not needed.

Subdivision surfaces

- Subdivision surfaces
 - Modelling basis = elementary mesh
 - By successive iterations, this mesh is refined up to the accuracy needed for the application
 - It is more like an algorithmic description vs. an algebraic representation, because the algorithm that is used to subdivide the mesh determines the final shape and the properties of the limiting surface (*ie.* when the number of subdivisions tends to the infinite)
 - Some of these limiting surfaces are equivalent to “regular” parametric surfaces, therefore have the same “accuracy”.

Subdivision surfaces

- History

- 1974 – George Chaikin

- An algorithm for high speed curve generation

- 1978 – Daniel Doo & Malcolm Sabin

- (D) A subdivision algorithm for smoothing irregularly shaped polyhedrons
(D&S) Behaviour of recursive division surfaces near extraordinary points.

- 1978 – Edwin Catmull & Jim Clark

- Recursively generated B-Spline surfaces on arbitrary topological meshes

- 1987 – Charles Loop

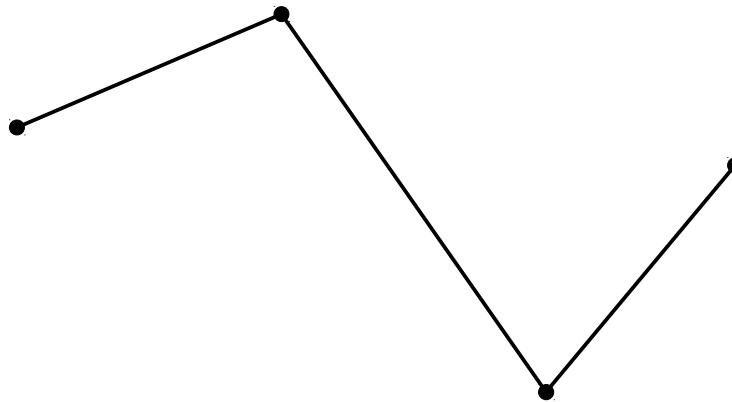
- Smooth subdivision surfaces based on triangles

- 2000 – Leif Kobbelt

- $\sqrt{3}$ – subdivision (interpolating scheme)

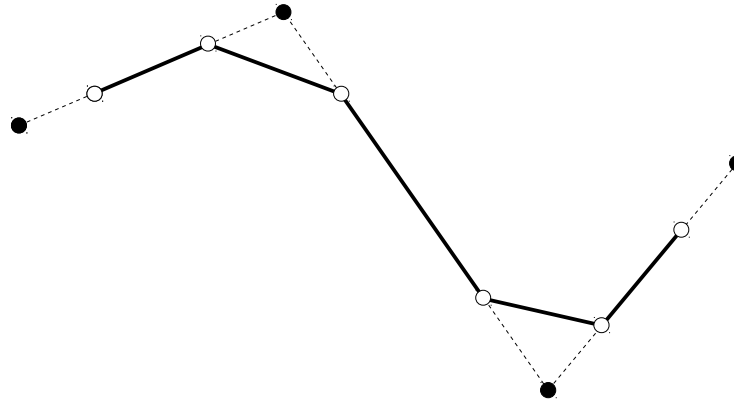
Subdivision surfaces

- Chaikin's scheme



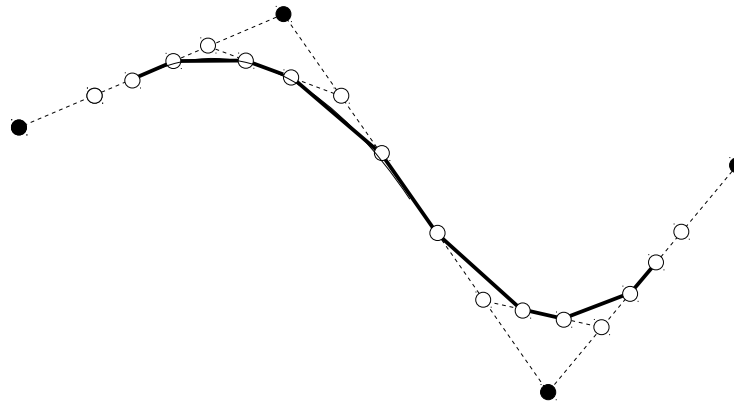
Subdivision surfaces

- Chaikin's scheme
or « Corner-cutting »



Subdivision surfaces

- Chaikin's scheme



Chaikin's idea was simple : repeating the corner cutting, to the limit, one obtains a smooth curve

Subdivision surfaces

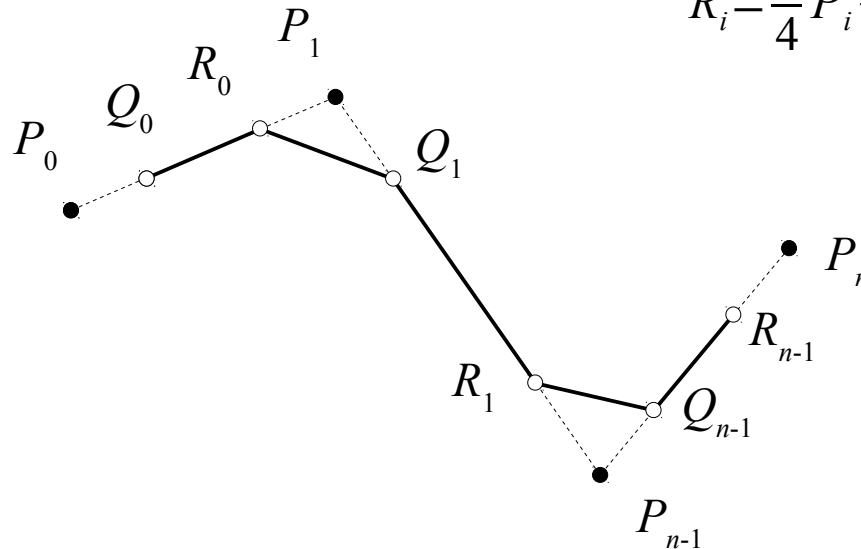
- Chaikin's scheme

- Starting from a polygon having n vertices $\{P_0, P_1, \dots, P_{n-1}\}$, one builds the polygon having $2n$ vertices $\{Q_0, R_0, Q_1, R_1, \dots, Q_{n-1}, R_{n-1}\}$. This polygon serves as a basis for the next step of the algorithm: $\{P'_0, P'_1, \dots, P'_{2n-1}\}$

- The new vertices are :

$$Q_i = \frac{3}{4} P_i + \frac{1}{4} P_{i+1}$$

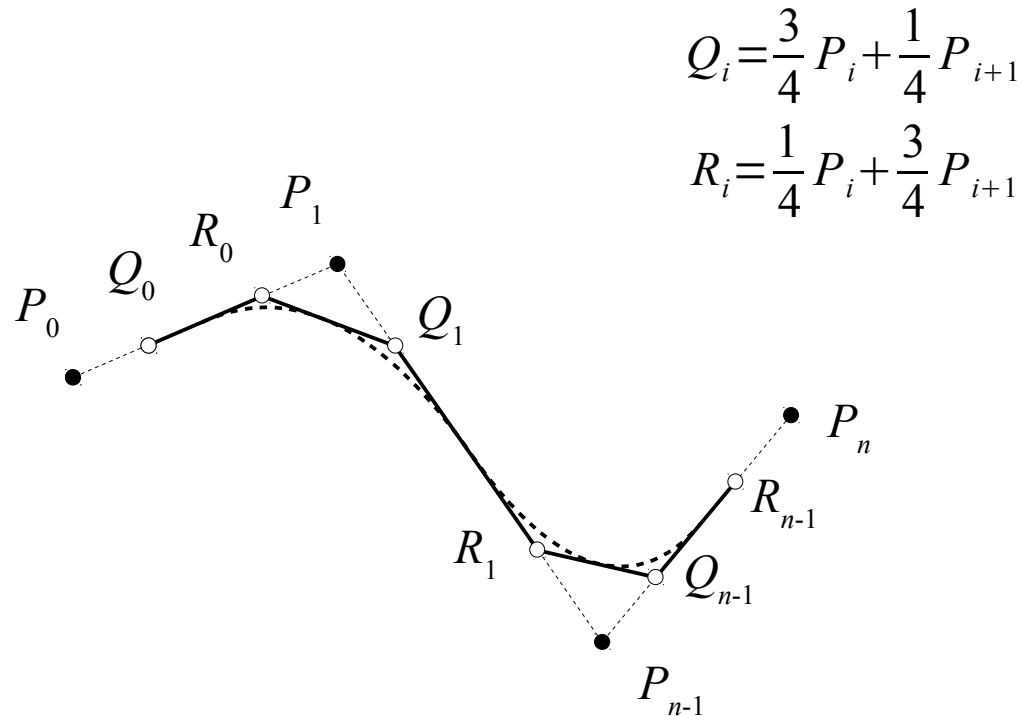
$$R_i = \frac{1}{4} P_i + \frac{3}{4} P_{i+1}$$



Subdivision surfaces

- Chaikin's scheme

- Riesenfeld (1978) has shown that this algorithm leads at the limit to an uniform quadratic B-spline, which exhibits a C^1 continuity.



Subdivision surfaces

- Demonstration of the equivalence of Chaikin's scheme and uniform quadratic B-Splines

- The B-Spline curve is defined

$$\text{by : } P(u) = \sum_{i=0}^n P_i N_i^2(u)$$

$$N_i^2(u) = \frac{u - u_i}{u_{i+2} - u_i} N_i^1(u) + \frac{u_{i+3} - u}{u_{i+3} - u_{i+1}} N_{i+1}^1(u)$$

$$N_i^1(u) = \frac{u - u_i}{u_{i+1} - u_i} N_i^0(u) + \frac{u_{i+2} - u}{u_{i+2} - u_{i+1}} N_{i+1}^0(u)$$

$$N_i^0(u) = \begin{cases} 1 & \text{if } u_i \leq u < u_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$U = \{u_0, \dots, u_{n+2}\}, \quad u_{i+1} - u_i = 1, \quad i = 0 \dots n+1$$

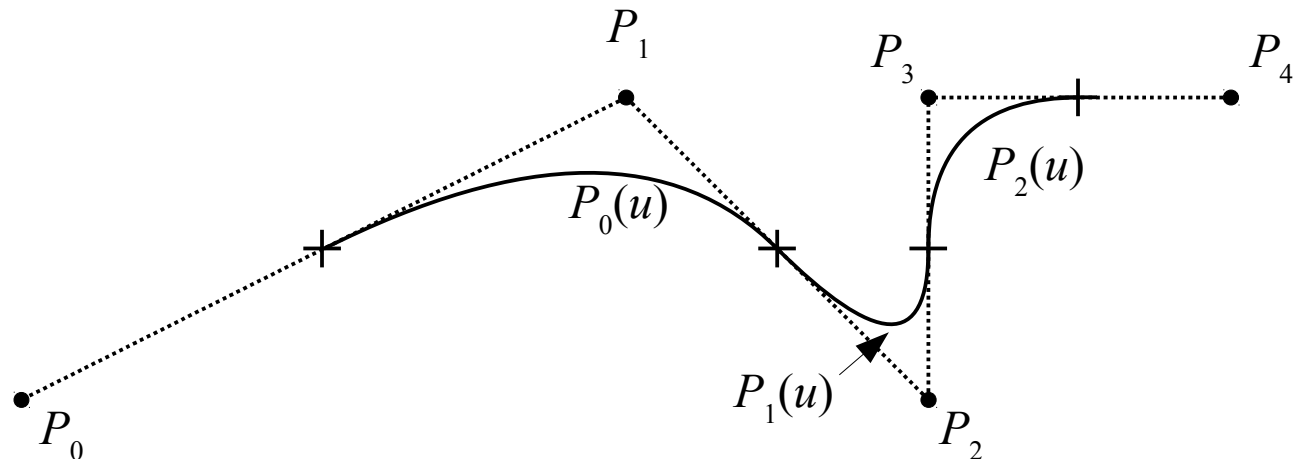
Subdivision surfaces

- It can be rewritten as a “monomial” form :

$$P(u) = \sum_{i=0}^n P_i N_i^2(u) = \sum_{k=0}^{n-2} P_k(u) \quad \leftarrow \text{« Portion » of curve}$$

$$\text{with } P_k(u) = [1 \quad u \quad u^2] \cdot M_k \cdot \begin{bmatrix} P_k \\ P_{k+1} \\ P_{k+2} \end{bmatrix}$$

- The matrix M_k depends on the nodal sequence U .



Subdivision surfaces

- Computation of shape functions of degree $d \leq 2$ for $u_2 = 0 \leq u \leq u_3 = 1$

$$U = \{u_0 = -2, u_1 = -1, u_2 = 0, u_3 = 1, u_4 = 2, u_5 = 3\}$$

$$\begin{array}{l}
 N_0^0 = 0 \\
 N_1^0 = 0 \\
 N_2^0 = 1 \\
 N_3^0 = 0 \\
 N_4^0 = 0
 \end{array}
 \begin{array}{l}
 \longrightarrow \\
 \longrightarrow \\
 \longrightarrow \\
 \longrightarrow \\
 \longrightarrow
 \end{array}
 \begin{array}{l}
 N_0^1 = 0 \\
 N_1^1 = 1 - u \\
 N_2^1 = u \\
 N_3^1 = 0
 \end{array}
 \begin{array}{l}
 \longrightarrow \\
 \longrightarrow \\
 \longrightarrow \\
 \longrightarrow
 \end{array}
 \begin{array}{l}
 N_0^2 = \frac{1}{2}(1 - 2u + u^2) \\
 N_1^2 = \frac{1}{2}(1 + 2u - 2u^2) \\
 N_2^2 = \frac{1}{2}(u^2)
 \end{array}$$

Therefore,
$$M_0 = \frac{1}{2} \begin{bmatrix} 1 & 1 & 0 \\ -2 & 2 & 0 \\ 1 & -2 & 1 \end{bmatrix}$$

General case :
$$M_k = \frac{1}{u_{k+3} - u_{k+2}} \begin{bmatrix} \frac{u_{k+3}^2}{\alpha} & -\frac{u_{k+3}u_{k+1}}{\alpha} - \frac{u_{k+4}u_{k+2}}{\beta} & \frac{u_{k+2}^2}{\beta} \\ -2\frac{u_{k+3}}{\alpha} & \frac{u_{k+3} + u_{k+1}}{\alpha} + \frac{u_{k+4} + u_{k+2}}{\beta} & -2\frac{u_{k+2}}{\beta} \\ \frac{1}{\alpha} & \frac{1}{\alpha} + \frac{1}{\beta} & \frac{1}{\beta} \end{bmatrix}$$

with :
$$\alpha = u_{k+3} - u_{k+1}$$

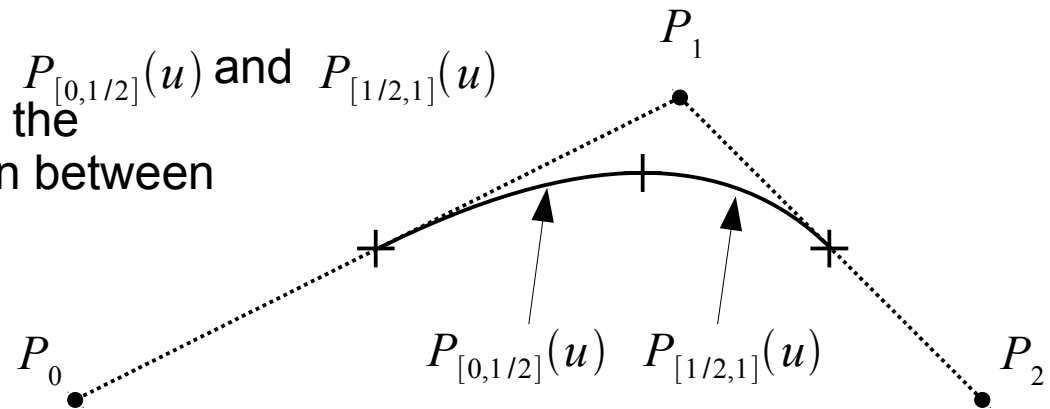
$$\beta = u_{k+4} - u_{k+2}$$

Subdivision surfaces

- Binary subdivision of a B-Spline curve for $0 \leq u \leq 1$
 - One has to find the new set of control points for each half of the curve
 - We set $n=2$ (number of control points)

$$P(u) = [1 \quad u \quad u^2] \cdot M \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix} \qquad M = \frac{1}{2} \begin{bmatrix} 1 & 1 & 0 \\ -2 & 2 & 0 \\ 1 & -2 & 1 \end{bmatrix}$$

- One wants to express $P_{[0,1/2]}(u)$ and $P_{[1/2,1]}(u)$
 - on each subdivision, the parameter u shall be in between 0 and 1.



Subdivision surfaces

- Case of $P_{[0,1/2]}(u)$

$$\begin{aligned}
 P_{[0,1/2]}(u) &= P(u/2) = [1 \quad u/2 \quad u^2/4] \cdot M \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix} \\
 &= [1 \quad u \quad u^2] \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} \cdot M \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix} \\
 &= [1 \quad u \quad u^2] \cdot M \cdot M^{-1} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} \cdot M \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix} \\
 &= [1 \quad u \quad u^2] \cdot M \cdot \begin{bmatrix} Q_0 \\ Q_1 \\ Q_2 \end{bmatrix} \quad \text{avec} \quad \begin{bmatrix} Q_0 \\ Q_1 \\ Q_2 \end{bmatrix} = \underbrace{M^{-1} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} \cdot M}_{S_{[0,1/2]}} \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix}
 \end{aligned}$$

Subdivision surfaces

- Case of $P_{[1/2,1]}(u)$

$$P_{[1/2,1]}(u) = P((1+u)/2) = [1 \quad (1+u)/2 \quad (1+u)^2/4] \cdot M \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix}$$

$$= [1 \quad u \quad u^2] \cdot \begin{bmatrix} 1 & 1/2 & 1/4 \\ 0 & 1/2 & 1/2 \\ 0 & 0 & 1/4 \end{bmatrix} \cdot M \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix}$$

$$= [1 \quad u \quad u^2] \cdot M \cdot M^{-1} \cdot \begin{bmatrix} 1 & 1/2 & 1/4 \\ 0 & 1/2 & 1/2 \\ 0 & 0 & 1/4 \end{bmatrix} \cdot M \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix}$$

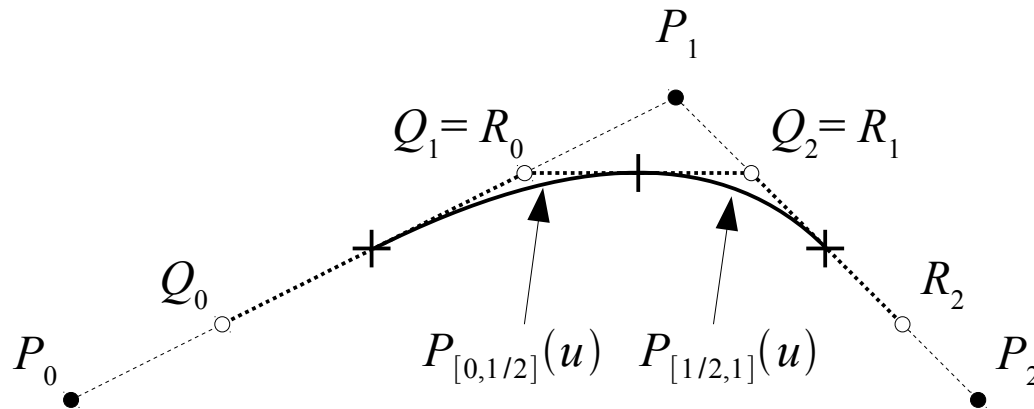
$$= [1 \quad u \quad u^2] \cdot M \cdot \begin{bmatrix} R_0 \\ R_1 \\ R_2 \end{bmatrix} \quad \text{avec} \quad \begin{bmatrix} R_0 \\ R_1 \\ R_2 \end{bmatrix} = \underbrace{M^{-1} \cdot \begin{bmatrix} 1 & 1/2 & 1/4 \\ 0 & 1/2 & 1/2 \\ 0 & 0 & 1/4 \end{bmatrix} \cdot M}_{S_{[1/2,1]}} \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix}$$

Subdivision surfaces

- Finally,

$$\begin{bmatrix} Q_0 \\ Q_1 \\ Q_2 \end{bmatrix} = S_{[0,1/2]} \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix} \quad S_{[0,1/2]} = M^{-1} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} \cdot M = \frac{1}{4} \begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 3 & 1 \end{bmatrix} \quad \begin{bmatrix} Q_0 \\ Q_1 \\ Q_2 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 3P_0 + P_1 \\ P_0 + 3P_1 \\ 3P_1 + P_2 \end{bmatrix}$$

$$\begin{bmatrix} R_0 \\ R_1 \\ R_2 \end{bmatrix} = S_{[1/2,1]} \cdot \begin{bmatrix} P_0 \\ P_1 \\ P_2 \end{bmatrix} \quad S_{[1/2,1]} = M^{-1} \cdot \begin{bmatrix} 1 & 1/2 & 1/4 \\ 0 & 1/2 & 1/2 \\ 0 & 0 & 1/4 \end{bmatrix} \cdot M = \frac{1}{4} \begin{bmatrix} 1 & 3 & 0 \\ 0 & 3 & 1 \\ 0 & 1 & 3 \end{bmatrix} \quad \begin{bmatrix} R_0 \\ R_1 \\ R_2 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} P_0 + 3P_1 \\ 3P_1 + P_2 \\ P_1 + 3P_2 \end{bmatrix}$$



One finds the same coefficients as in Chaikin's scheme... (except for the indices)

$$Q_i = \frac{3}{4} P_i + \frac{1}{4} P_{i+1}$$

$$R_i = \frac{1}{4} P_i + \frac{3}{4} P_{i+1}$$

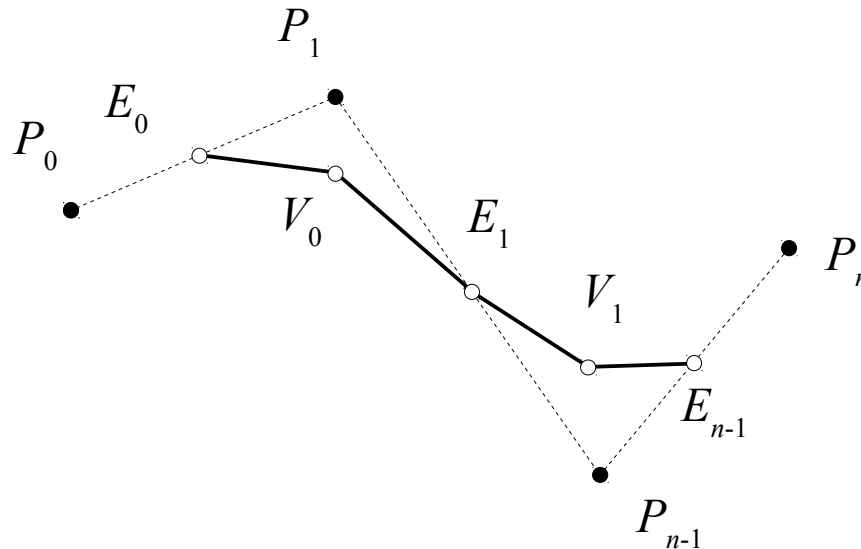
Subdivision surfaces

- This can be extended to cubic B-Splines

- C^2 continuity

$$E_i = \frac{1}{2}P_i + \frac{1}{2}P_{i+1}$$

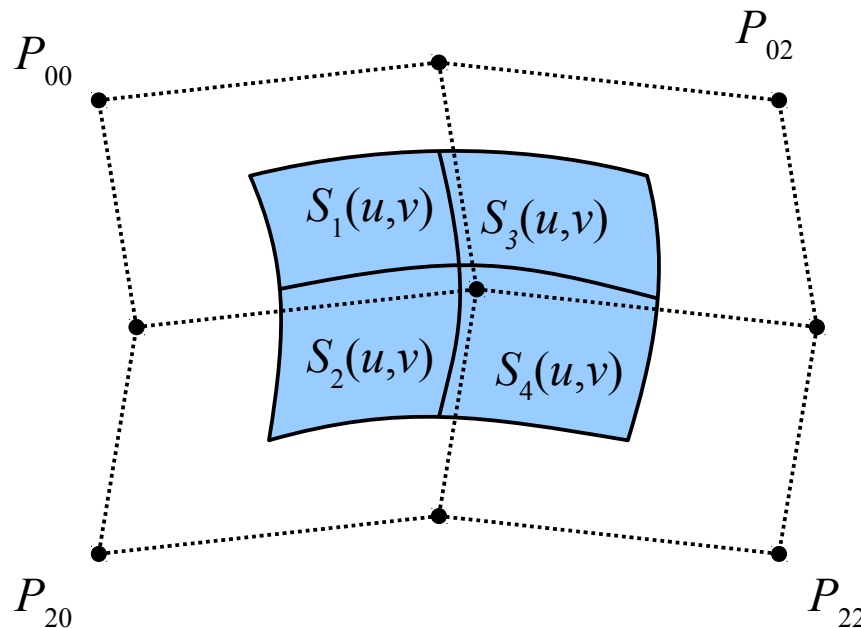
$$V_i = \frac{1}{8}P_i + \frac{3}{4}P_{i+1} + \frac{1}{8}P_{i+2}$$



Subdivision surfaces

- Doo-Sabin scheme

- This is an extension of Chaikin's scheme for a uniform biquadratic B-Spline surface
- The new mesh is built using the control points resulting from the subdivision of the original patch into 4 new sub-patches.



Subdivision surfaces

- Expression of the bi-quadratic patch as a monomial form for $0 \leq u \leq 1$ and $0 \leq v \leq 1$:

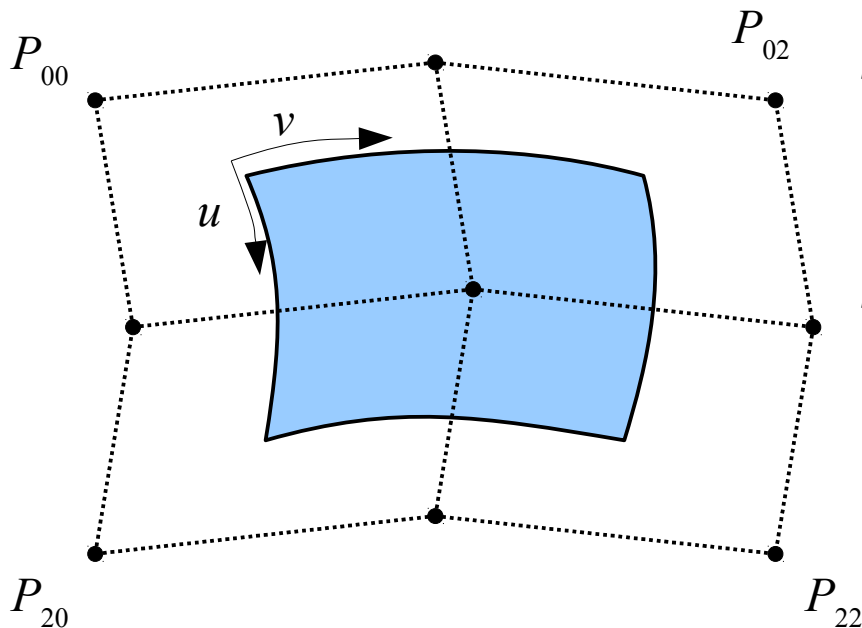
$$S(u, v) = \sum_{i=0}^2 \sum_{j=0}^2 N_i^2(u) N_j^2(v) P_{ij}$$

$$U = V = \{-2, -1, 0, 1, 2, 3\}$$

$$N_0^2(t) = \frac{1}{2}(1 - 2t + t^2)$$

$$N_1^2(t) = \frac{1}{2}(1 + 2t - 2t^2) \quad (\text{with } t = u \text{ or } v)$$

$$N_2^2(t) = \frac{1}{2}(t^2)$$



$$S(u, v) = [1 \quad u \quad u^2] \cdot M \cdot \begin{bmatrix} P_0(v) \\ P_1(v) \\ P_2(v) \end{bmatrix}$$

$$S(u, v) = [1 \quad u \quad u^2] \cdot M \cdot \begin{bmatrix} P_{00} & P_{01} & P_{02} \\ P_{10} & P_{11} & P_{12} \\ P_{20} & P_{21} & P_{22} \end{bmatrix} \cdot M^T \cdot \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix}$$

$$\text{Again, } M = \frac{1}{2} \begin{bmatrix} 1 & 1 & 0 \\ -2 & 2 & 0 \\ 1 & -2 & 1 \end{bmatrix}$$

Subdivision surfaces

- Subdivision - patch $S_1(u, v)$

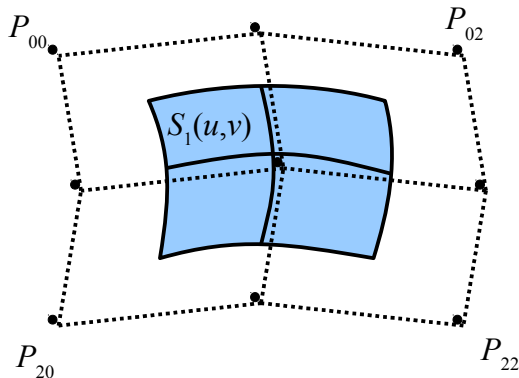
$$S_1(u, v) = S(u/2, v/2) = [1 \quad u/2 \quad u^2/4] \cdot M \cdot P \cdot M^T \cdot \begin{bmatrix} 1 \\ v/2 \\ v^2/4 \end{bmatrix} \quad P = \begin{bmatrix} P_{00} & P_{01} & P_{02} \\ P_{10} & P_{11} & P_{12} \\ P_{20} & P_{21} & P_{22} \end{bmatrix}$$

$$S_1(u, v) = S(u/2, v/2) = [1 \quad u \quad u^2] \cdot C \cdot M \cdot P \cdot M^T \cdot C^T \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix}$$

$$= [1 \quad u \quad u^2] \cdot M \cdot M^{-1} C \cdot M \cdot P \cdot M^T \cdot C^T \cdot (M^{-1})^T \cdot M^T \cdot \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix}$$

$$= [1 \quad u \quad u^2] \cdot M \cdot (M^{-1} C \cdot M) \cdot P \cdot (M^{-1} \cdot C \cdot M)^T \cdot M^T \cdot \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix}$$

$$= [1 \quad u \quad u^2] \cdot M \cdot P' \cdot M^T \cdot \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix} \quad P' = S \cdot P \cdot S^T \quad S = M^{-1} C \cdot M$$

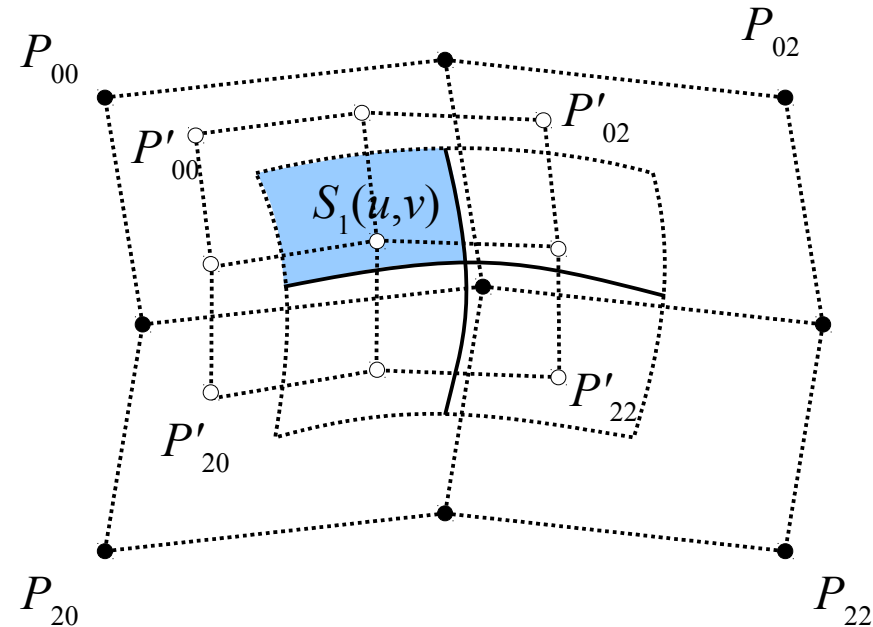


Subdivision surfaces

- Finally,

$$P' = S \cdot P \cdot S^T \quad S = M^{-1} \cdot C \cdot M = \frac{1}{4} \begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 3 & 1 \end{bmatrix}$$

$$P' = \frac{1}{16} \begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} P_{00} & P_{01} & P_{02} \\ P_{10} & P_{11} & P_{12} \\ P_{20} & P_{21} & P_{22} \end{bmatrix} \cdot \begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 3 \\ 0 & 0 & 1 \end{bmatrix}$$



$$P' = \frac{1}{16} \begin{bmatrix} 3(3P_{00} + P_{10}) + 3P_{01} + P_{11} & 3P_{00} + P_{10} + 3(3P_{01} + P_{11}) & 3(3P_{01} + P_{11}) + 3P_{02} + P_{12} \\ 3(P_{00} + 3P_{10}) + P_{01} + 3P_{11} & P_{00} + 3P_{10} + 3(P_{01} + 3P_{11}) & 3(P_{01} + 3P_{11}) + P_{02} + 3P_{12} \\ 3(3P_{10} + P_{20}) + 3P_{11} + P_{21} & 3P_{10} + P_{20} + 3(3P_{11} + P_{21}) & 3(3P_{11} + P_{21}) + 3P_{12} + P_{22} \end{bmatrix}$$

- Same developments should be done with the 3 other quadrants, and will lead to the same “structure”

Subdivision surfaces

- Subdivision - patch $S_2(u, v)$

$$S_2(u, v) = S(u/2, (1+v)/2) = [1 \quad u/2 \quad u^2/4] \cdot M \cdot P \cdot M^T \cdot \begin{bmatrix} 1 \\ (1+v)/2 \\ (1+v)^2/4 \end{bmatrix} \quad P = \begin{bmatrix} P_{00} & P_{01} & P_{02} \\ P_{10} & P_{11} & P_{12} \\ P_{20} & P_{21} & P_{22} \end{bmatrix}$$

$$S_2(u, v) = S(u/2, (1+v)/2) = [1 \quad u \quad u^2] \cdot C_u \cdot M \cdot P \cdot M^T \cdot C_v^T \cdot \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix}$$

$$C_u = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1/4 \end{bmatrix} \quad C_v = \begin{bmatrix} 1 & 1/2 & 1/4 \\ 0 & 1/2 & 1/2 \\ 0 & 0 & 1/4 \end{bmatrix}$$

$$= [1 \quad u \quad u^2] \cdot M \cdot (M^{-1} C_u \cdot M) \cdot P \cdot (M^{-1} \cdot C_v \cdot M)^T \cdot M^T \cdot \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix}$$

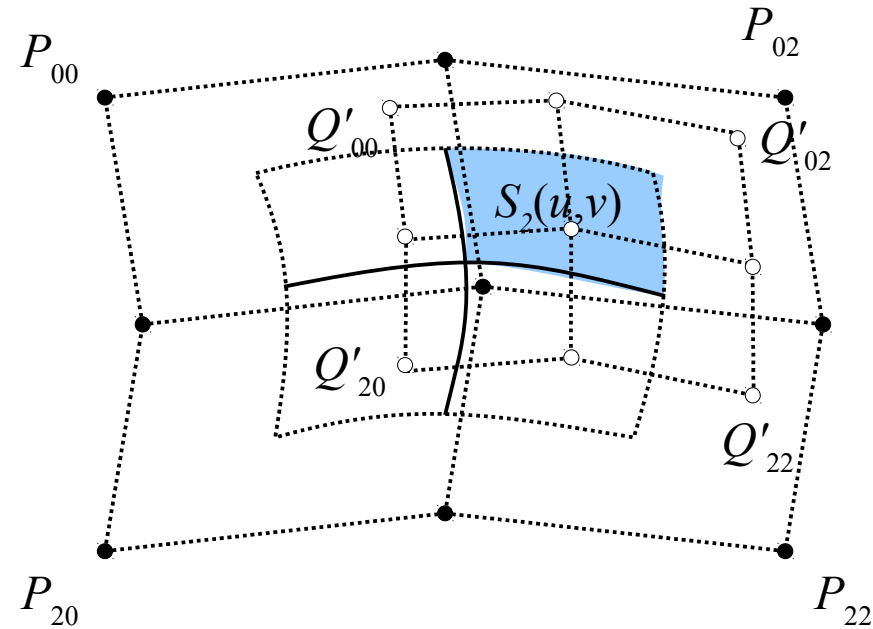
$$= [1 \quad u \quad u^2] \cdot M \cdot Q' \cdot M^T \cdot \begin{bmatrix} 1 \\ v \\ v^2 \end{bmatrix} \quad Q' = S_u \cdot P \cdot S_v^T \quad \begin{aligned} S_u &= M^{-1} C_u \cdot M \\ S_v &= M^{-1} C_v \cdot M \end{aligned}$$

Subdivision surfaces

$$Q' = S_u \cdot P S_v^T \quad S_u = M^{-1} \cdot C_u \cdot M = \frac{1}{4} \begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 3 & 1 \end{bmatrix}$$

$$S_v = M^{-1} \cdot C_v \cdot M = \frac{1}{4} \begin{bmatrix} 1 & 3 & 0 \\ 0 & 3 & 1 \\ 0 & 1 & 3 \end{bmatrix}$$

$$Q' = \frac{1}{16} \begin{bmatrix} 3 & 1 & 0 \\ 1 & 3 & 0 \\ 0 & 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} P_{00} & P_{01} & P_{02} \\ P_{10} & P_{11} & P_{12} \\ P_{20} & P_{21} & P_{22} \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 3 & 3 & 1 \\ 0 & 1 & 3 \end{bmatrix}$$



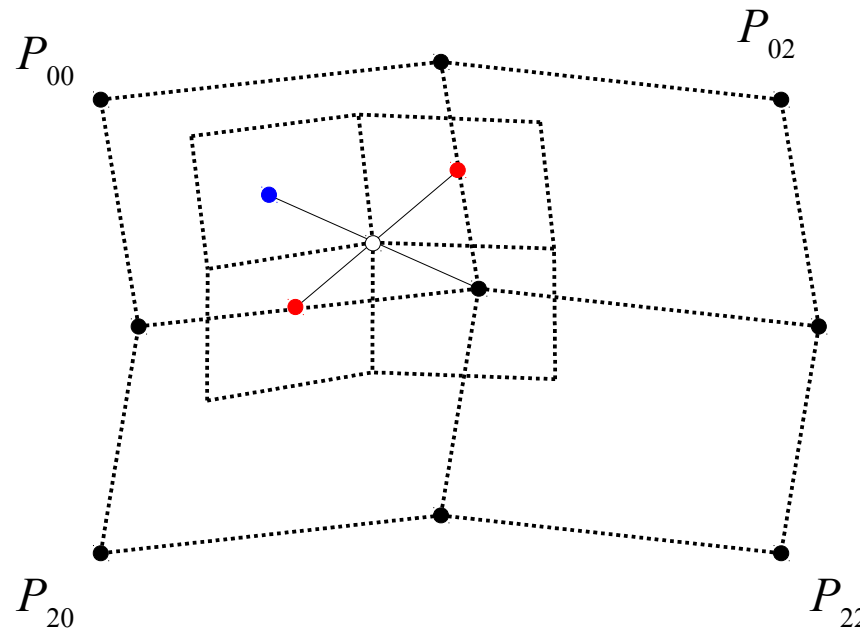
- Some of the points (2) are already computed, e.g.

$$Q'_{00} = 3(P_{00} + 3P_{01}) + P_{10} + 3P_{11}$$

$$(= P'_{01} = 3P_{00} + P_{10} + 3(3P_{01} + P_{11}))$$

Subdivision surfaces

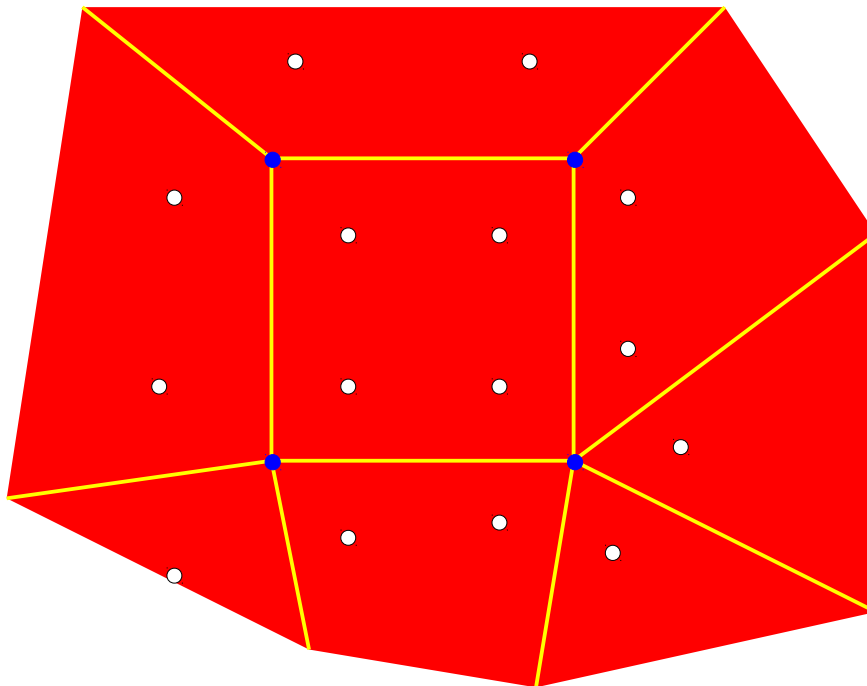
- Extension to meshes showing an arbitrary topology



- The new vertices are obtained as a simple arithmetic mean of 3 categories of vertices :
 - The vertices of the old mesh
 - Vertices on the edges (barycentre of the extremities of the edge)
 - Vertices inside a face (barycentre of the vertices of the face)

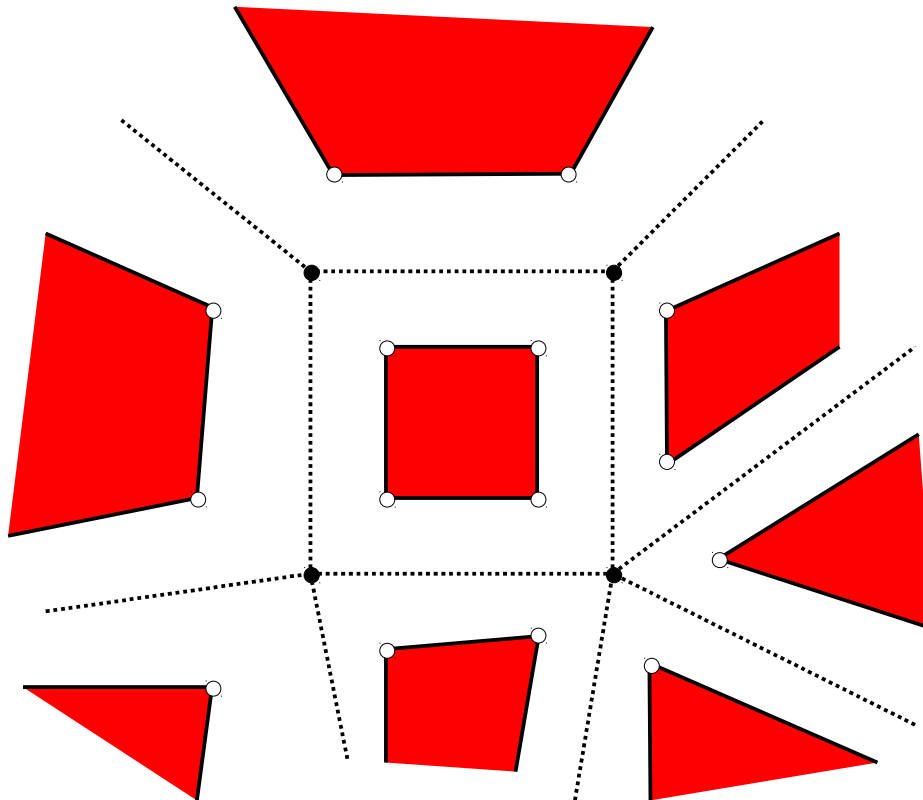
Subdivision surfaces

- Extension to meshes showing an arbitrary topology
 - 1 – Computation of the vertices P' (for each vertex P , compute the mean between P , the vertices on the adjacent faces, and the vertices on adjacent edges)



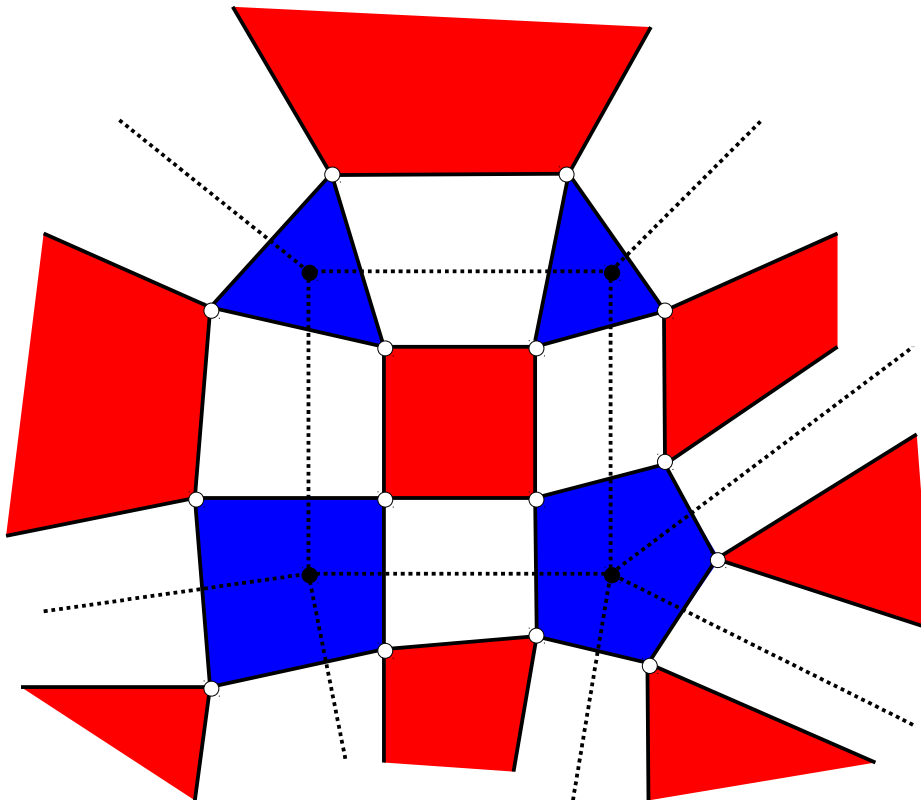
Subdivision surfaces

- Extension to meshes showing an arbitrary topology
 - 2 – For each face, link the corresponding vertices P'



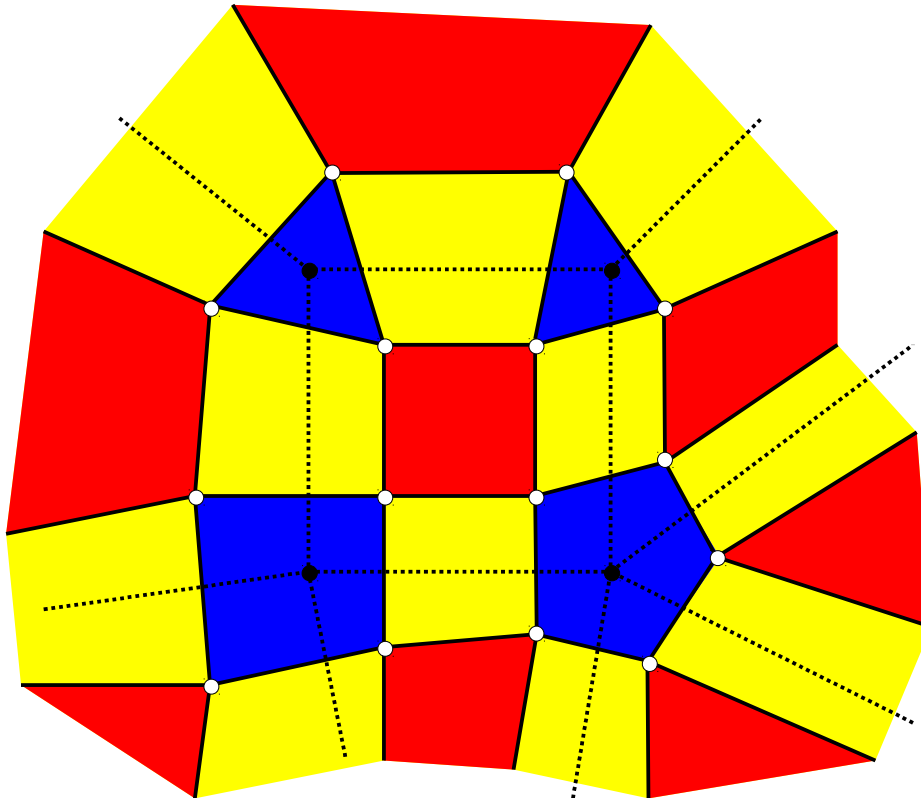
Subdivision surfaces

- Extension to meshes showing an arbitrary topology
 - 3 – For each old vertex, connect the new ones that have been created for each adjacent face to this old vertex.



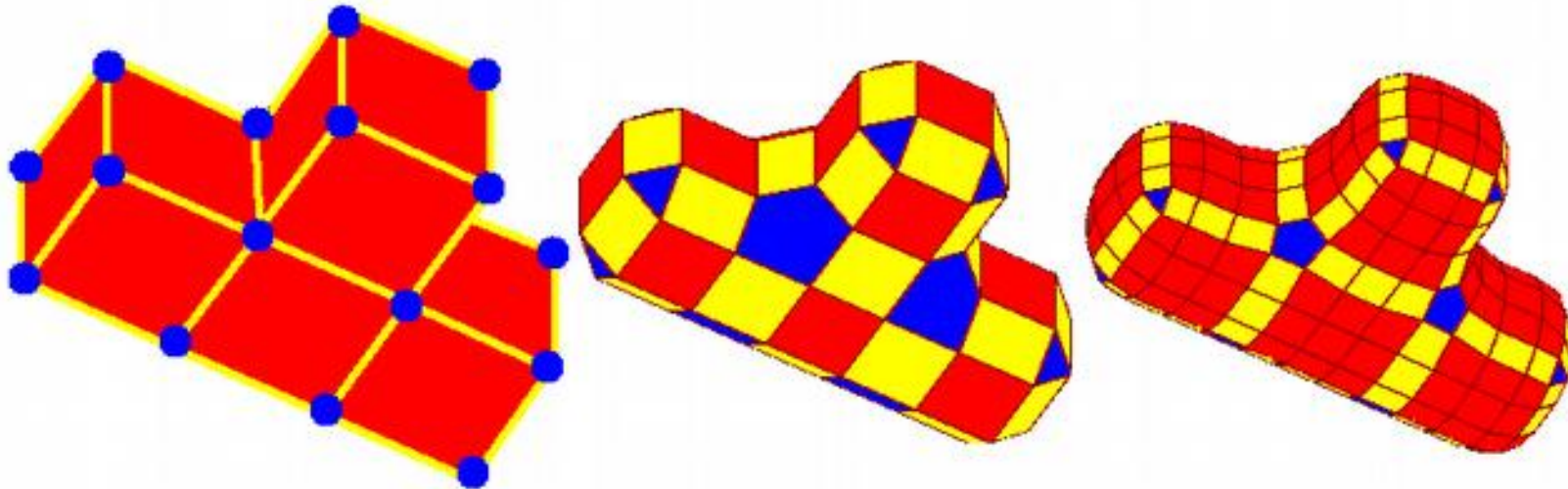
Subdivision surfaces

- Extension to meshes showing an arbitrary topology
 - 4 – For each old edge, connect the new vertices that have been created for each adjacent face to this old edge.



Subdivision surfaces

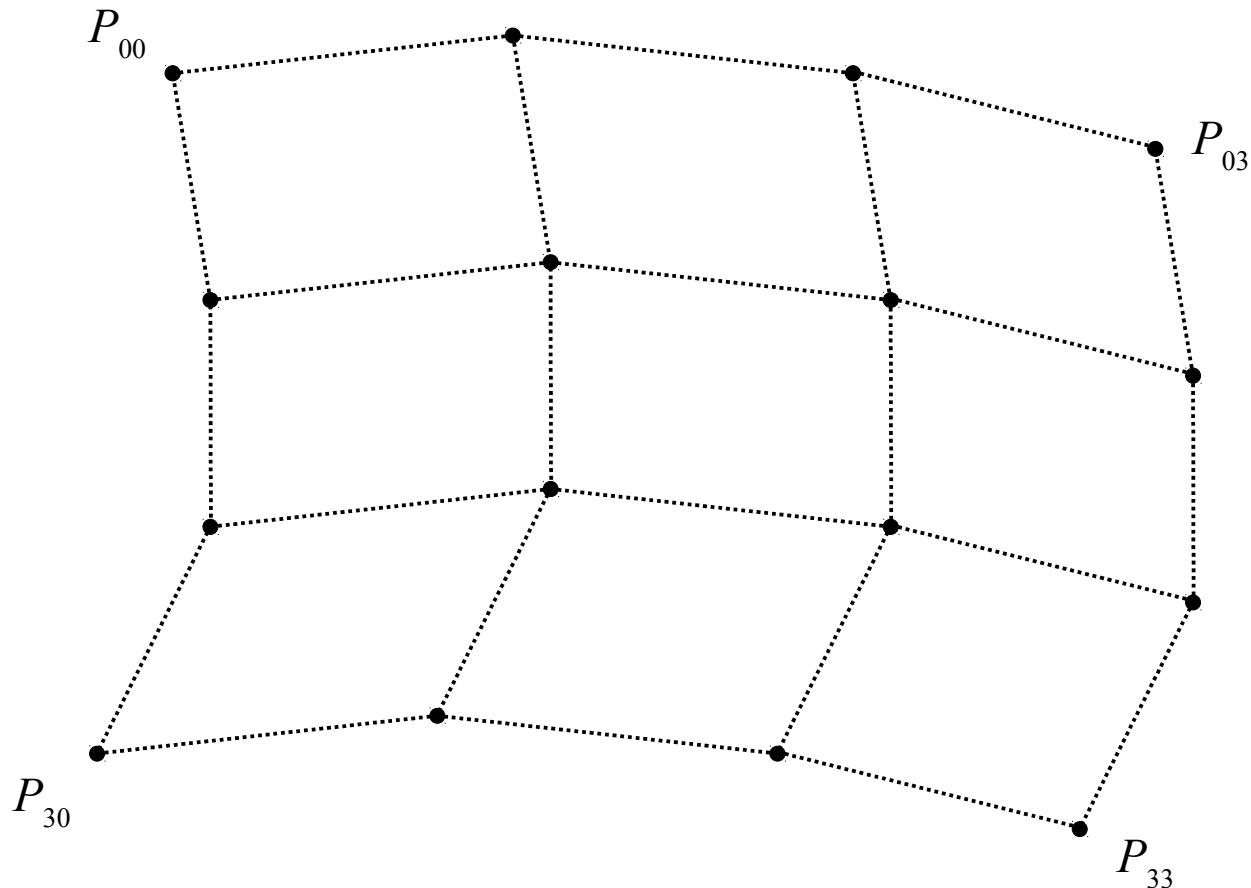
- Some points on the mesh and the limiting surface are « extraordinary »
 - These are vertices with a valence (number of incident edges) that is different from 4.



- Everywhere the continuity of the limiting surface is C^1 ; except at extraordinary points, where it decreases to C^0 .

Subdivision surfaces

- Catmull-Clark scheme
 - Similar idea for bicubic B-Splines.



Subdivision surfaces

- Three types of vertices

- « Face » vertices are at the barycentre of the vertices of that face:

$$P_f = Q$$

- « Edge » vertices are at the barycentre of the extremities of the edge and the two « Face » vertices of the adjacent faces :

$$P_e = \frac{Q + R}{2}$$

- « Corner » vertices are positioned such that :

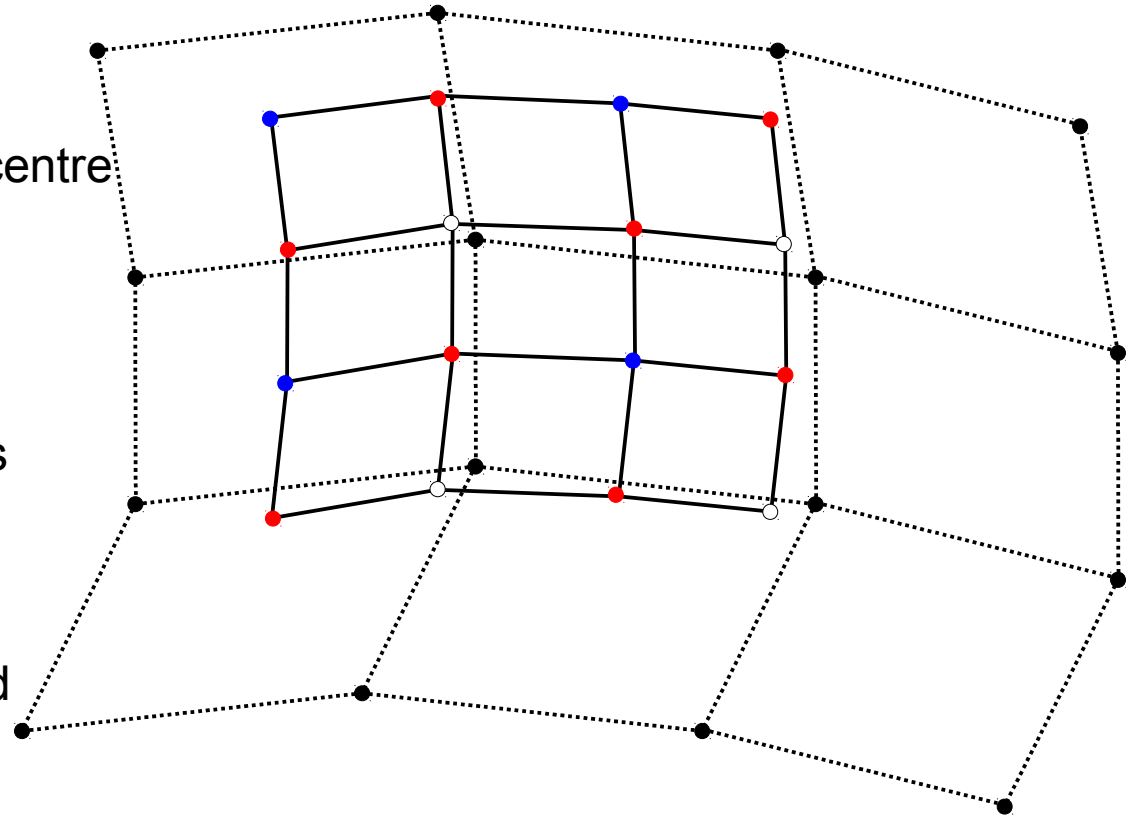
$$P_v = \frac{Q + 2R + (n - 3)S}{n}$$

Q = mean of the barycentre of the incident faces

R = mean of the barycentre of the incident edges

S = original vertex

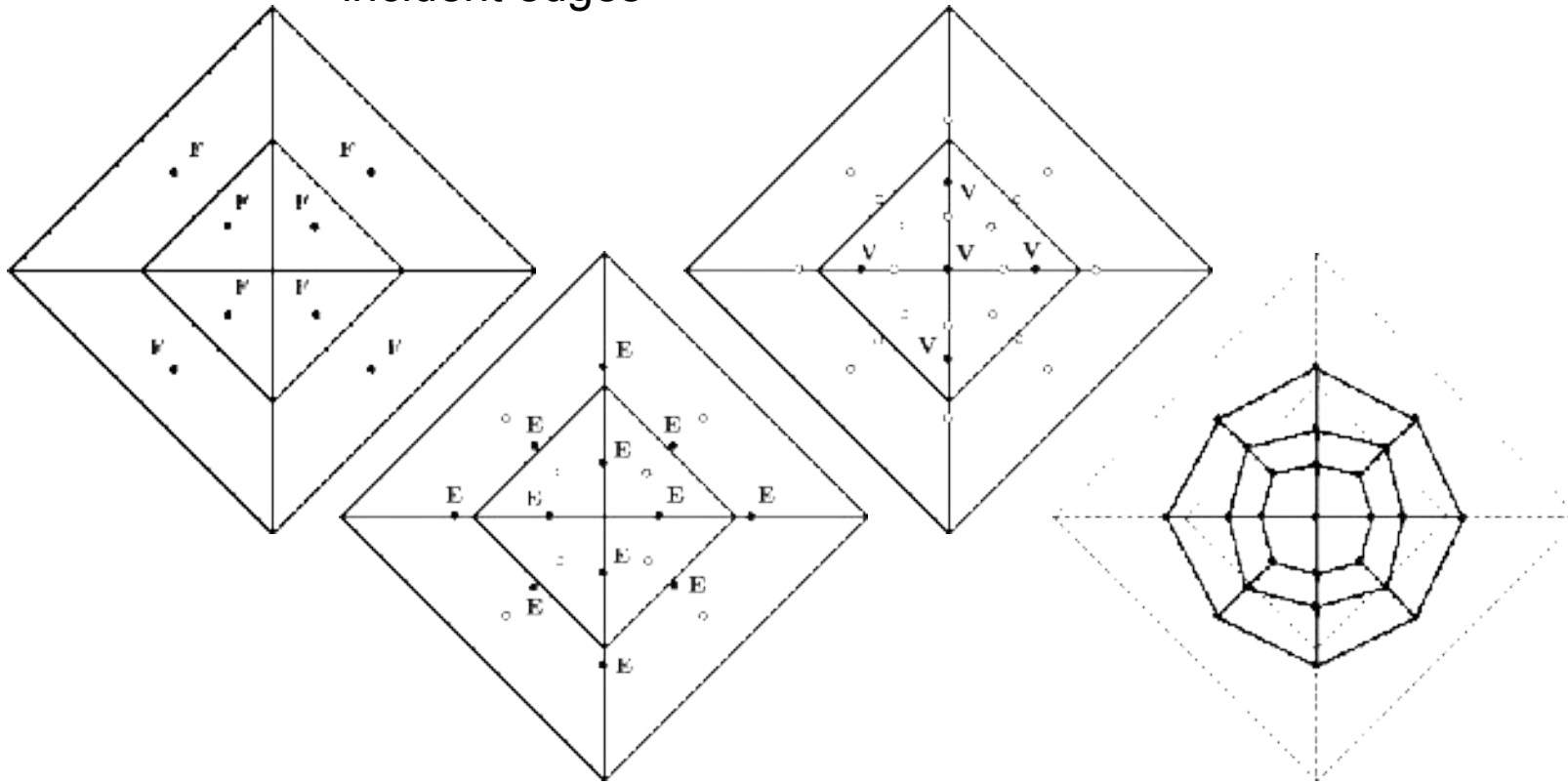
n = number of incident edges to S .



- « face » vertices
- « edge » vertices
- « corner » vertices

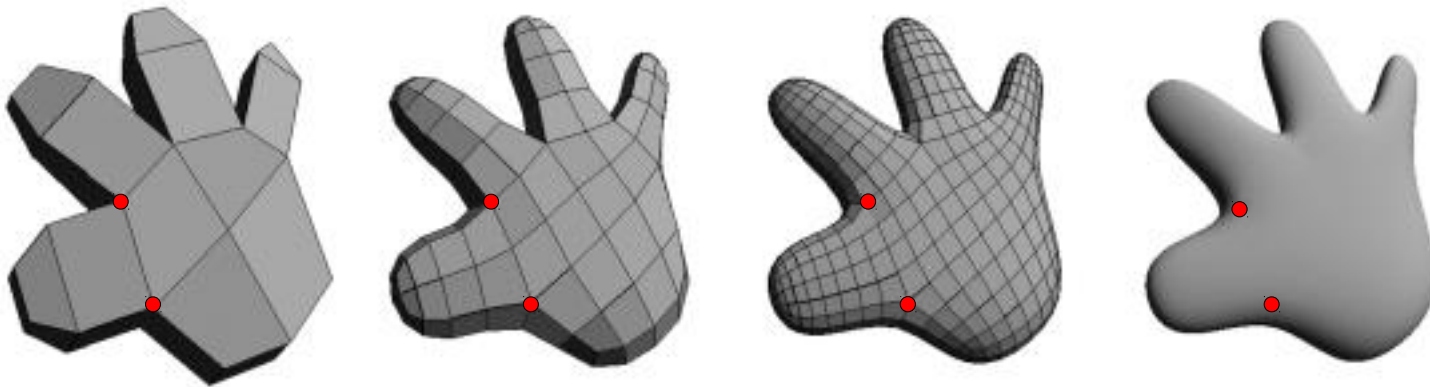
Subdivision surfaces

- Reconnecting the new vertices
 - 1 – Connect the « face » vertices to the « edge » vertices of neighbouring edges
 - 2 – Connect the « corner » vertices to the « edge » vertices of incident edges



Subdivision surfaces

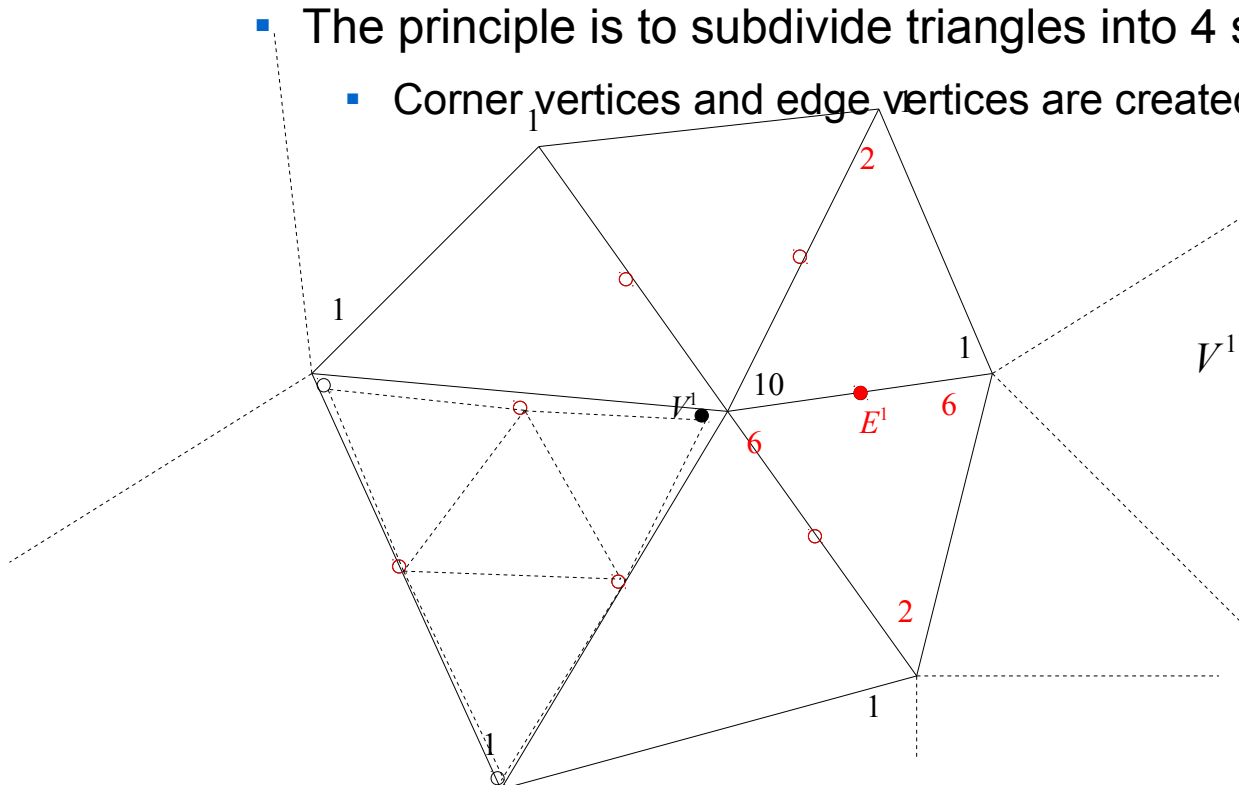
- As with Doo-Sabin surface, the continuity is degraded for some extraordinary vertices. The bicubic surfaces are therefore C^2 everywhere except at extraordinary points : it is then only C^1 .



Subdivision surfaces

- Loop's scheme

- Allows to subdivide triangular meshes
- The limiting surface is C^2 , except at extraordinary vertices of valence $\neq 6$, where it is only C^1 .
- The principle is to subdivide triangles into 4 sub-triangles.
 - Corner vertices and edge vertices are created (in red).



$$V^1 = \frac{10V + Q_1 + Q_2 + Q_3 + Q_4 + Q_5 + Q_6}{16}$$

$$= \frac{5}{8}V + \frac{3}{8}Q$$

$$E^1 = \frac{6V_1 + 6V_2 + 2F_1 + 2F_2}{16}$$

Subdivision surfaces

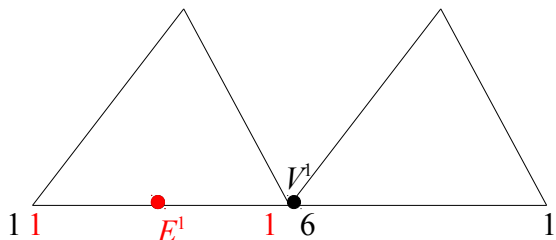
- As such, works only for vertices with a valence equal to 6
- It may be extended to other valences, but the formula has to be adapted such that the resulting surface is smooth.

- Let $V^1 = \alpha_n V + (1 - \alpha_n) Q$

with $\alpha_n = \left(\frac{3}{8} + \frac{1}{4} \cos \frac{3\pi}{n} \right)^2 + \frac{3}{8}$

n is the valence of the original vertex.

- On boundaries : vertices should not move inside the surface, they should rather slide along the boundary. One recovers the classical cubic B-Spline scheme in that case



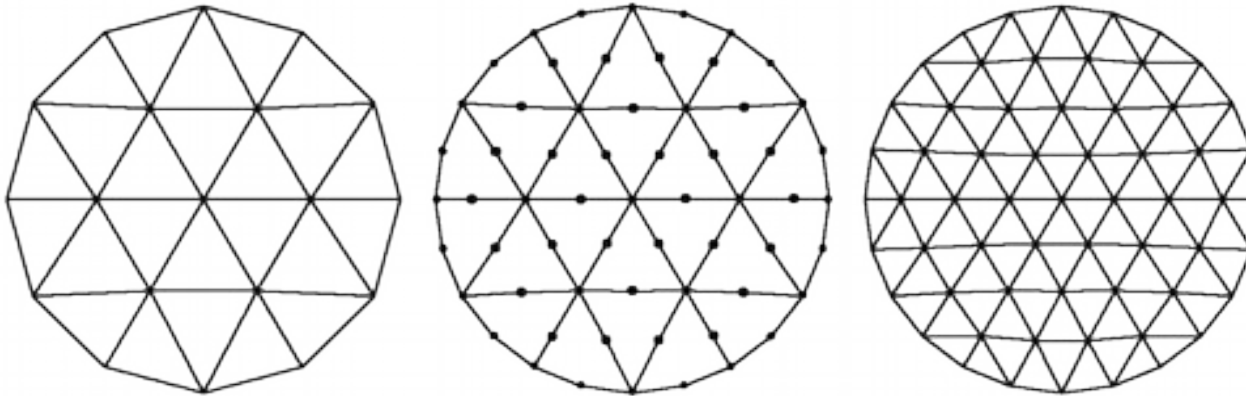
$$E^1 = \frac{V_1 + V_2}{2}$$

$$V^1 = \frac{6}{8} V + \frac{Q_1^* + Q_2^*}{8}$$

← (only if the vertices are neighbours on the boundary)

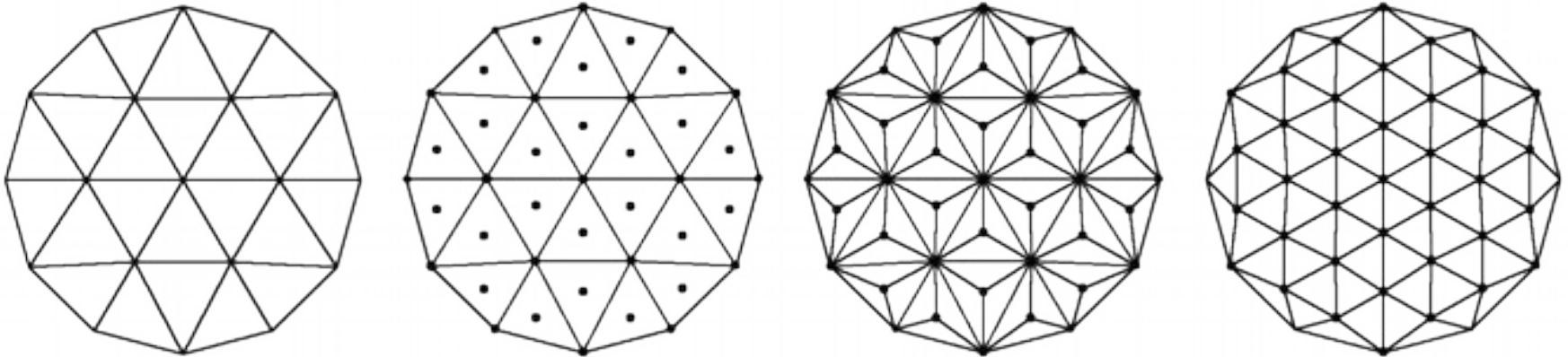
Subdivision surfaces

- Loop's scheme



Subdivision surfaces

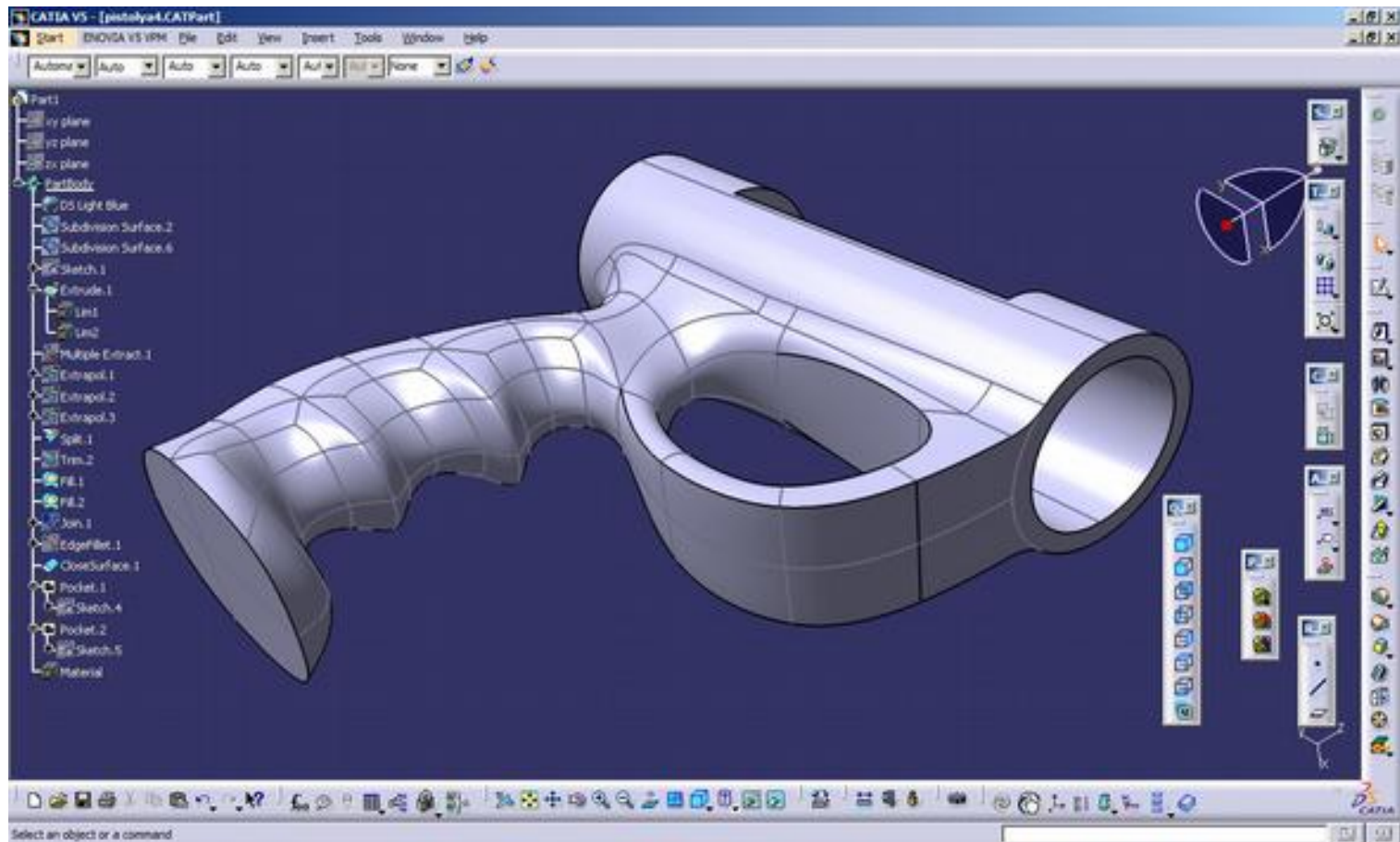
- Kobbelt's $\sqrt{3}$ scheme
 - See paper on the course's website
 - For a similar level of refinement, it generates less triangles than Loop's scheme



Subdivision surfaces

- Subdivision surfaces in CATIA

- A very easy-to-use design tool
- As S-s are equivalent to some class of B-Spline surfaces, they retain a good degree of accuracy
- “CATIA Shape” module
Imagine Shape (IMA) tool



Architectural applications

- Catia in architecture
 - Frank O. Gehry (Fish sculpture , Barcelona ,1992)



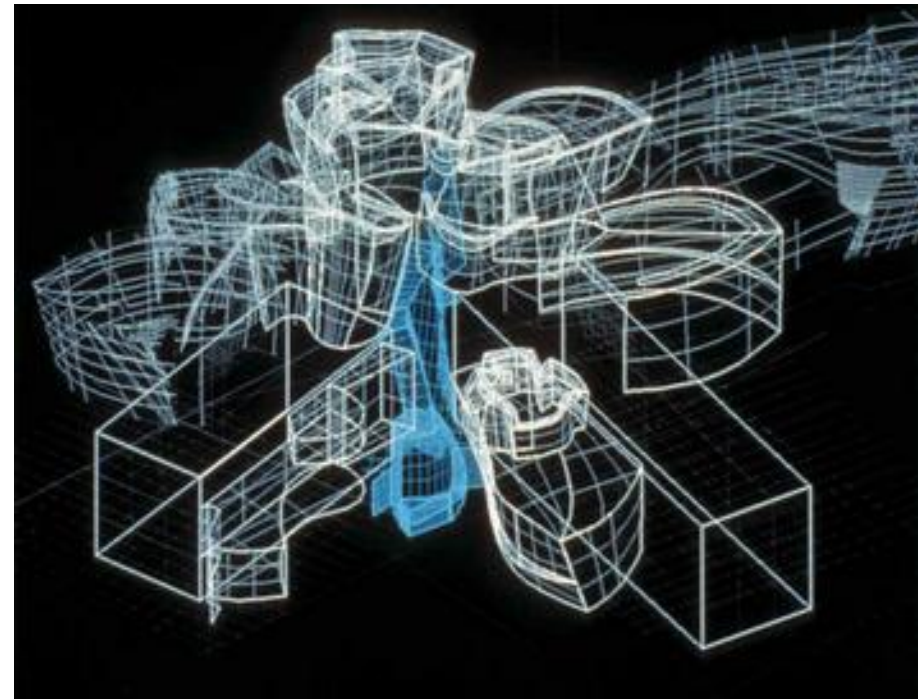
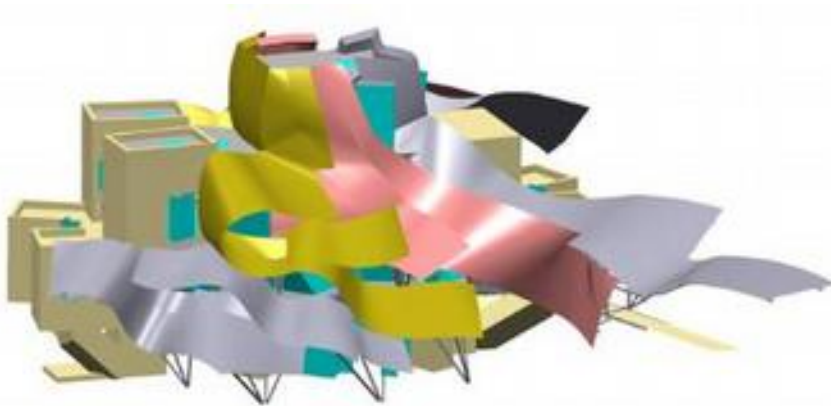
Architectural applications

- Catia in architecture
 - Frank O. Gehry (Guggenheim museum, Bilbao ,1997)



Architectural applications

- Catia in architecture
 - Frank O. Gehry (Guggenheim museum, Bilbao ,1997)



Architectural applications

- Catia in architecture
 - Frank O. Gehry (Walt Disney concert hall, Los Angeles ,2003)



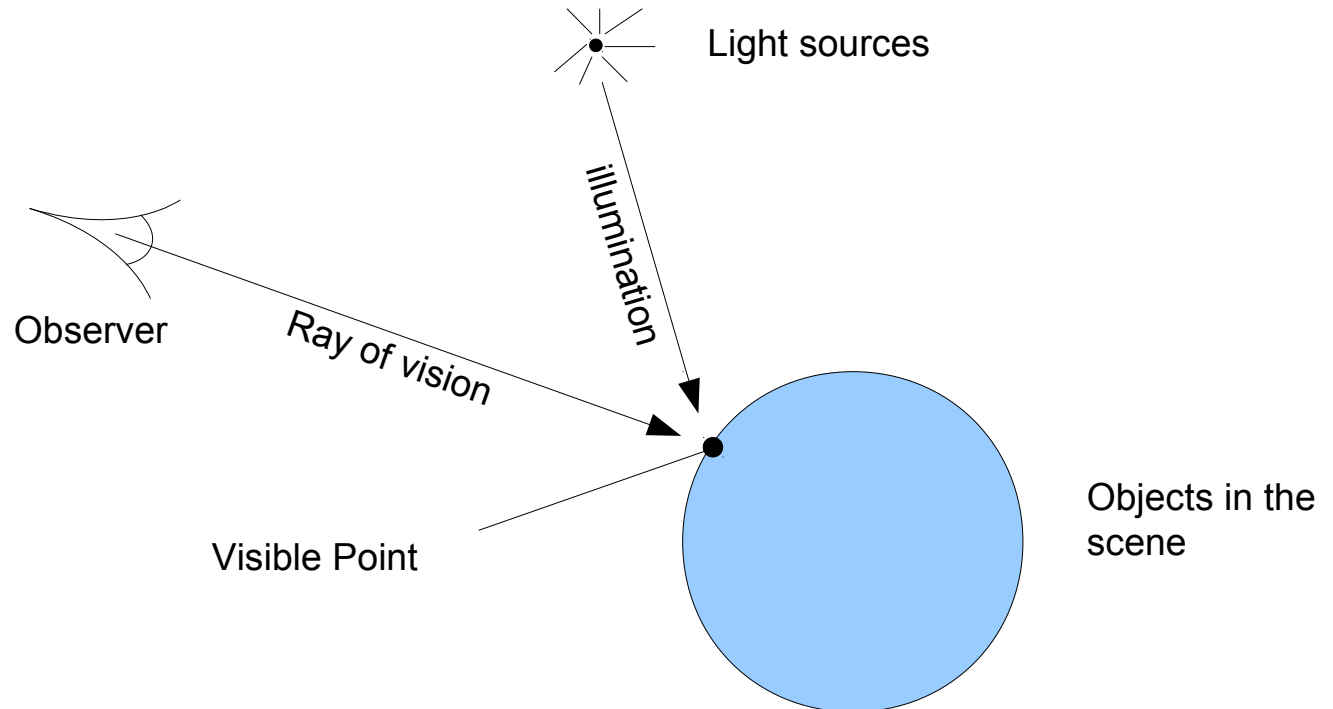
Carol M. Highsmith

Ray tracing

Ray tracing

- Ray tracing principles

- Intersection of a ray shot from the eye with the objects in the scene

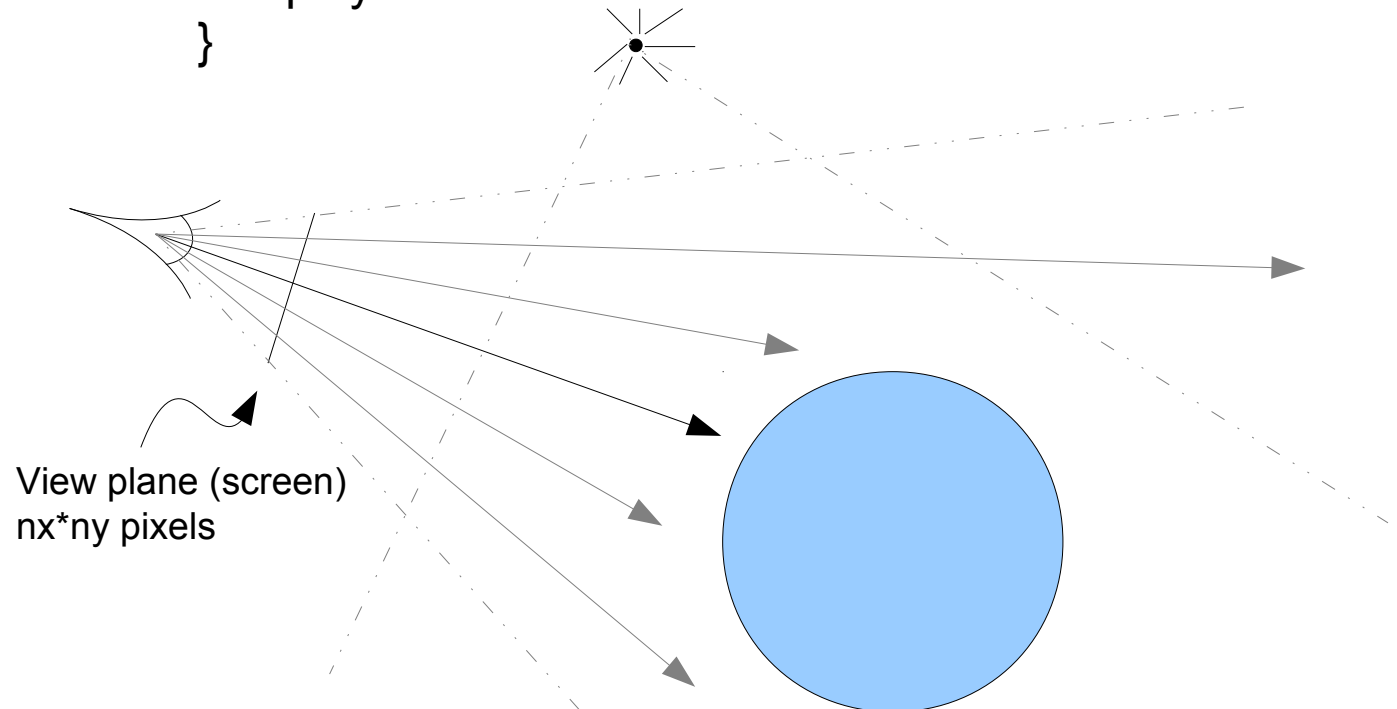


Ray tracing

- Algorithm

For every pixel

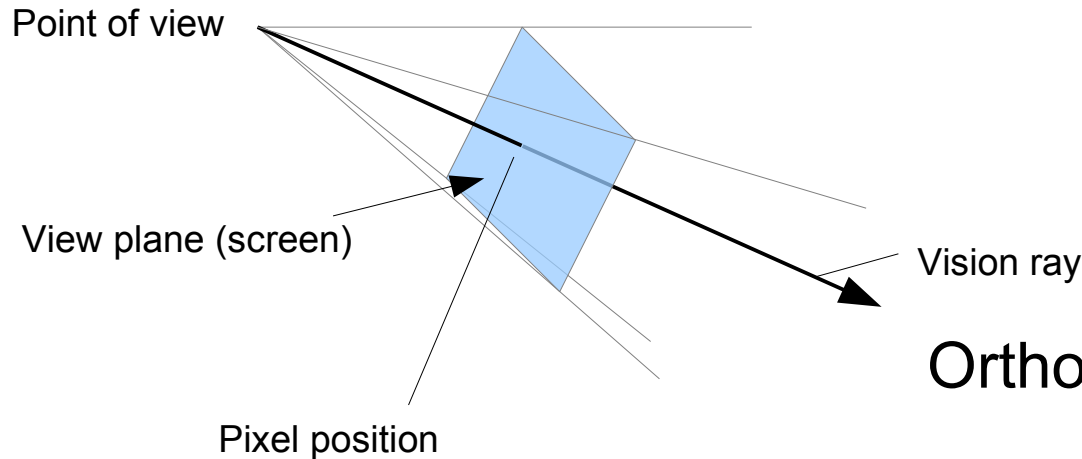
```
{  
  calculate the ray of vision  
  intersect the ray with the scene  
  calculate the illumination of the visible point  
  display the colour that is obtained  
}
```



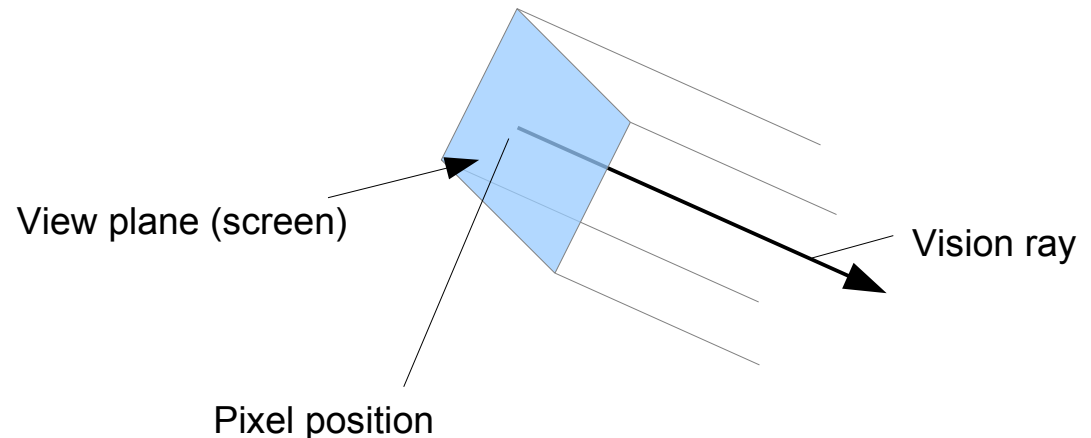
Ray tracing

- Ray calculation

Perspective projection

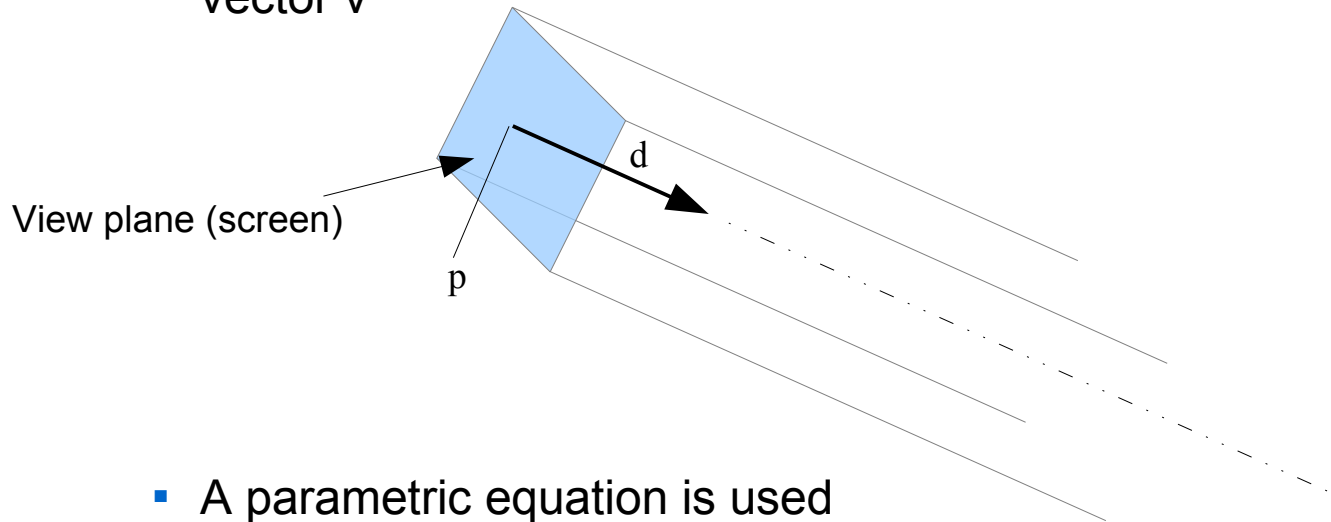


Orthographic projection



Ray tracing

- Ray calculation – orthographic projection
 - We shall calculate the position \mathbf{p} in the plane of the screen, and the vector \mathbf{v}



- A parametric equation is used
 - $\mathbf{r}(t) = \mathbf{p} + t \mathbf{d}$
 - But where is the screen in 3D space?

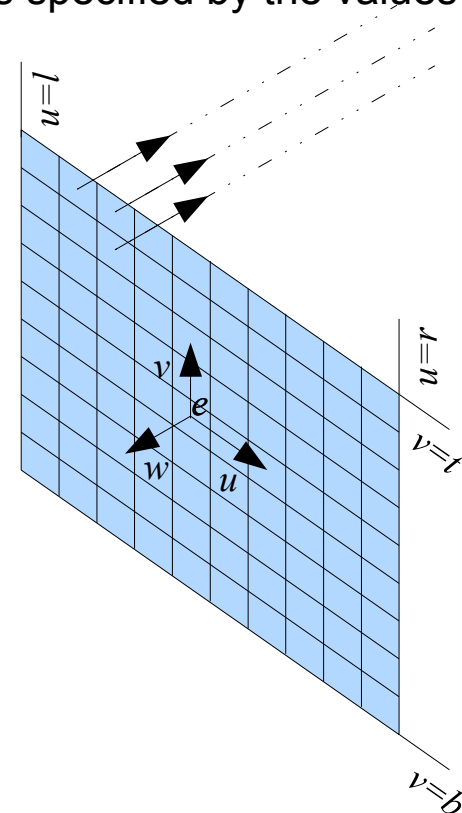
Ray tracing

- Ray calculation – orthographic projection
 - Determination of the view plane
 - A reference linked to the camera is established : (e, u, v, w)
 - The view plane is in the u - v plane ; that is specified by the values l, r, t, b (see course on homogeneous coord.)
 - The ray is then expressed as a function of the (u, v) coordinates

$$\mathbf{p} = \mathbf{e} + u \mathbf{u} + v \mathbf{v}$$

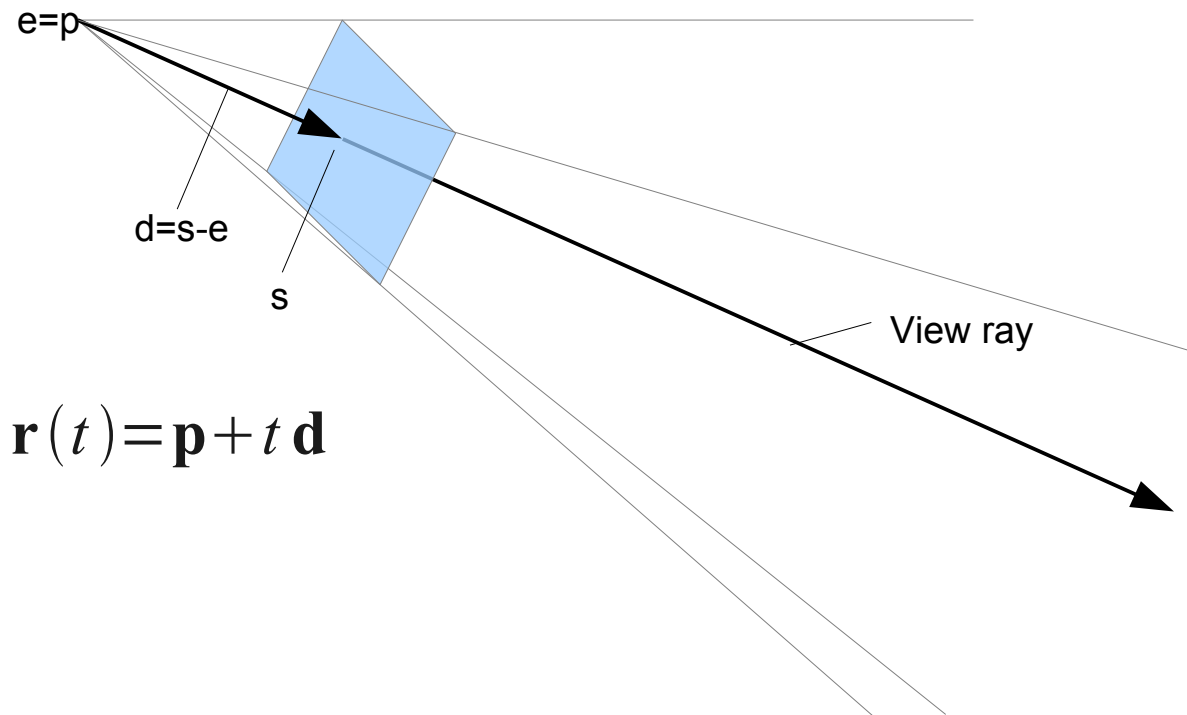
$$\mathbf{d} = -\mathbf{w}$$

$$\mathbf{r}(t) = \mathbf{p} + t \mathbf{d}$$



Ray tracing

- Ray calculation – perspective projection



Ray tracing

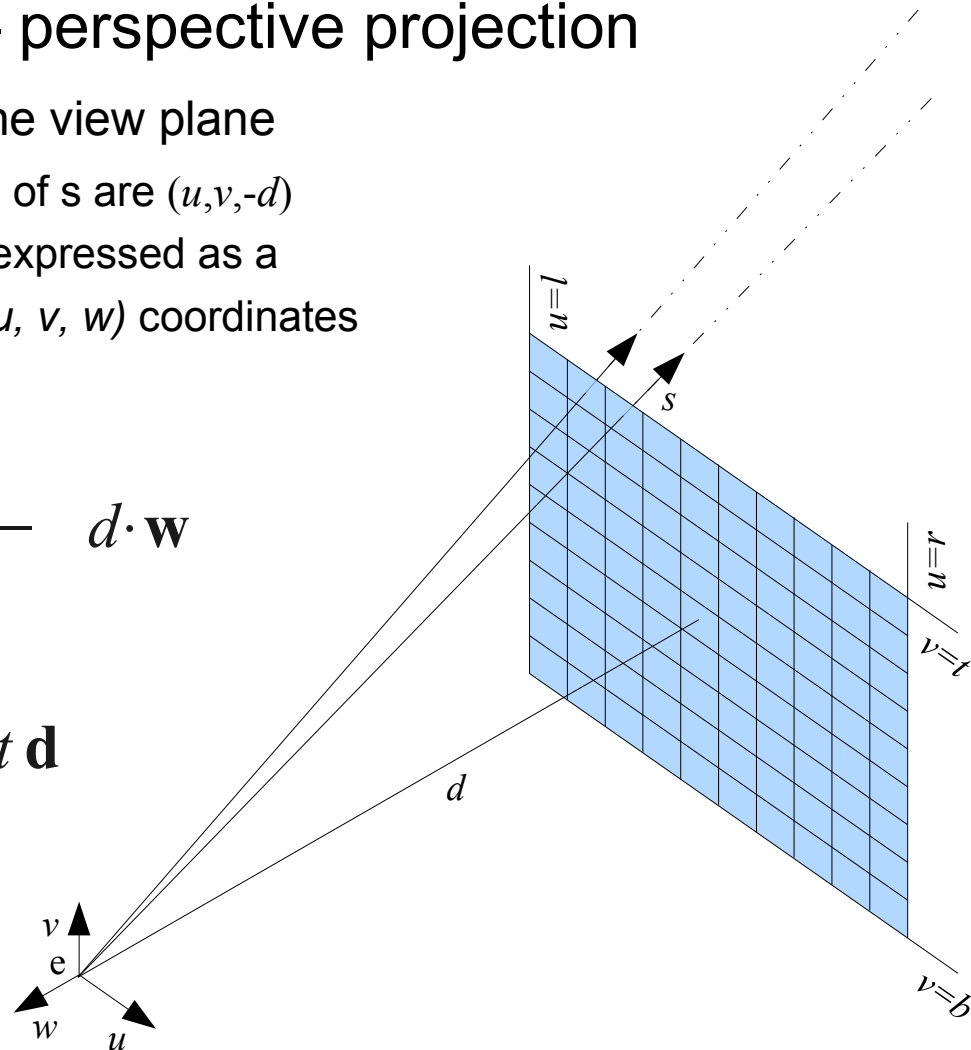
- Ray calculation – perspective projection
 - Determination of the view plane
 - The coordinates of s are $(u, v, -d)$
 - The ray is then expressed as a function of the (u, v, w) coordinates of the pixel

$$\mathbf{s} = \mathbf{e} + u \mathbf{u} + v \mathbf{v} - d \cdot \mathbf{w}$$

$$\mathbf{p} = \mathbf{e}$$

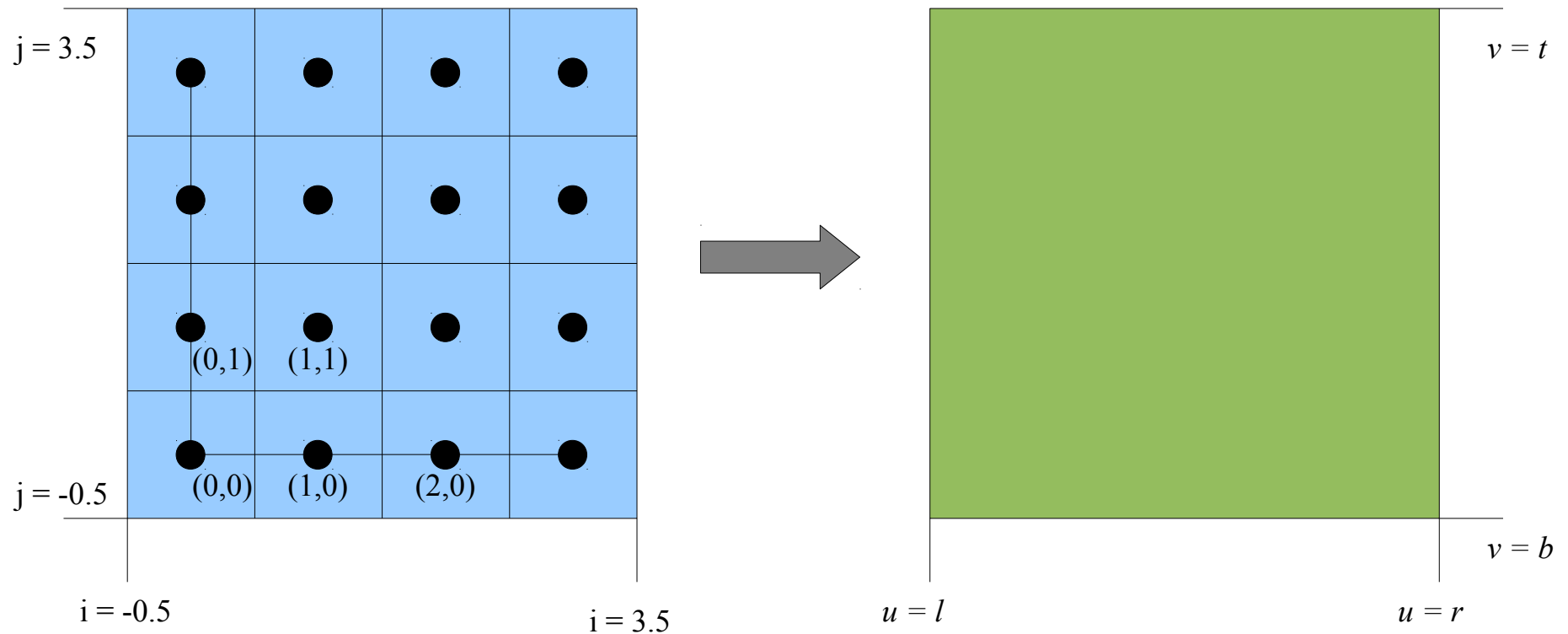
$$\mathbf{d} = \mathbf{s} - \mathbf{e}$$

$$\mathbf{r}(t) = \mathbf{p} + t \mathbf{d}$$



Ray tracing

- From the pixel to the picture

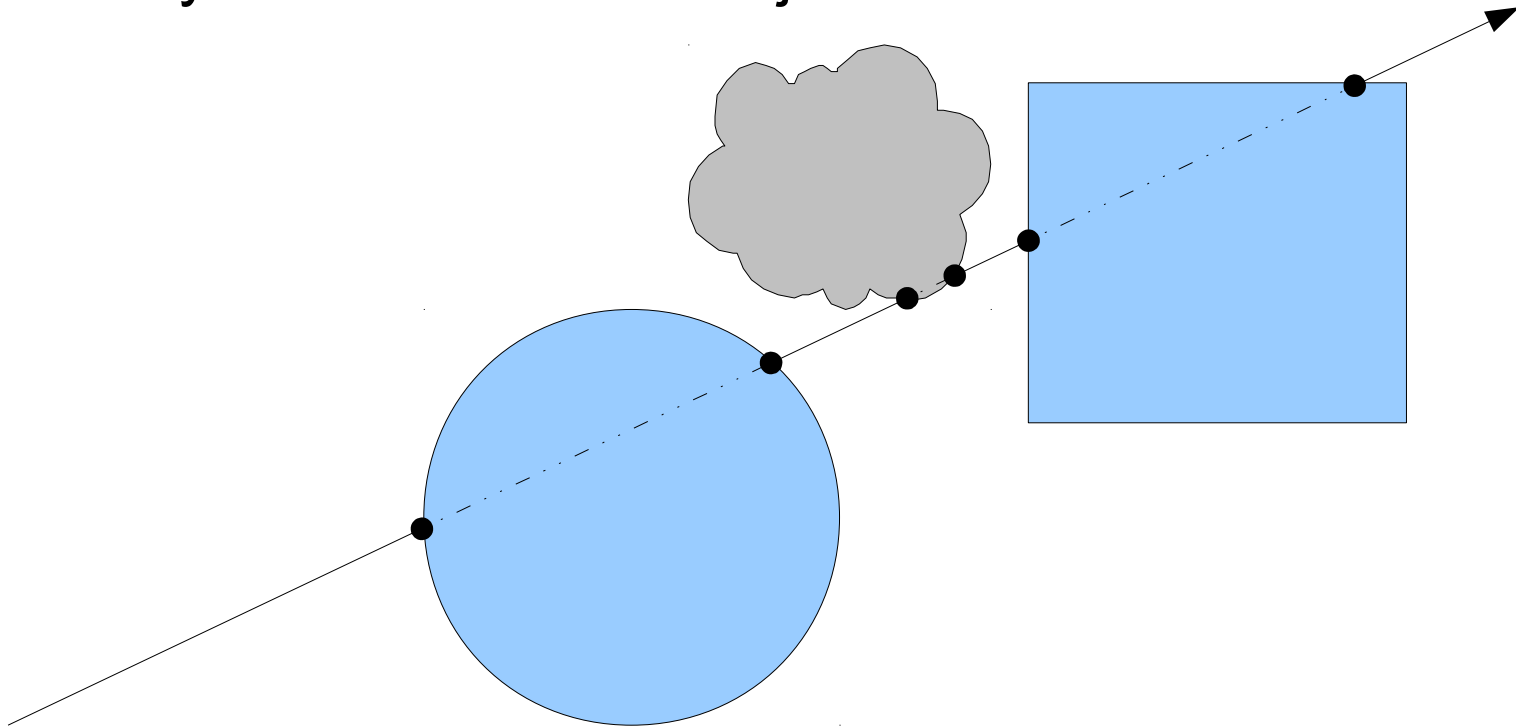


$$u = l + \frac{(r-l)(i+0.5)}{n_x}$$

$$v = b + \frac{(t-b)(j+0.5)}{n_y}$$

Ray tracing

- Intersection calculation
 - Rays intersect scene objects

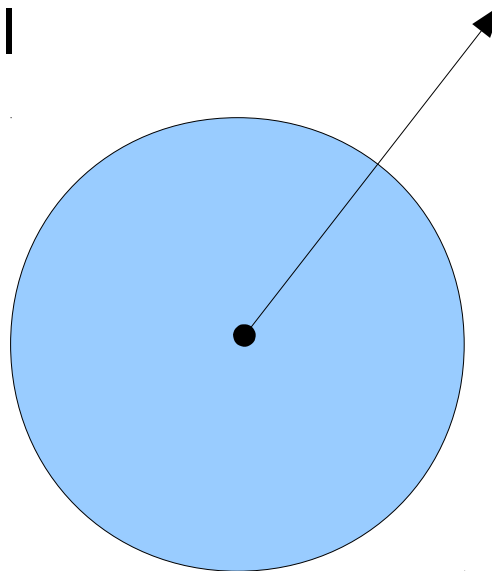


Ray tracing

- The ray has a parametric equation

$$\mathbf{r}(t) = \mathbf{p} + t \mathbf{d}$$

- $t > 0$ is needed to form a half-line
- A ray is directional



Ray tracing

■ Ray – sphere intersection

- Condition 1: to be along the ray

$$\mathbf{r}(t) = \mathbf{p} + t \mathbf{d}$$

- Condition 2: to be on the surface of the sphere

$$\|\mathbf{x}\| = 1 \Leftrightarrow |\mathbf{x}|^2 = 1$$

$$f(\mathbf{x}) = \mathbf{x} \cdot \mathbf{x} - 1 = 0 \longrightarrow \text{Implicit form}$$

- We substitute and transfer

$$(\mathbf{p} + t \mathbf{d}) \cdot (\mathbf{p} + t \mathbf{d}) - 1 = 0$$

- Quadratic equation in
- t

Ray tracing

- Ray – sphere intersection
 - SolutionS for t :

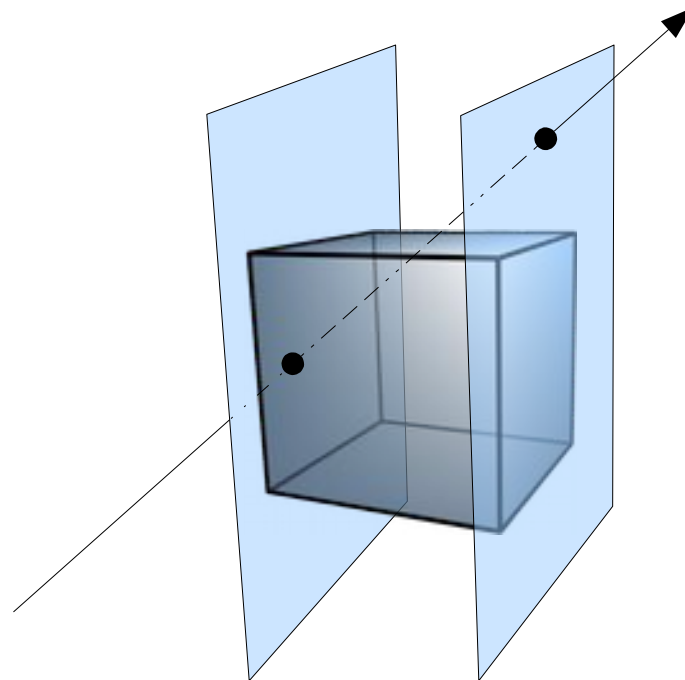
$$t = \frac{-\mathbf{d} \cdot \mathbf{p} \pm \sqrt{(\mathbf{d} \cdot \mathbf{p})^2 - (\mathbf{d} \cdot \mathbf{d})(\mathbf{p} \cdot \mathbf{p} - 1)}}{\mathbf{d} \cdot \mathbf{d}}$$

$$t = -\mathbf{d} \cdot \mathbf{p} \pm \sqrt{(\mathbf{d} \cdot \mathbf{p})^2 - \mathbf{p} \cdot \mathbf{p} + 1} \quad \text{if } \mathbf{d} \text{ is unitary}$$

- Take the nearest solutions (t_{min}) located on the “good” side of the ray ($t > 0$)

Ray tracing

- Ray – box intersection
 - It could be individually done by computing the intersections with the 6 faces
 - It is easier to do the intersections with the three "slices"



Ray tracing

- Ray – slice intersection

$$\mathbf{r}(t) = \mathbf{p} + t \mathbf{d} \Leftrightarrow \begin{cases} r_x(t) = p_x + t d_x \\ r_y(t) = p_y + t d_y \end{cases}$$

$$p_x + t_{xmin} d_x = x_{min}$$

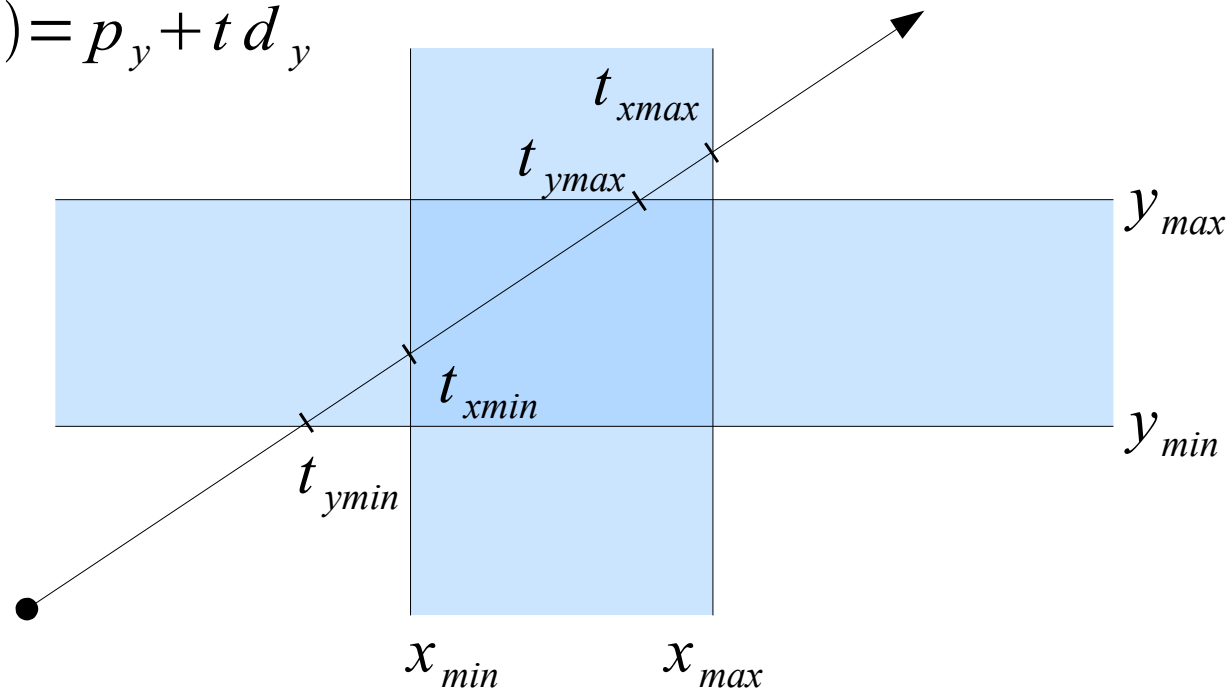
$$t_{xmin} = (x_{min} - p_x) / d_x$$

$$t_{xmax} = (x_{max} - p_x) / d_x$$

$$p_y + t_{ymin} d_y = y_{min}$$

$$t_{ymin} = (y_{min} - p_y) / d_y$$

$$t_{ymax} = (y_{max} - p_y) / d_y$$



Ray tracing

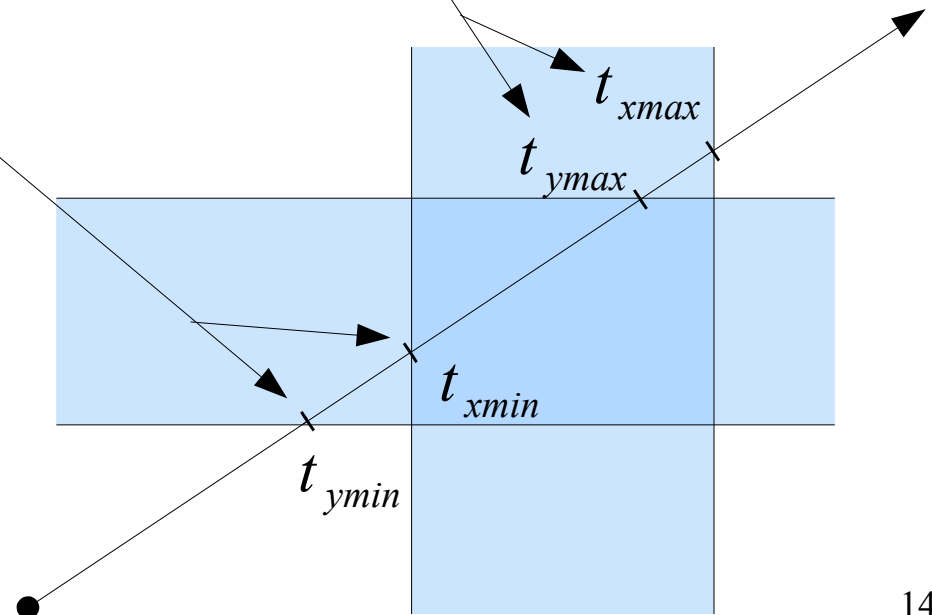
- We have intersections with the slices. How to get the intersections with the box?
 - Combine the results ...
 - The exit point of the ray is the smallest t_{*max}
 - The entry point of the ray is the largest t_{*min}

$$t_{max} = \min(t_{xmax}, t_{ymax})$$

$$t_{min} = \max(t_{xmin}, t_{ymin})$$

- It must also verify

$$t_{max} \geq t_{min}$$



Ray tracing

- Ray – triangle Intersection

- Condition 1: to be along the ray

$$\mathbf{r}(t) = \mathbf{p} + t \mathbf{d}$$

- Condition 2: to be on the plane of the triangle

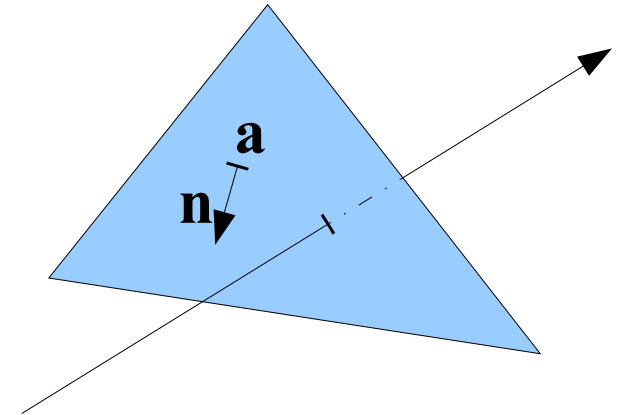
$$(\mathbf{x} - \mathbf{a}) \cdot \mathbf{n} = 0$$

- Condition 3 : to be inside the triangle

- We solve 1&2

$$(\mathbf{p} + t \mathbf{d} - \mathbf{a}) \cdot \mathbf{n} = 0$$

$$t = \frac{(\mathbf{a} - \mathbf{p}) \cdot \mathbf{n}}{\mathbf{d} \cdot \mathbf{n}}$$



Ray tracing

- Inside the triangle (oriented ...)
- Intersection of three half - planes
- We verify that we are

in each half plane

$$(\mathbf{b} - \mathbf{a}) \times (\mathbf{x} - \mathbf{a}) \cdot \mathbf{n} > 0$$

$$(\mathbf{c} - \mathbf{b}) \times (\mathbf{x} - \mathbf{b}) \cdot \mathbf{n} > 0$$

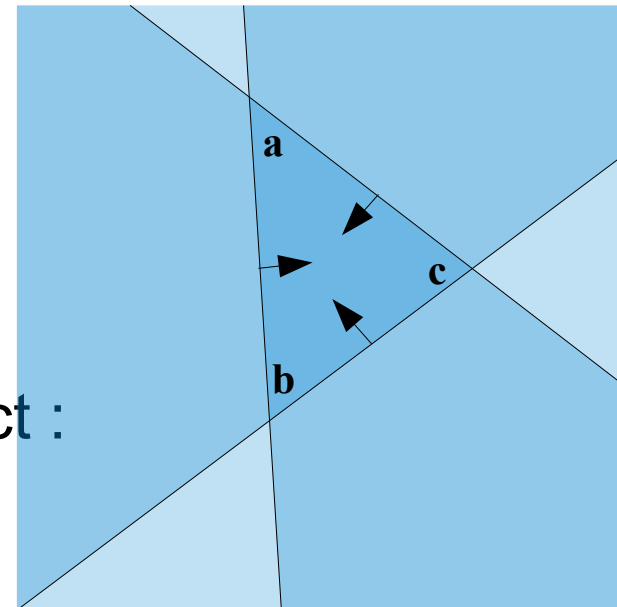
$$(\mathbf{a} - \mathbf{c}) \times (\mathbf{x} - \mathbf{c}) \cdot \mathbf{n} > 0$$

or with the scalar triple product :

$$(\mathbf{b} - \mathbf{a}, \mathbf{x} - \mathbf{a}, \mathbf{n}) > 0$$

$$(\mathbf{c} - \mathbf{b}, \mathbf{x} - \mathbf{b}, \mathbf{n}) > 0$$

$$(\mathbf{a} - \mathbf{c}, \mathbf{x} - \mathbf{c}, \mathbf{n}) > 0$$



Ray tracing

- Ray – triangle Intersection
 - Has to be fast
 - Has to be robust
 - There are a number of efficient algorithms
 - Cf course website for pdfs

T. Möller and B. Trumbore. *Fast, Minimum Storage Ray-Triangle Intersection*. Journal of Graphic Tools, 2(1), 21–28, 1997.

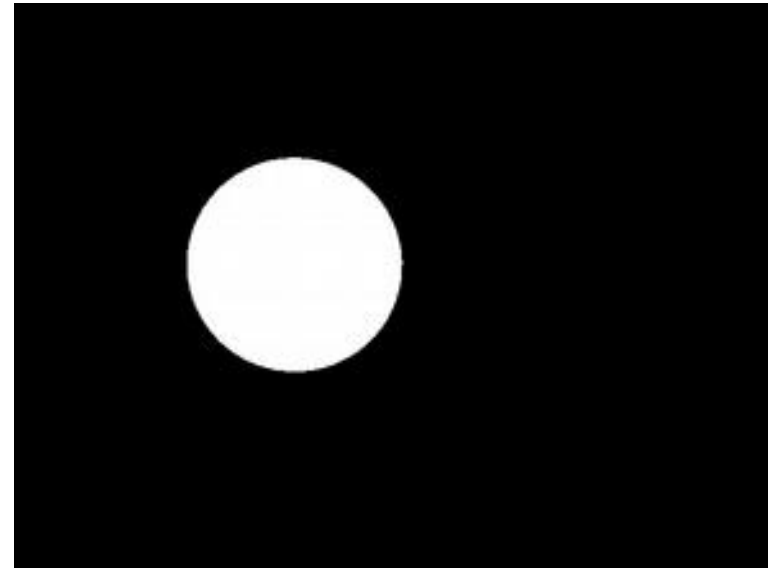
E. Galin, S. Akkouche, *Fast processing of triangle meshes using triangle fans*, Intl Conf. On Shape Modeling and Applications, 2005

M. Shevtsov, A. Soupikov and A. Kapustin, *Ray-Triangle Intersection Algorithm for Modern CPU Architectures*, GraphiCon International Conference on Computer graphics & Vision, 2007

Ray tracing

- Raytracing basis :

```
Let S be a sphere(O,r)
For each pixel of the screen i,j
{
  ray r=camera.calculate_ray(i,j)
  S.intersect(r,0,+infinite,test_intersect,t)
  if (test_intersect) is true
    colourize the pixel i,j in white
}
```



Ray tracing

- Intersection with many primitives
 - For a given direction, we want the nearest intersection of the observer ... the idea is the following :

```
group.intersect(r,tmin,tmax, tbest,nearest_surface)
{
  initialize tbest to +infinite
  initialize nearest_surface to « nothing »
  For each primitive s
  {
    Calculate the intersection of s with the ray r
    intersection = s.intersect(r,tmin,tbest)
    if (intersection) tbest = t ; nearest_surface = s
  }
}
```

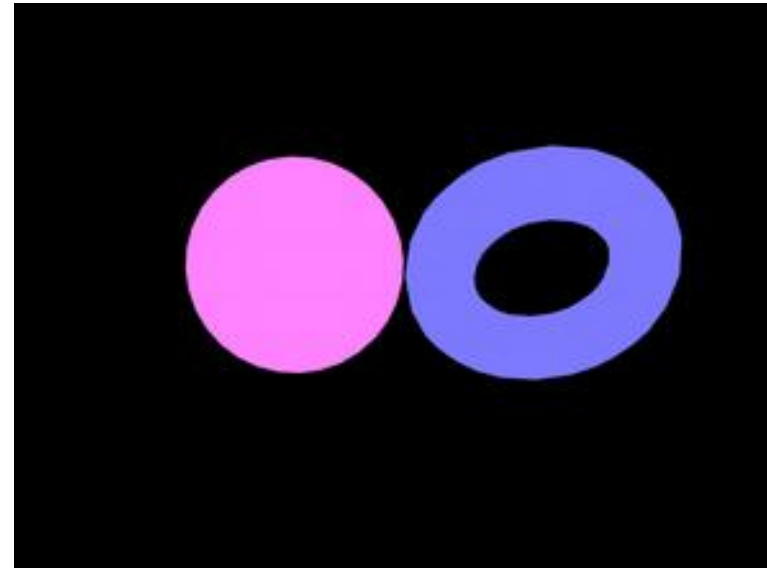
- This algorithm is linear with respect to the number of primitives (there are better algorithms, see later)

Ray tracing

- Modification of ray-tracing and image obtained

```
Let be S a scene
For each pixel of the screen i,j
{
  ray r=camera.compute_ray(i,j)
  c=S.getcolor(r,0,+inf)
  colorize the pixel with colour c
}
```

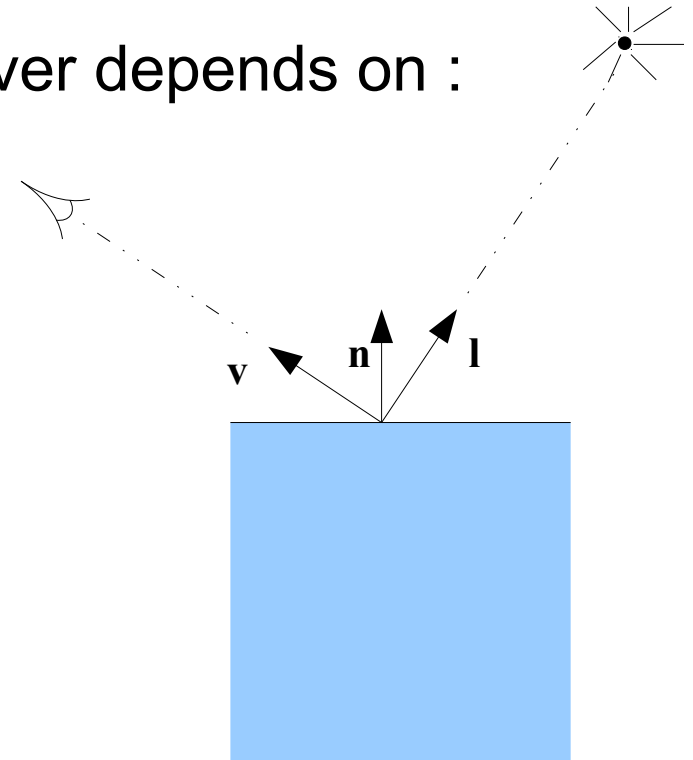
```
Scene.getcolor(r,tmin,tmax)
{
  group.intersect(r,tmin,tmax,surf,t)
  if (surf isn't empty) return the colour of
  the surface
  else return the black colour
}
```

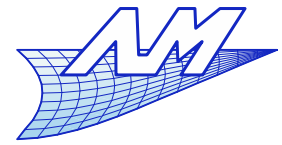


Ray tracing

■ Shades

- We must consider the illumination of surfaces by one or more lights
- The color returned to the observer depends on :
 - the surface (material, colour)
 - the angle of view
 - angle of the light source
 - the surface normal.
 - These parameters are associated with a model to calculate the color perceived by the observer
 - This model can be empirical or based on physical considerations





Physical quantities associated with the propagation of light

Ray tracing

- What is the appropriate size or units to represent a colour or brightness?
 - Power (W) ?
 - Radiosity ($W \cdot m^{-2}$) ?
 - Radiant intensity ($W \cdot sr^{-1}$) ?
 - Radiance ($W \cdot m^{-2} \cdot sr^{-1}$) ?
 - All these units assume the independence of the eye's sensitivity to the wavelength. This is obviously not true!

Note : The *solid angle* of a cone is measured in steradians: it is the measure of the intercepted surface by a unit radius sphere centred on the apex of the cone.

Ray tracing

- We prefer to use a different base unit that is the **candela** or lumens per steradian ;
- $1 \text{ lm} = f(\text{emission spectrum}) * 1 \text{ W}$, is an SI unit connected to the watt *via* a multiplicative factor depending on the so called “**standard observer**”
 - Luminous flux (Lumen = lm)
 - Luminous intensity ($\text{lm} \cdot \text{sr}^{-1} = \text{cd}$)
 - Illuminance ($\text{lm} \cdot \text{m}^{-2}$)
 - Luminance ($\text{lm} \cdot \text{m}^{-2} \cdot \text{sr}^{-1}$)

Ray tracing

- Radiance / Radiometry $W \cdot m^{-2} \cdot sr^{-1}$ or $W \cdot m^{-2} \cdot sr^{-1} \cdot (nm)^{-1}$
 - Physical measurement of the electromagnetic energy
 - unit based on the SI watt
 - Laws of conservation of energy
- Luminance / Chroma $lm \cdot m^{-2} \cdot sr^{-1} = cd \cdot m^{-2}$
 - Perceptual measurement of the quantity of light at different wavelengths
 - Units are based on the SI lumen
 - Conservation laws if no fluorescence (change of wavelength of the light by fluorescent chemical compounds)

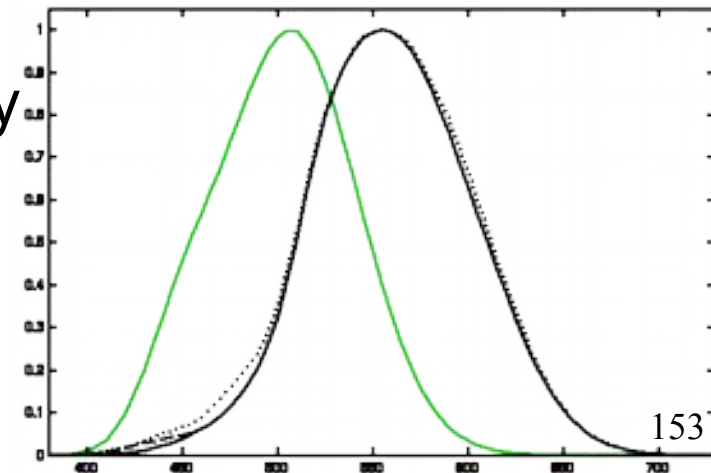
Ray tracing

- A 100 W incandescent lamp
 - 1500 lumens or about 120 candelas if it emits uniformly in all directions.
 - Only a part of the light energy is emitted in the visible spectrum (the rest is in infrared)
- A 21W fluorescent compact lamp
 - About 1500 lumens too !
 - Almost all of the energy is emitted in visible light

- Relationship between physical intensity and perceptual intensity

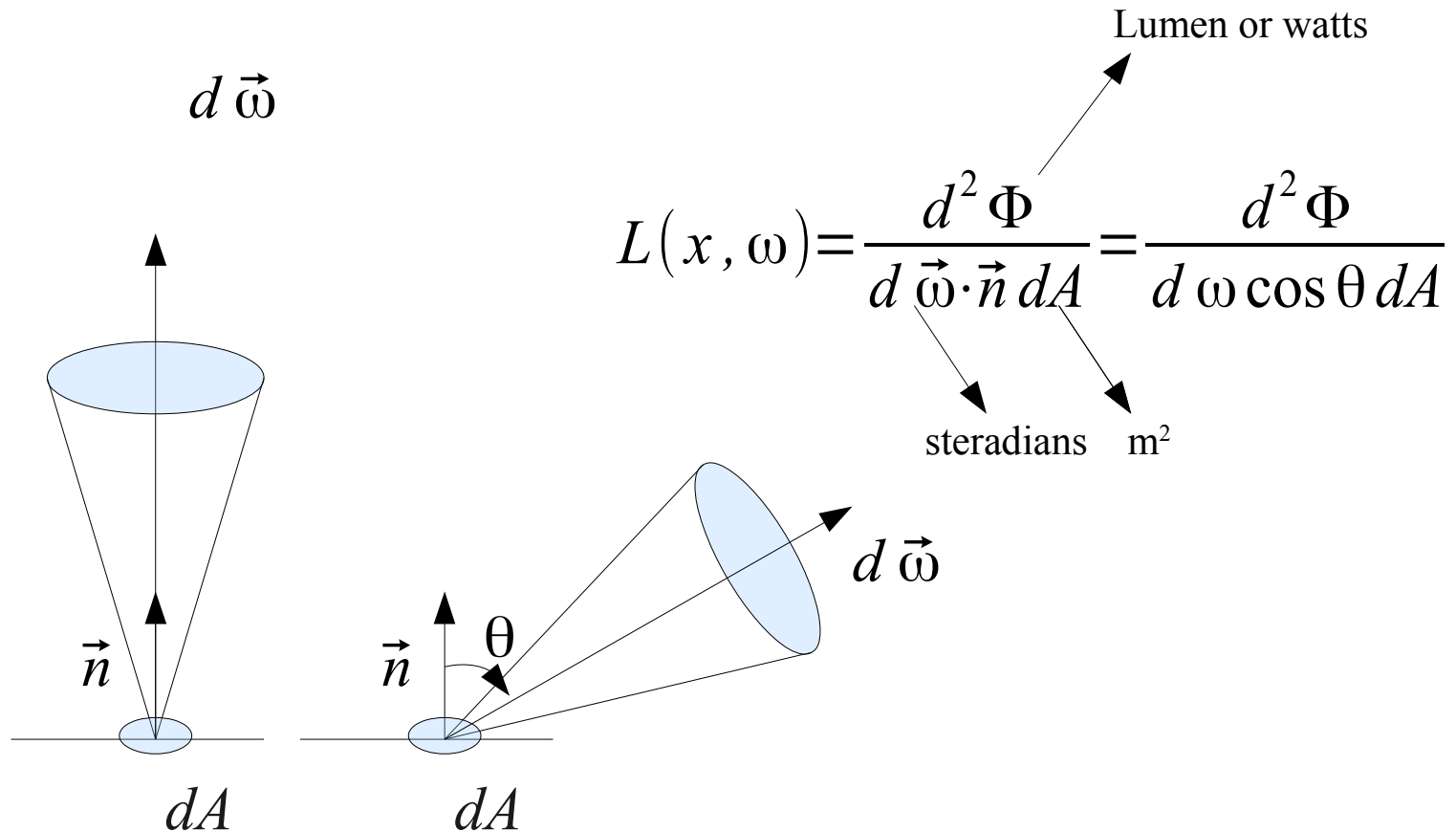
$$I_v = 683.002 \int_0^{\infty} I(\lambda) \bar{y}(\lambda) d\lambda$$

lm ← I_v lm/W ← 683.002 W · nm⁻¹ ← $I(\lambda)$



Ray tracing

- Luminance (radiance) is the power per unit area oriented perpendicularly to the ray; per unit solid angle



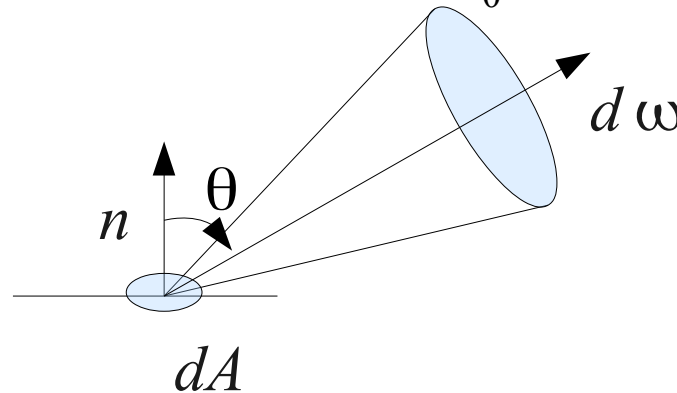
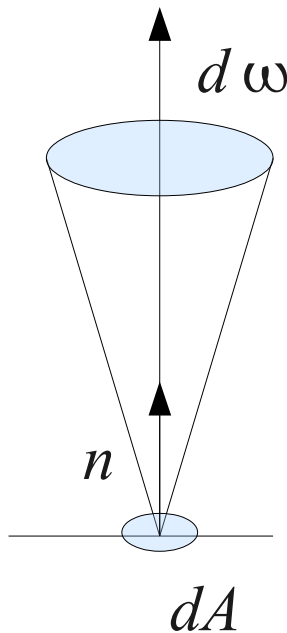
Ray tracing

- If the surface uniformly emits (constant L) a power dQ , this relationship is verified:

$$dQ = \int d^2 \Phi = \int L d\omega \cos \theta dA$$

$$dQ = \int_0^{\frac{\pi}{2}} \int_0^{2\pi} L \cos \theta \sin \theta d\phi d\theta dA$$

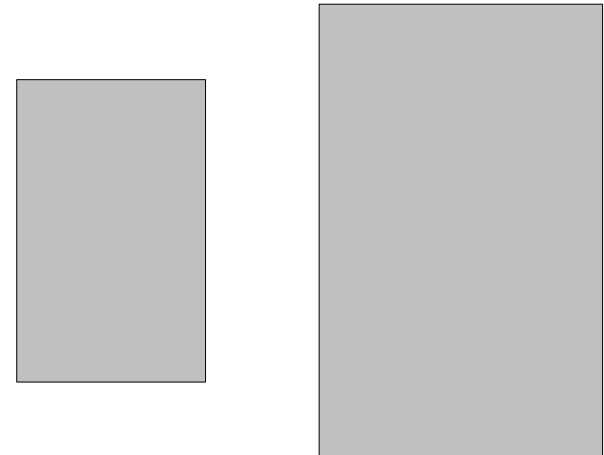
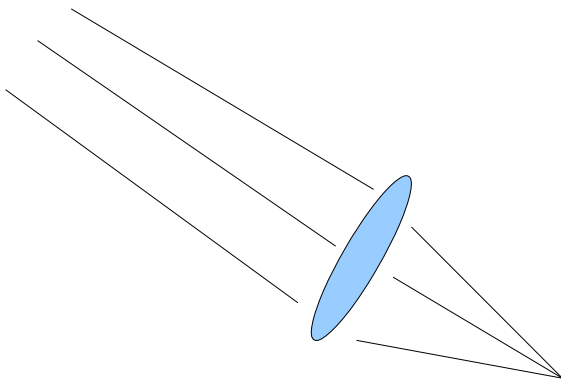
$$dQ = 2\pi \cdot L \int_0^{\frac{\pi}{2}} \sin \theta \cos \theta d\theta dA = \pi L dA$$



Ray tracing

Quiz :

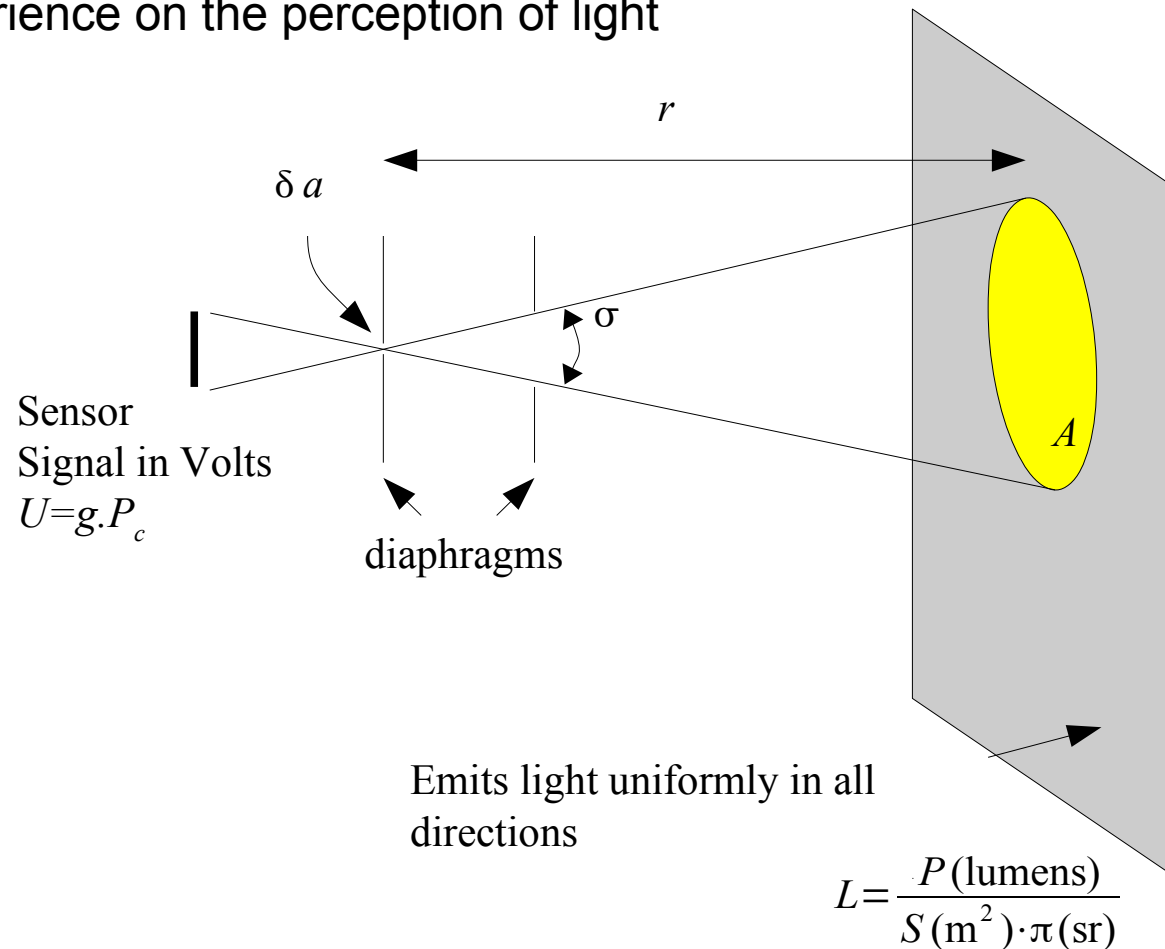
- Does Radiance / luminance increase if the sunlight is concentrated with a magnifying glass?
- Do radiance / luminance of an illuminating surface depend on the visible area (or the viewing distance)?



Ray tracing

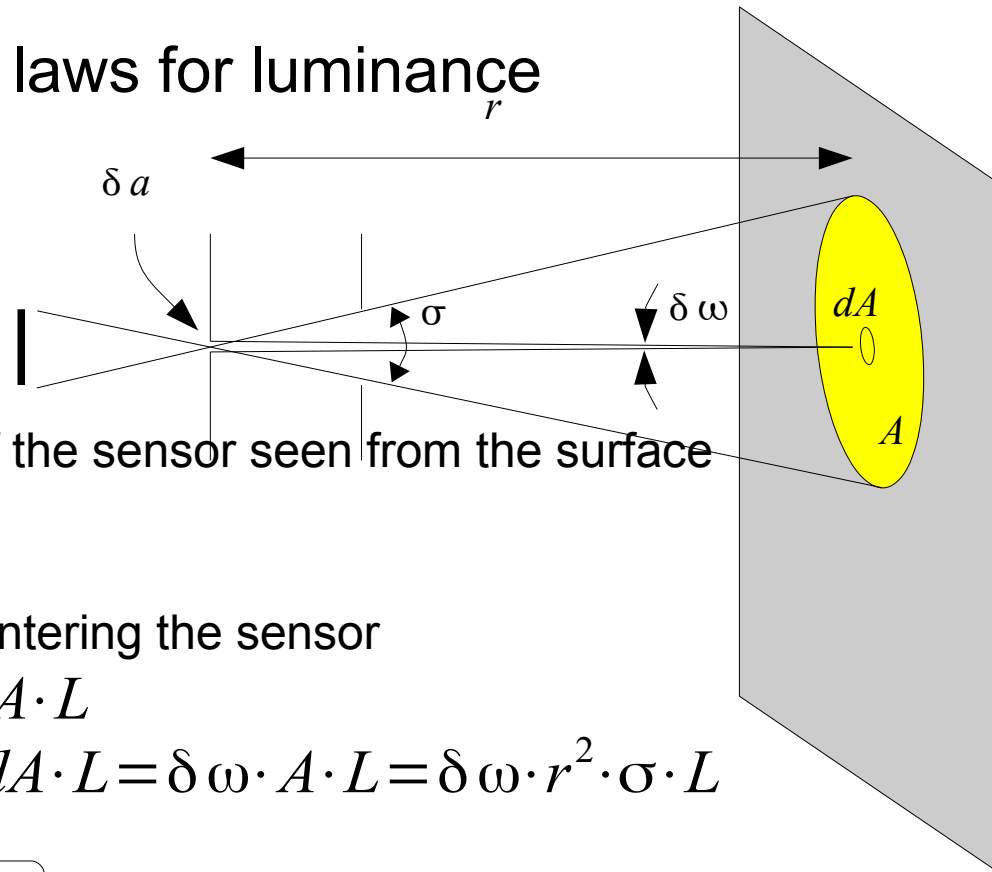
- Conservation laws for luminance

Experience on the perception of light



Ray tracing

- Conservation laws for luminance



- Solid angle of the sensor seen from the surface

$$\delta \omega = \frac{\delta a}{r^2}$$

- Light power entering the sensor

$$dP_c = \delta \omega \cdot dA \cdot L$$

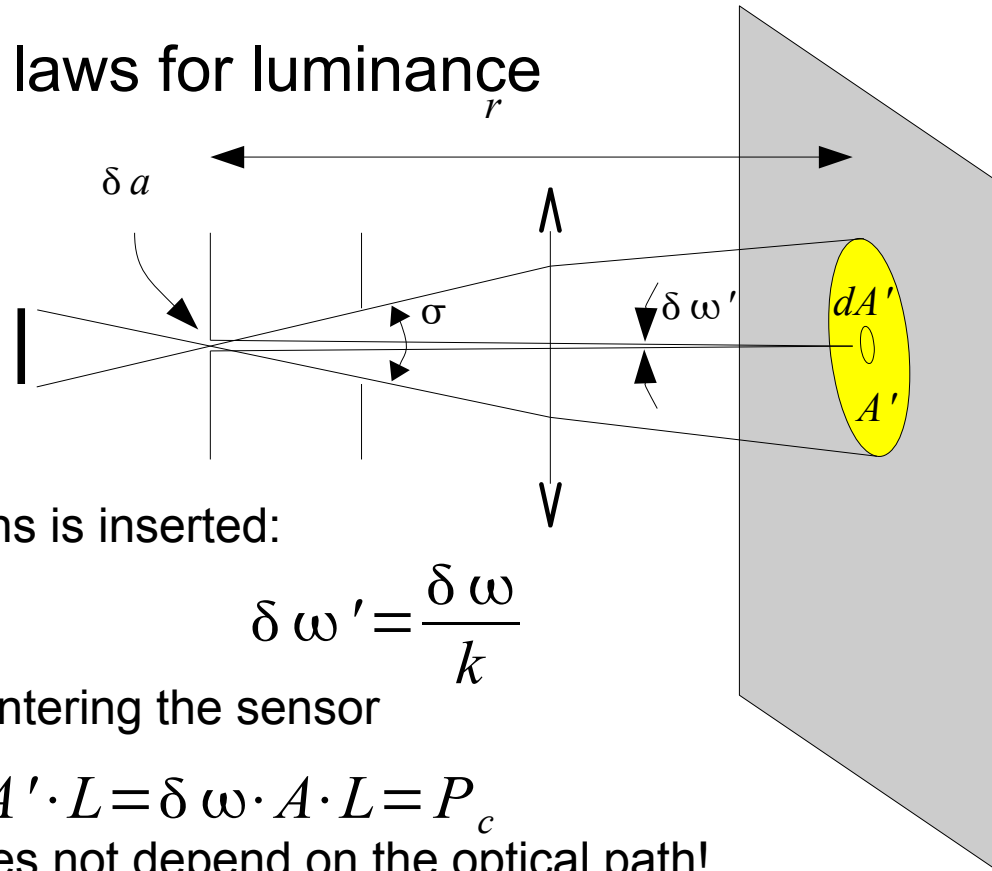
$$P_c = \int_A \delta \omega \cdot dA \cdot L = \delta \omega \cdot A \cdot L = \delta \omega \cdot r^2 \cdot \sigma \cdot L$$

Sensor geometry $\leftarrow \boxed{\delta a \cdot \sigma} \cdot L$

- The result does not depend on the distance!
- Luminance (or radiance) are preserved

Ray tracing

- Conservation laws for luminance



- If a perfect lens is inserted:

$$A' = k \cdot A \quad \delta \omega' = \frac{\delta \omega}{k}$$

- Light power entering the sensor

$$P_c' = \delta \omega' \cdot A' \cdot L = \delta \omega \cdot A \cdot L = P_c$$

- The result does not depend on the optical path!

Ray tracing

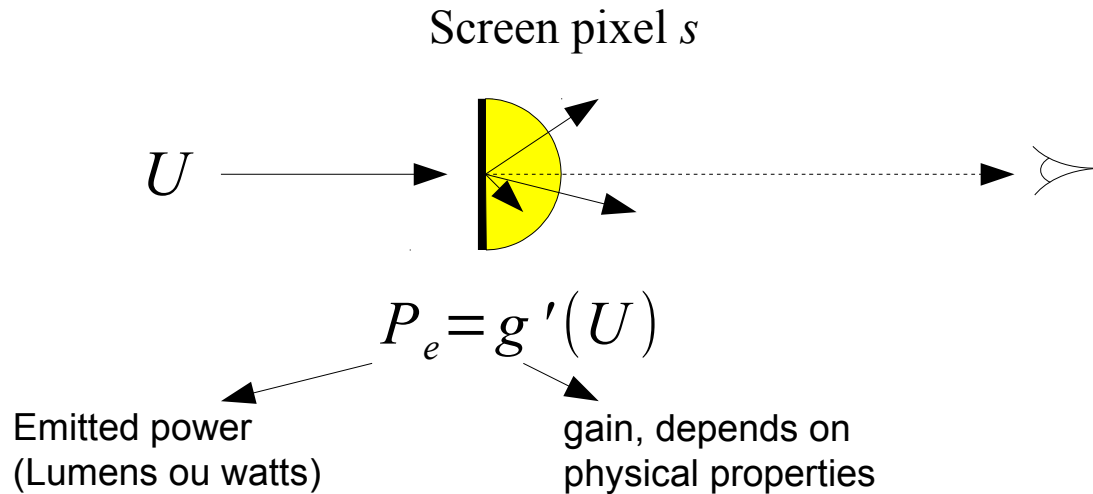
- Some characteristic values of the luminance
 - Surface of the Sun: noon and clear atmosphere
 $L_v \sim 1.6 \cdot 10^9 \text{ lm} \cdot \text{m}^{-2} \cdot \text{sr}^{-1}$
 (out of atmosphere : $L_v^0 \sim 2 \cdot 10^9 \text{ lm} \cdot \text{m}^{-2} \cdot \text{sr}^{-1}$)
 - Full moon at midnight $L_m \sim 4000 \text{ lm} \cdot \text{m}^{-2} \cdot \text{sr}^{-1}$
 - TV screen, computer etc... $L_{TV} = 50 \sim 500 \text{ lm} \cdot \text{m}^{-2} \cdot \text{sr}^{-1}$
 - A 2000 lumens multimedia projector, with a 2m*1.5m projection screen , perfectly reflecting (albedo=1), light scattering by Lambert's law

$$L_p = \frac{2000}{1.5 \cdot 2 \cdot \pi} \sim 200 \text{ lm} \cdot \text{m}^{-2} \cdot \text{sr}^{-1}$$
 - Luminance of a blank sheet of A4 paper (albedo = 0.6) exposed perpendicularly to the sun rays ($\Omega_v \sim 6 \cdot 10^{-5} \text{ sr}$), perfectly diffusing

$$L_f \sim 0.6 \frac{1.6 \cdot 10^9 \cdot 6 \cdot 10^{-5}}{\pi} \sim 18000 \text{ lm} \cdot \text{m}^{-2} \cdot \text{sr}^{-1}$$
 - In reality, closer to $L_f \sim 10000 \text{ lm} \cdot \text{m}^{-2} \cdot \text{sr}^{-1}$, because paper is not perfectly diffusing (a fair part of the incoming flux is reflected mirror-like, thus decreasing the amount of diffuse reflection)

Ray tracing

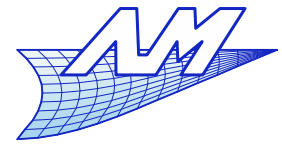
- During the restitution (display), a signal in volts U is converted to light intensity.



- Let state that the emission of light is uniform and made in all directions, so the luminance is given by

$$L = \frac{P_e}{s \cdot \pi} = \frac{g'(U)}{s \cdot \pi} = g''(U)$$

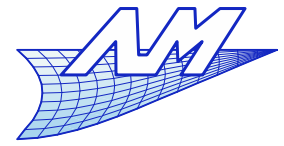
Depends only on the geometrical and physical characteristics of the "pixel" ¹⁶¹



Ray tracing

Conclusion :

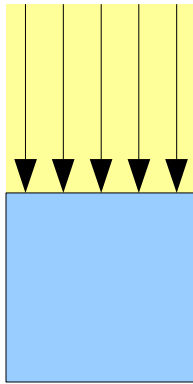
- Luminance is the right variable to "display" in order to accurately reflect the reality
- When using ray tracing, this is the adequate physical variable that is associated with a ray and is computed / transported along the ray.



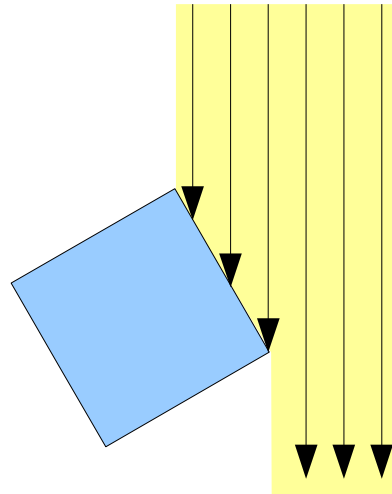
Some laws of light reflection

Ray tracing

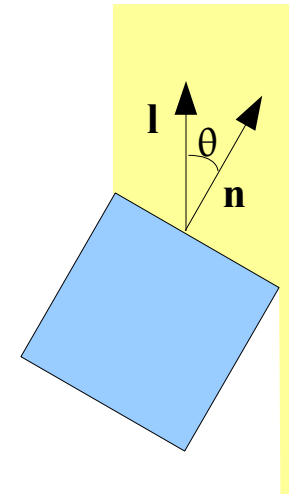
- Incident light
 - Lambert's law (Jean-Henri Lambert 1728-1777)



The upper face receives a quantity of light



The upper face receives a fraction of the quantity of the emitted light

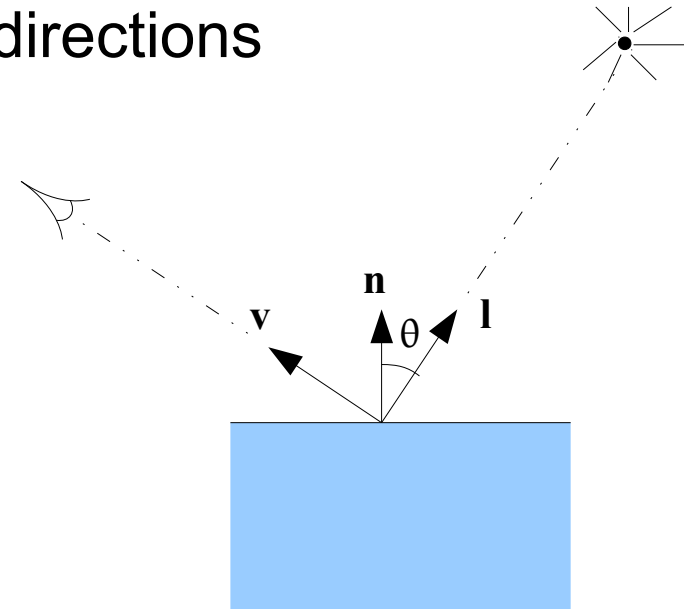


In fact, the upper face receives a quantity of light Proportional to $\cos \theta = \mathbf{l} \cdot \mathbf{n}$

In all cases, the incident light is reflected in all directions with the same intensity

Ray tracing

- Perfectly diffuse surface model
 - Also called Lambertian Surface
 - It reflects light "equitably" in all directions
 - The appearance (brightness) is not depending on the position of the viewer



Ray tracing

- Lambertian surface model

- Quantity of light received by elementary surface

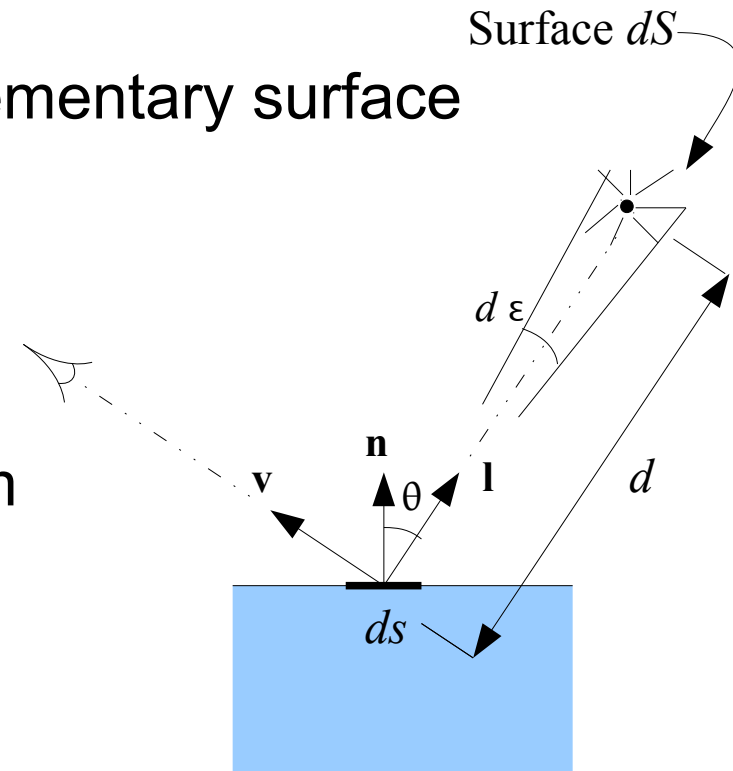
$$d\varepsilon = \frac{dS}{d^2}$$

$$dQ = L_i \cdot d\varepsilon \cdot \cos \theta \cdot ds \quad (\text{Lumens})$$

- This is re-emitted with a uniform luminance pattern

$$L_d = \frac{dQ}{ds \cdot \pi} = L_i \cdot \cos \theta \frac{d\varepsilon}{\pi} = \cos \theta I_i$$

$$I_i = L_i \cdot \frac{d\varepsilon}{\pi} = L_i \cdot \frac{dS}{\pi d^2}$$



is the incident illumination.

Ray tracing

- Lambertian surface model
 - This surface has a color and an albedo (coefficient of reflection)

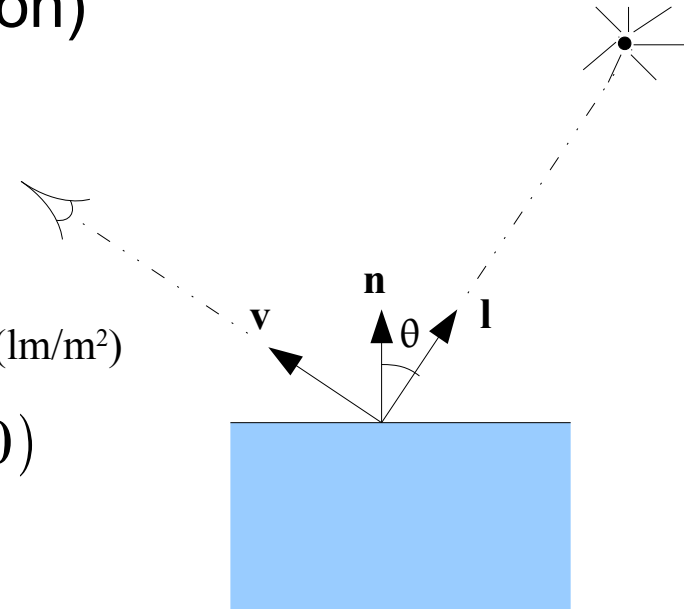
Light (luminance) diffused towards the observer

Coefficient linked to the albedo

Incident illumination (lm/m^2)

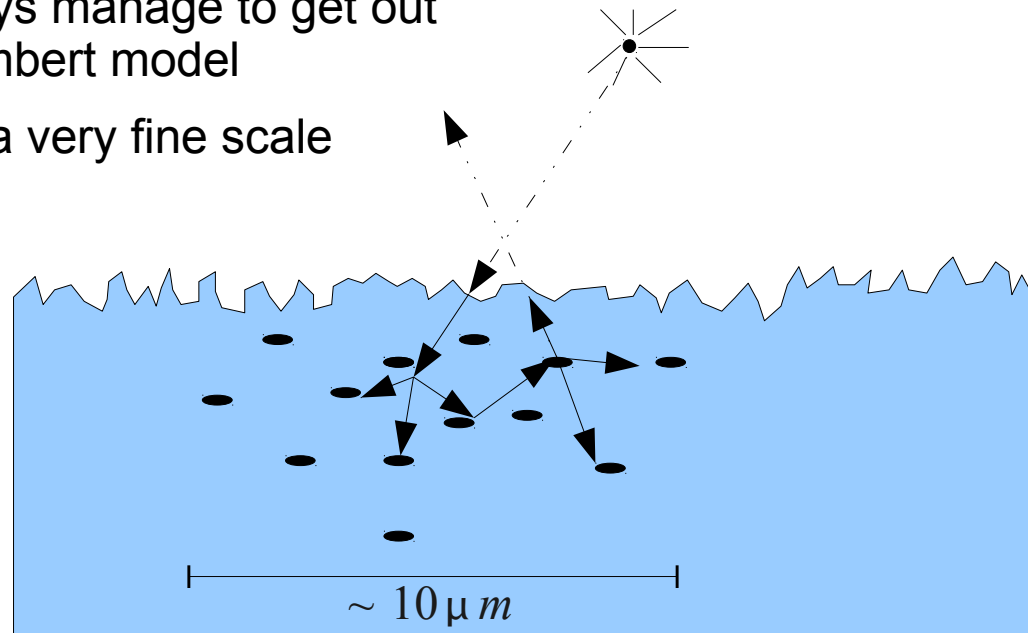
$$L_d = A I \max(\mathbf{n} \cdot \mathbf{l}, 0)$$

$$L_d = k_d I$$



Ray tracing

- Lambertian surface : Physical explanation
 - Rough surface (no mirror effect)
 - Penetration of the light rays
 - diffusion by particles
(The particles and the matrix give the color of the surface !)
 - A fraction of the light rays manage to get out
(parameter A of the Lambert model
 - Everything happens at a very fine scale
 - The outgoing light rays
can be considered as
coming out exactly
from the point of entry

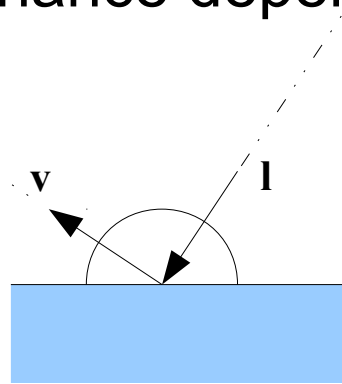


Ray tracing

- Limits of the lambertian model
 - Not adapted to conditions of grazing light...
 - The lambertian model tends towards zero reflection...which is mostly inaccurate with usual surfaces
 - e.g. Moon's surface : the luminance of the edges do not vanish ! (on a full moon)
 - Non-conductive surfaces (not metallic)
 - "Rough" plaster, frosted ceramic, etc., bitumen, "rough" concrete, opaque and frosted glass

Ray tracing

- Lambertian surface : matt appearance
 - The returned luminance depends only on the position of the source.



$A=0.4$

$A=0.6$

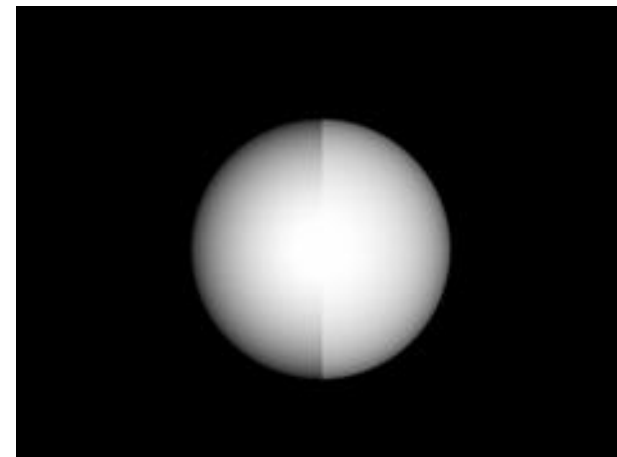
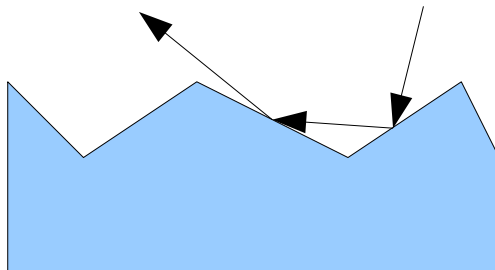
$A=0.8$

$A=1.0$



Ray tracing

- Other models of diffuse reflection
 - Oren-Nayar model
 - Includes inter-reflection on the surface irregularities ... adds a roughness parameter.



Ray tracing

- Lambertian surface : colour
 - Albedo coefficient different for the three primary colors
 - In practice, a single fixed coefficient (A) multiplied by a coefficient depending on each colour (triplet representing the colour of the material) are used

$$\begin{pmatrix} L_d^R \\ L_d^G \\ L_d^B \end{pmatrix} = A \begin{pmatrix} C^R \cdot I^R \\ C^G \cdot I^G \\ C^B \cdot I^B \end{pmatrix} \max(\mathbf{n} \cdot \mathbf{l}, 0)$$

~ Colour of material

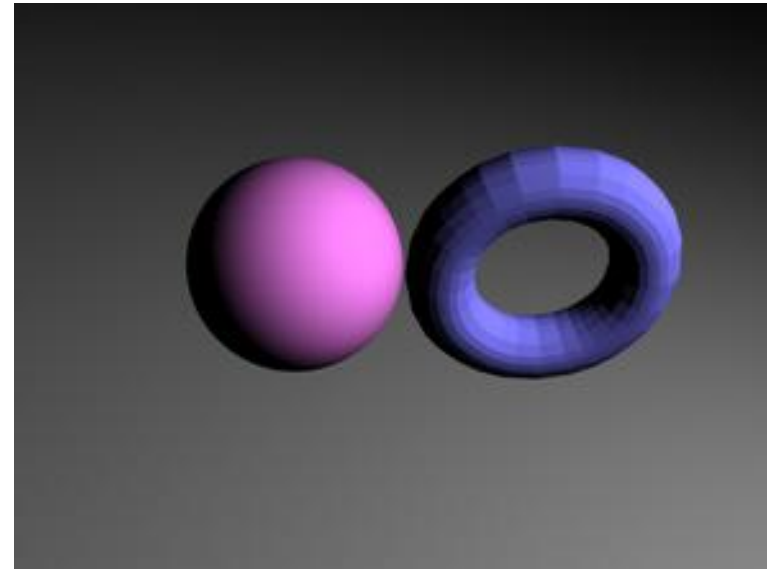
~ Colour of the incident light

Ray tracing

- Image obtained with Lambertian shading

```
Scene.getcolor(r,tmin,tmax)
{
  group.intersect(r,tmin,tmax,surf,t)
  if (surf non empty)
  {
    point = r.eval(t)
    normal=surf.give_normal(point)
    return surf.shading(r,point,normal,source)
  }
  Else return the background color
}
```

```
surface.shading(ray,point,normal,source)
{
  v = -normalize(ray.direction)
  l = normalize(source.position-point)
  // calculate the lambertian shading
  return the colour
}
```



■ References

J. H. Lambert. *Photometria sive de mensura de gratibus luminis, colorum umbrae*. Eberhard Klett, 1760

P. Kubelka and F. Lunk. Ein Beitrag zur Optik der Farbanstriche. *Z. Techn. Physik*, **12**:593-601, 1931

S. Orchard. Reflection and transmission of light by diffusing suspensions. *J. Opt. Soc. Am.*, **59**:1584-1597, 1969

J. Reichmann. Determination of absorption and scattering coefficients for non homogeneous media. *Applied Optics*, **12**:1811-1815, 1973.

M. Oren and S. K. Nayar “Generalization of Lambert’s reflectance model”, ACM SIGGRAPH 1994 Proceeding.

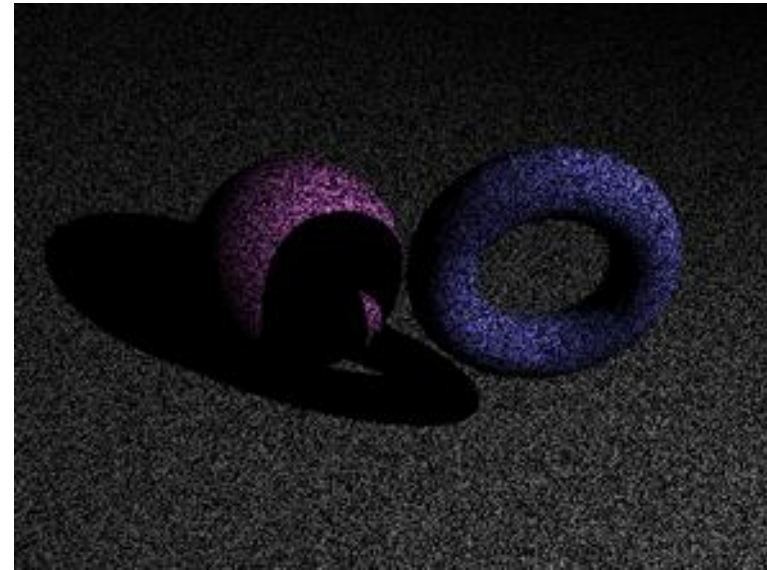
Ray tracing

- Shadows ...
 - The surfaces are not aware if something blocks the light from the light source
 - We have all we need to add a small test !
 - Construct a ray from the lamp through the current point on the surface and check that it does not intersect any other surface in between...

Ray tracing

- Lambertian shading + shadows

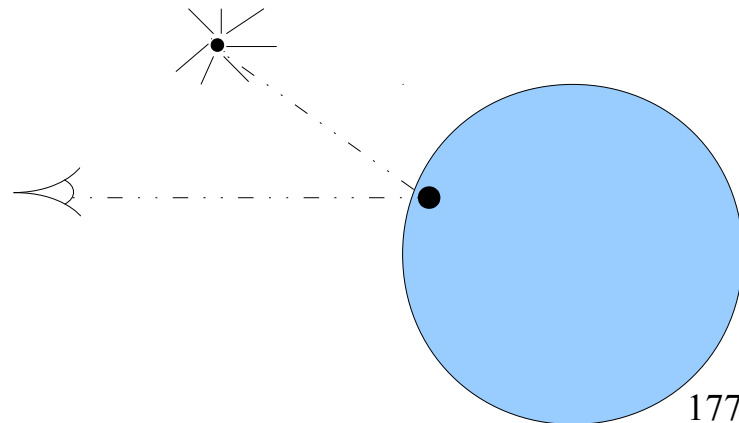
```
surface.shading(rayon,point,normal,source)
{
  shadow_ray = ray(point,source.position - point)
  group.intersect(shadow_ray,tmin,tmax,t,surf)
  if surf is empty
  {
    v = -normalize(ray.direction)
    l = normalize(source.position-point)
    Calculate the lambertian shading
    return the obtained colour
  }
  else return black colour
}
```



Ray tracing

■ Shadows

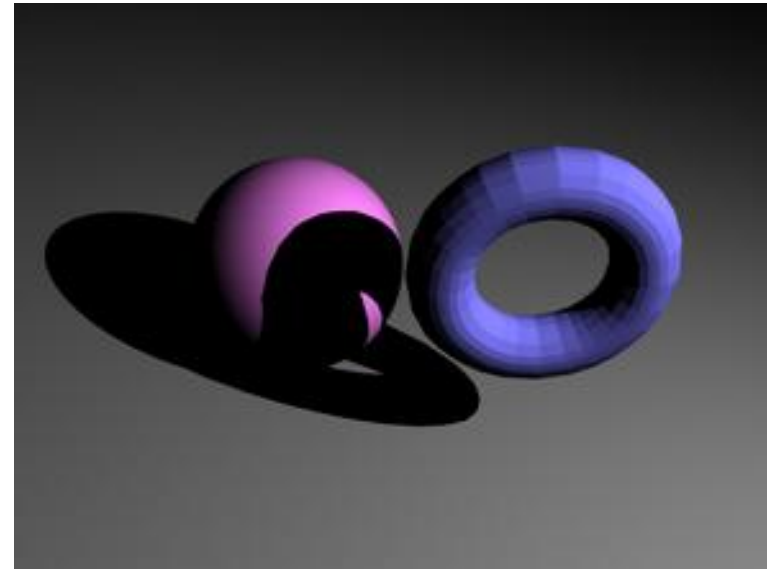
- Be careful when searching intersection of the ray from the light source
 - Is the numerically calculated intersection point with the ray from the observer located exactly on the surface ?
 - If it is a little bit toward the "outside", then everything goes as planned
 - If it is a little bit toward the "inside," then we find an intersection for the ray from the source ...
 - Solution :
ensuring that the beam used for computing shadows starts at a small distance from the surface (toward the outside)



Ray tracing

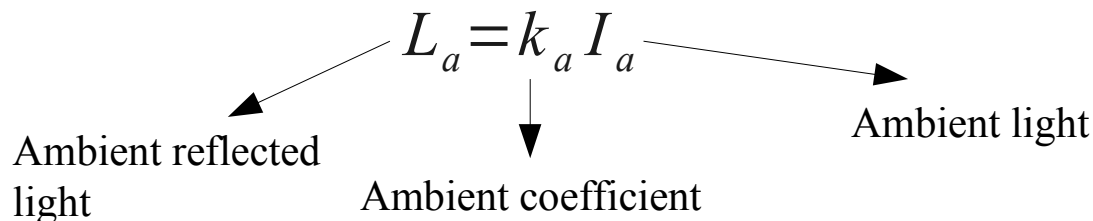
- Shadows – correct treatment !

```
shadow_ray = ray(point+normal*epsilon,  
                source.position - point)
```



Ray tracing

- Multiple light sources
 - Important to "fill-in" shadows
 - Very simple implementation: for each light, add the corresponding contribution
- Ambient shading
 - A perfectly dark shadow is unrealistic
 - Solution 1 : place a small source of light just next to the camera
 - Solution 2 : add a constant contribution to shading determined somewhere else

$$L_a = k_a I_a$$


Ambient reflected light

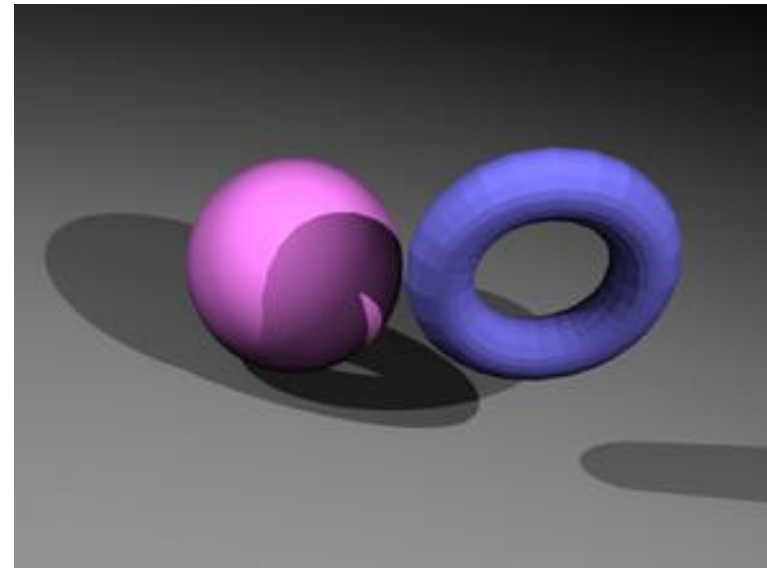
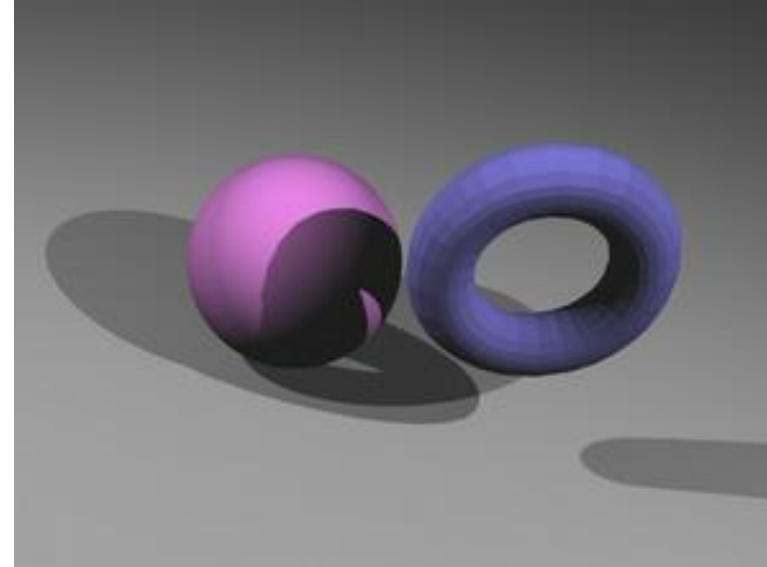
Ambient coefficient

Ambient light

Ray tracing

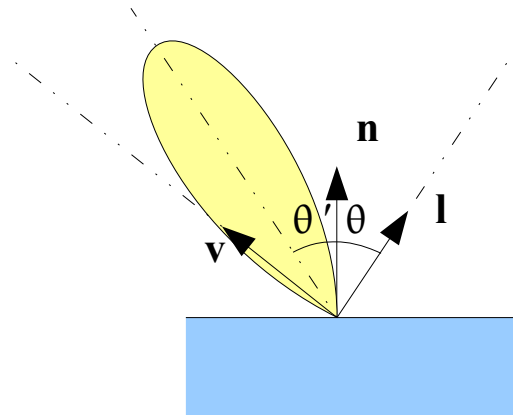
- Multiple light sources

```
surface.shading(ray,point,normal,sourceS)
{
  colour= ambient // eventuality black !
  For each source in sourceS
  {
    if not in shadows (see previously)
    {
      v = -normalize(ray.direction)
      l = normalize(source.position-point)
      calculate the lambertian shading
      colour = colour + calculated shading
    }
  }
  return colour
}
```



Ray tracing

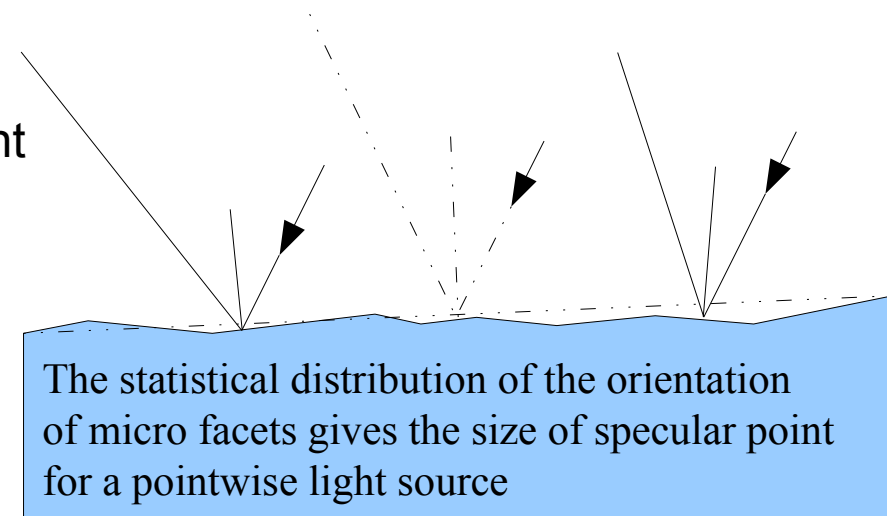
- Specular reflection
 - Light directly reflected from the surface (that do not penetrate the material)
 - For relatively smooth surfaces
 - These surfaces reflect light preferentially in one direction



- The preferred direction is symmetric like for a mirror : $\theta = \theta'$

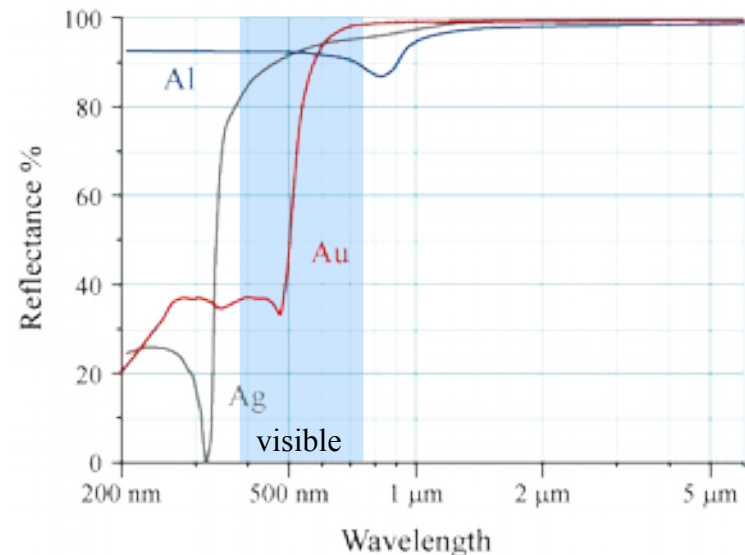
Ray tracing

- Specular reflection : physics
 - The surface is perfectly smooth: mirror
 - All the light from the source is reflected in one direction - special case (see later)
 - Some materials have a specular reflection that shows a diffuse spot
 - Even for pointwise sources !
 - There is dispersion of the orientation of the reflected light (on either side of the theoretical angle)
 - This is due to the presence of micro facets at the surface of the solid



Ray tracing

- Specular reflection : physics
 - The reflected colour is dominated by the colour of the incident light
 - The color of an object comes mainly from the partial absorption of the transmitted rays in the matrix of the object (plastics, etc ...)
 - However, the reflection coefficients are also dependent on the visible wavelengths for some materials
 - Copper, gold ...



Ray tracing

- Specular reflection (Blinn-Phong model)
 - Purely phenomenological model
 - The mirror configuration implies that the \mathbf{h} , the bisector of \mathbf{v} and \mathbf{l} , is close to the normal \mathbf{n} .
 - The Blinn-Phong model is then :

$$L_s = k_s I$$

$$k_s = C_s \max(0, \cos \alpha)^p$$

$$= C_s \max(0, \mathbf{n} \cdot \mathbf{h})^p$$

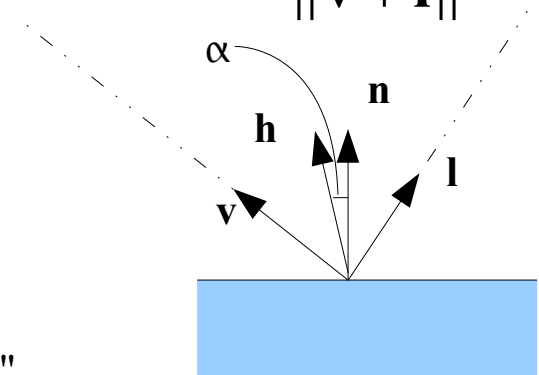
includes the term $\max(\mathbf{n} \cdot \mathbf{l}, 0)$!

Specular coefficient

Specular component of the reflected Light

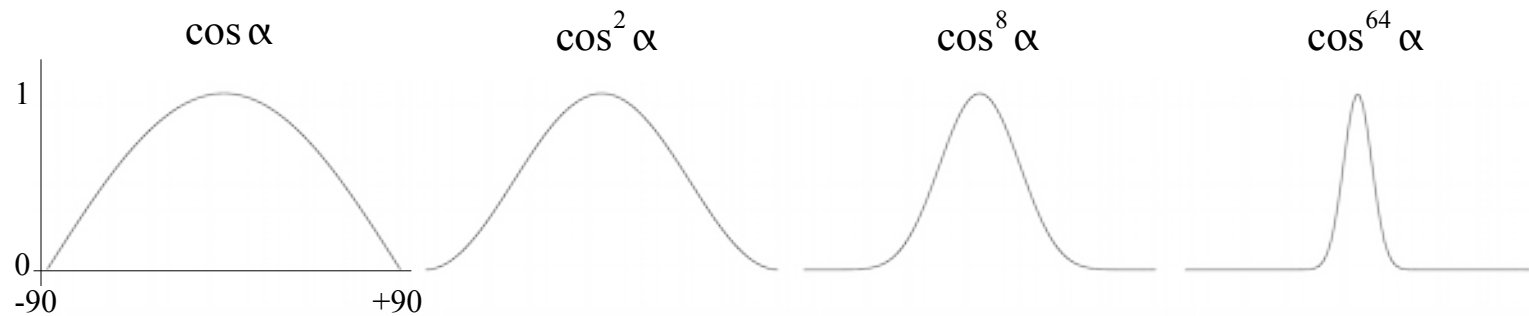
Phong exponent sometimes called "hardness"

$$\mathbf{h} = \frac{\mathbf{v} + \mathbf{l}}{\|\mathbf{v} + \mathbf{l}\|}$$



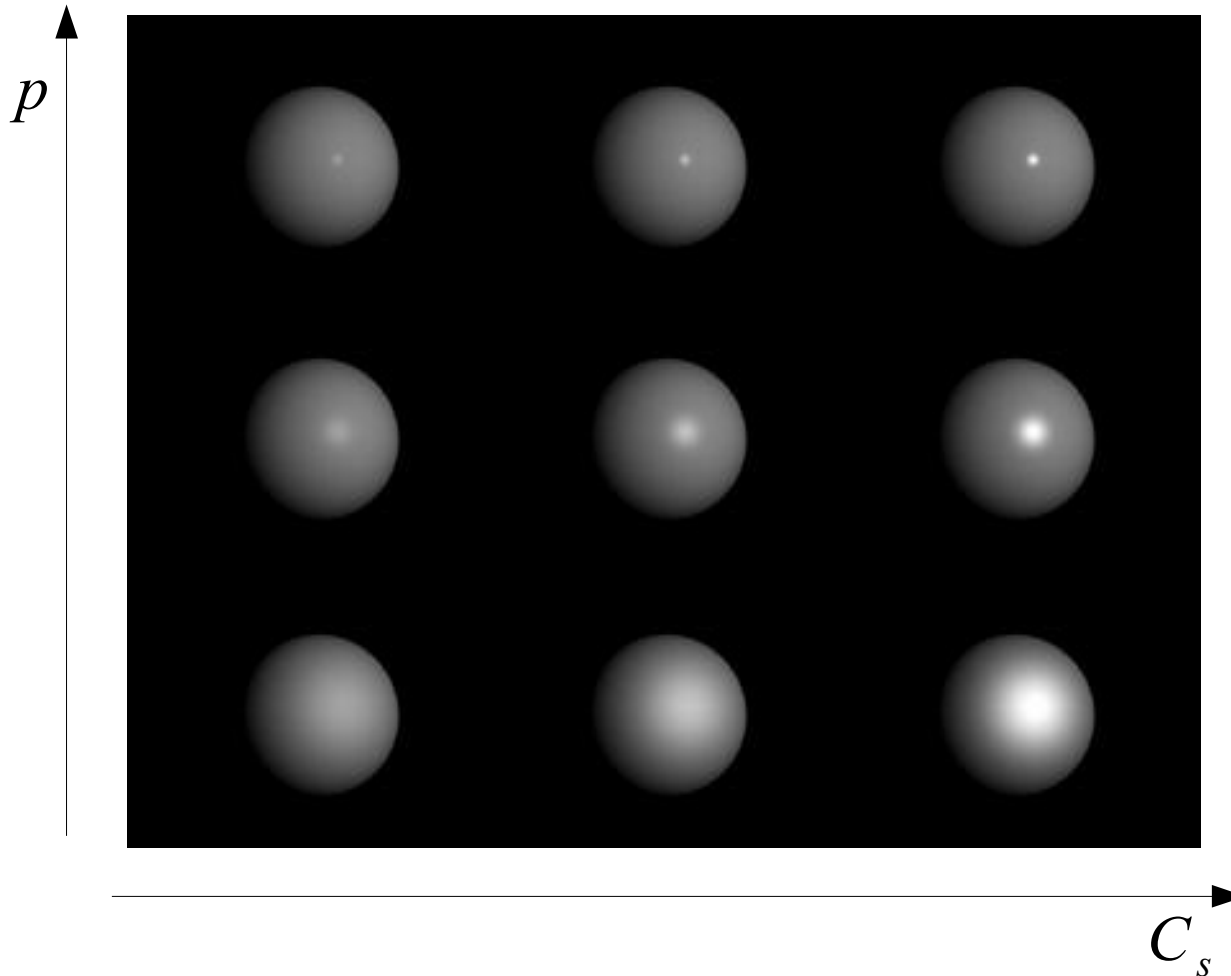
Ray tracing

- Specular reflection (Blinn-Phong model)
 - The Phong exponent relates to the sharpness of the lobe



Ray tracing

- Specular reflection (Blinn-Phong model)



Ray tracing

- Specular reflection (Blinn-Phong model)
 - If p is increased, we tend to a perfect mirror
 - But only light sources are reflected!
 - p is related to the smoothness of the surface
 - The model drifts away from the reality for increasing p
 - (we should see reflections of other objects in the scene!)
 - It is useful for modelling plastic, "wax"-like surfaces, rubber, etc....

Ray tracing

- Others models

- Phong model produces realistic results but does not match any physical law
 - Approximately, it approaches a Gaussian distribution of the orientation of micro facets that would give :

$$k_s = C_s e^{-\left(\frac{\alpha}{m}\right)^2} \max(\mathbf{n} \cdot \mathbf{l}, 0) \quad , m \text{ is a parameter between } 0 \text{ and } 1.$$

- Beckmann distribution: built on physical considerations

$$k_s = C_s D_b \max(\mathbf{n} \cdot \mathbf{l}, 0) \quad D_b = \frac{1}{4m^2 \cos^4 \alpha} e^{-\left(\frac{\tan \alpha}{m}\right)^2}$$

m is the average slope of micro facets.

Ray tracing


- Cook-Torrance models

- Based on the distribution of Beckmann

- + Fresnel terms (partially reflected wave as a function of the angle of incidence)

- + terms corresponding to the self-shadowing (projected shadows by the micro facets on other mf.)

$$k_s = C_s R_s \quad F = F_0 + (1 - \mathbf{h} \cdot \mathbf{v})^5 (1 - F_0)$$

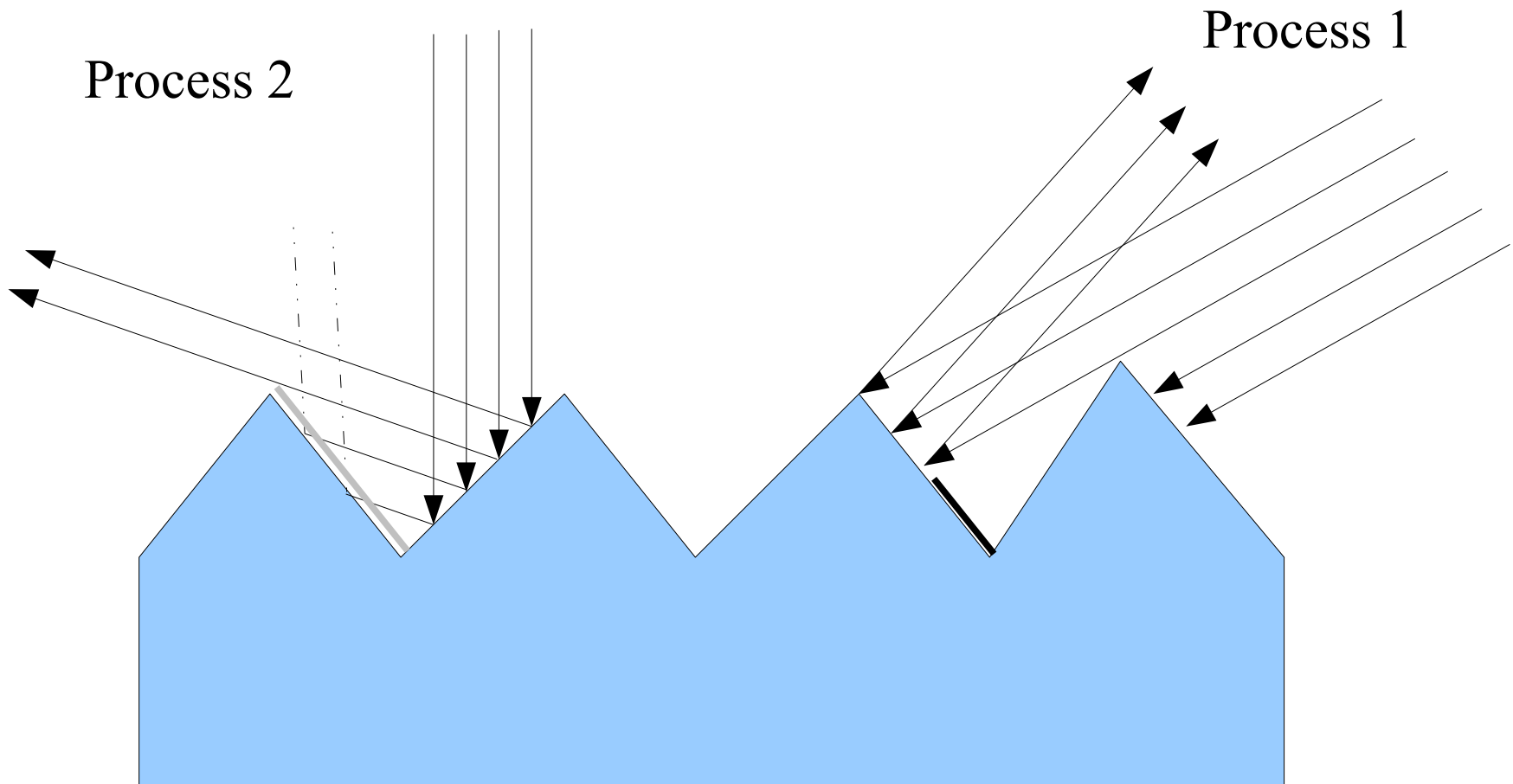
 Reflectance at normal incidence

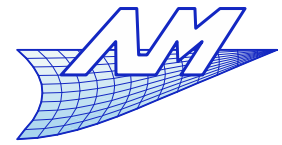
$$R_s = \frac{D_b F G}{\mathbf{v} \cdot \mathbf{n}} \quad G = \min \left(1, \frac{2(\mathbf{h} \cdot \mathbf{n})(\mathbf{v} \cdot \mathbf{n})}{\mathbf{h} \cdot \mathbf{v}}, \frac{2(\mathbf{h} \cdot \mathbf{n})(\mathbf{l} \cdot \mathbf{n})}{\mathbf{h} \cdot \mathbf{v}} \right)$$

R. Cook and K. Torrance “A Reflectance Model for Computer Graphics” ACM Transactions on Graphics, volume 1, number 1, January 1982 pages 7-24

Ray tracing

- Self-shadowing





Ray tracing

- Ward model
 - Includes anisotropy
 - The micro facets are preferentially oriented
 - Model for e.g. brushed metal.

Ray tracing

■ References

Beckmann and Spizzichino “The scattering of electromagnetic waves from rough surfaces.”
MacMillan, New York, 1963, pages 1-33 and 70-98.

Phong B.T. “Illumination for Computer Generated Images” 1973
“Illumination for computer generated pictures” ACM June 1975

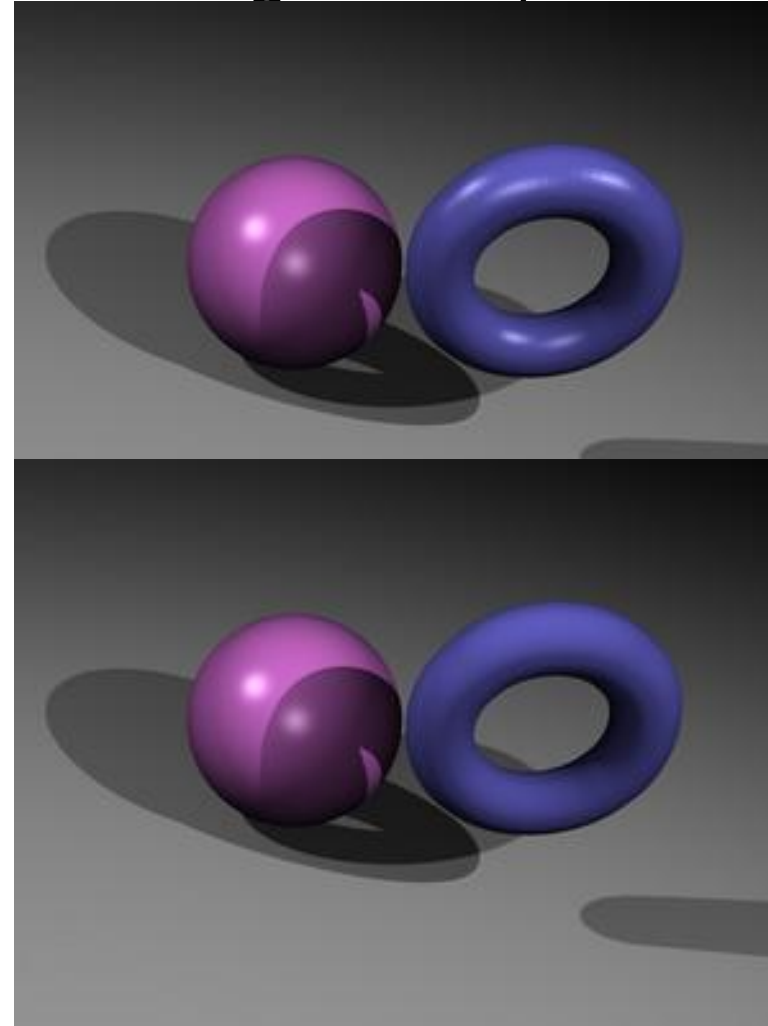
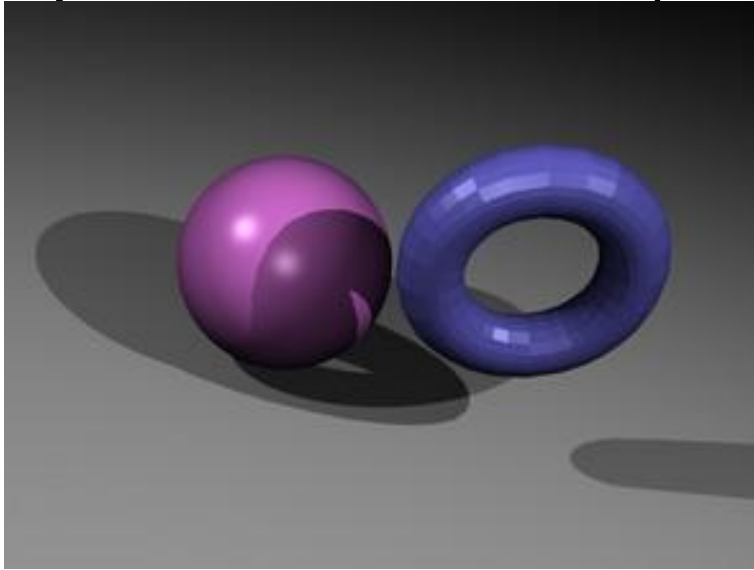
J. Blinn “Models of Light Reflection for Computer Synthesized Pictures”, James F. Blinn,
ACM Siggraph '77 Conference Proceedings

R. Cook and K. Torrance “A Reflectance Model for Computer Graphics” ACM Transactions
on Graphics, volume 1, number 1, January 1982 pages 7-24

G. Ward “Measuring and Modelling Anisotropic Reflection” , Computer Graphics 26, 2, July
1992

Ray tracing

- Specular reflection (Blinn-Phong model)

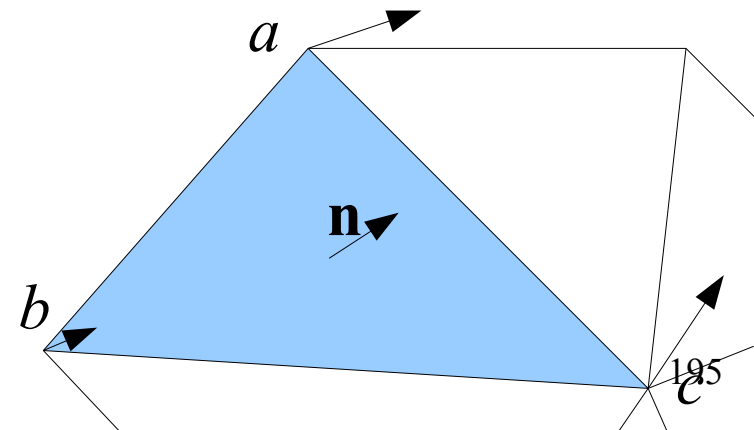


Ray tracing

- Specular reflection (Blinn-Phong model)
 - “Faceted” aspect
 - We must significantly increase the resolution to use the Phong model on geometries that are faceted to get a realistic result
 - Or It is necessary to know the exact geometry ...
 - It is very computationally expensive
 - There is an alternative: Phong interpolation

Ray tracing

- Phong interpolation
 - Faceted exact geometry = geometry + error term
 - The position error is small (we can barely see it)
 - On the contrary, the error on normals is considerable
 - The normal is constant on each facet !!
 - Phong proposes to linearly interpolate the normal in every facet from normal at the vertices of the facet
 - These are known if we have the exact geometry
 - If not, we take the "average" normal of connected facets



Ray tracing

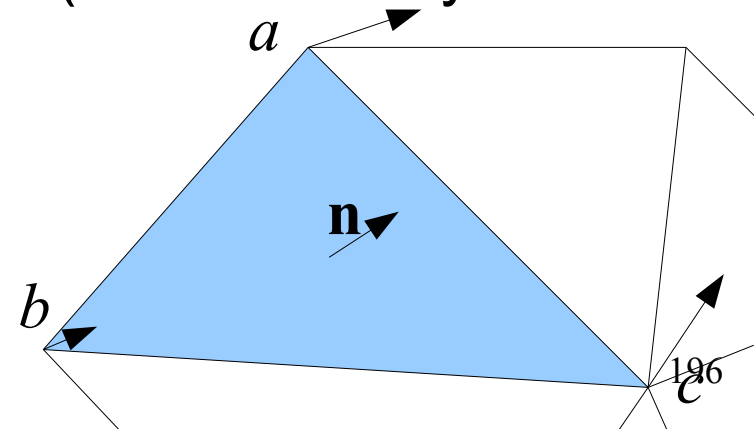
- Phong interpolation

- $$\mathbf{n}^* = (1-u)\mathbf{n}_c + (1-v)\mathbf{n}_b + (1-u-v)\mathbf{n}_a \quad (\text{Triangles})$$

- $$\mathbf{n}^* = (1-u)(1-v)\mathbf{n}_a + u(1-v)\mathbf{n}_b + uv\mathbf{n}_c + v(1-u)\mathbf{n}_d \quad (\text{Quads})$$

$$\mathbf{n} = \frac{\mathbf{n}^*}{\|\mathbf{n}^*\|}$$

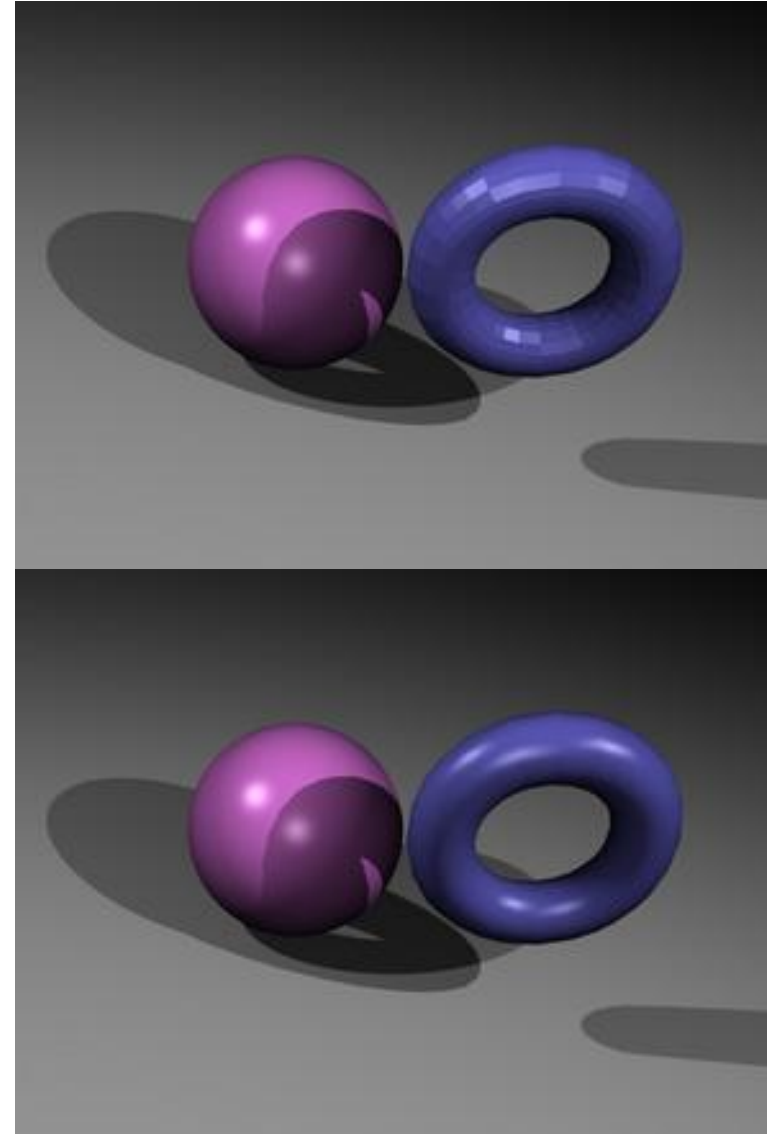
- u and v are calculated at the intersection of the ray and the facet
- $\mathbf{n}_a, \mathbf{n}_b, \mathbf{n}_c$ (, \mathbf{n}_d) are precalculated (before the ray tracing itself)



Ray tracing

- Phong interpolation

```
surface.shading(ray,point,normal,sourceS)
{
  colour= ambient // eventually black !
  For each source in sourceS
  {
    If not in shadow (cf previously)
    {
      v = -normalize(ray.direction)
      l = normalize(lamp.position-point)
      n=calculate_normal(point)
      calculate the shading (phong, lambertian, etc...)
      colour = colour + calculated shading
    }
  }
  return colour
}
```



Ray tracing

- Gouraud interpolation
 - Same idea as Phong interpolation
 - We're working on the color obtained at the vertices of the facet that are interpolated, instead of the normals
 - Does not work with the Phong shading!
 - Not used in ray tracing
 - Advantage: very fast

Ray tracing

- All in all ... it usually takes ambient, diffuse and Phong shading in the same model to have something natural.

- Perceived brightness is a sum :

$$\begin{aligned}L &= L_a + L_d + L_s \\ &= k_a I_a + k_d I \max(0, \mathbf{n} \cdot \mathbf{l}) + k_s I \max(0, \mathbf{n} \cdot \mathbf{h})^p\end{aligned}$$

- Sum over all visible light sources from the point

$$\begin{aligned}L &= L_a + \sum_i [(L_d)_i + (L_s)_i] \\ &= k_a I_a + \sum_i \left[k_d I_i \max(0, \mathbf{n} \cdot \mathbf{l}_i) + k_s I_i \max(0, \mathbf{n} \cdot \mathbf{h}_i)^p \right]\end{aligned}$$

Ray tracing

- Smooth surfaces
 - Example : mirror
 - Perfectly specular reflection
 - The phong model is too far from the reality (infinitesimal dimension if the point sources)
 - We can model this by starting a new ray.
 - The direction is the same as the one calculated in the model of Phong
 - The colour of the point is the color seen from that point in the direction of reflection
 - Some materials have a "glossy" appearance
 - Combination of a mirror behavior and Lambertian + ambient

$$L = L_a + L_d + L_m$$

Ray tracing

- Model for smooth surfaces $L_m = k_m I_i$
 - The intensity depends on the angle of incidence and material indices
 - Transparent dielectric materials (Fresnel - Snell)

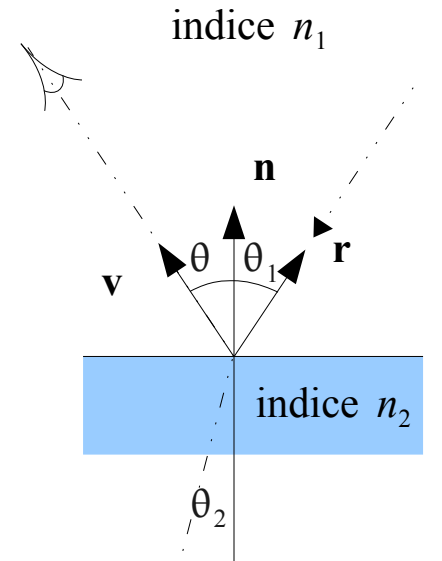
$$k_m^{perp} = \left(\frac{n_1 \cos \theta_1 - n_2 \cos \theta_2}{n_1 \cos \theta_1 + n_2 \cos \theta_2} \right)^2$$

Polarization perpendicular to the incidence plane

$$k_m^{para} = \left(\frac{n_2 \cos \theta_1 - n_1 \cos \theta_2}{n_2 \cos \theta_1 + n_1 \cos \theta_2} \right)^2$$

Polarization parallel to the incidence plane

$$k_m = (k_m^{perp} + k_m^{para}) / 2 \quad k_t = 1 - (k_m^{perp} + k_m^{para}) / 2$$



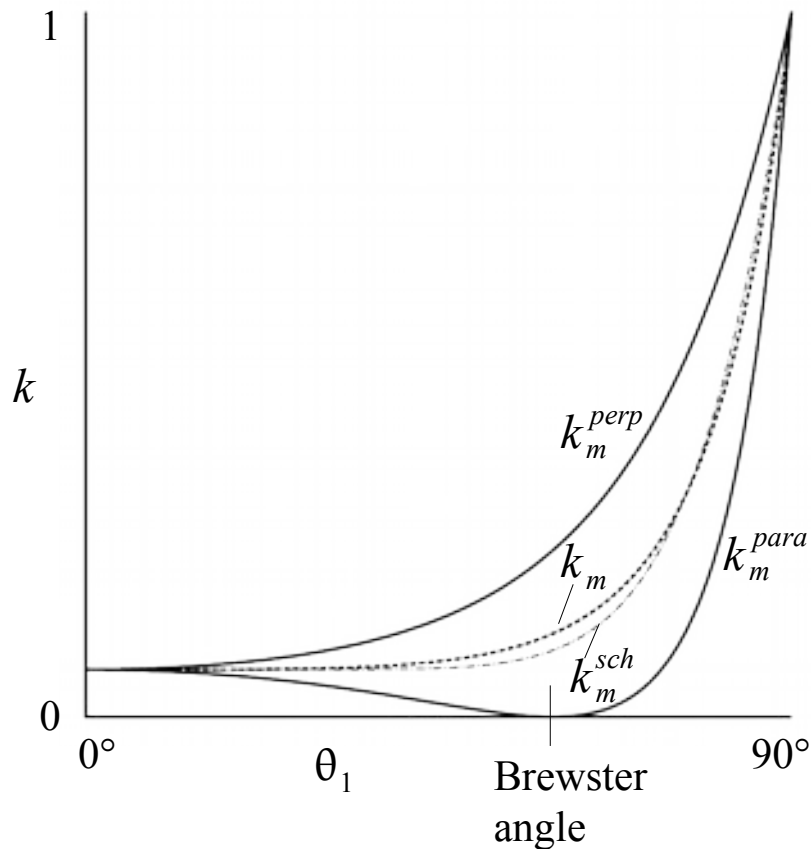
$$\mathbf{r} = \mathbf{v} + 2((\mathbf{n} \cdot \mathbf{v}) \mathbf{n} - \mathbf{v}) = 2(\mathbf{n} \cdot \mathbf{v}) \mathbf{n} - \mathbf{v}$$

$$n_1 \sin \theta_1 = n_2 \sin \theta_2$$

- Schlick's approximation : $k_m^{sch} = k_0 + (1 - \cos \theta_1)^5 (1 - k_0)$ with $k_0 = \left(\frac{n_2 - n_1}{n_2 + n_1} \right)^2$
- For electric conductors: almost total reflection (or constant $k_m = \text{cste}$ depending on the angle): a first approximation.

Ray tracing

- Model for smooth surfaces



$$k_m^{perp} = \left(\frac{n_1 \cos \theta_1 - n_2 \cos \theta_2}{n_1 \cos \theta_1 + n_2 \cos \theta_2} \right)^2$$

$$k_m^{para} = \left(\frac{n_2 \cos \theta_1 - n_1 \cos \theta_2}{n_2 \cos \theta_1 + n_1 \cos \theta_2} \right)^2$$

$$k_m = (k_m^{perp} + k_m^{para}) / 2$$

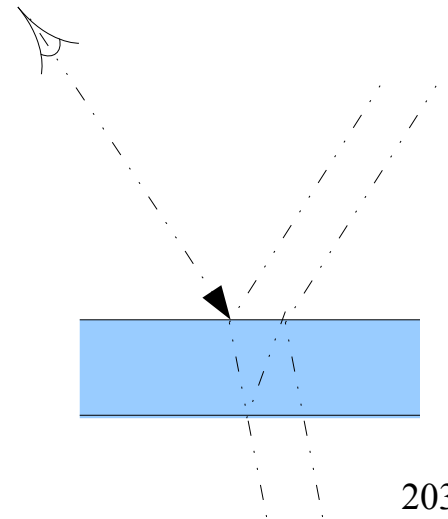
$$k_m^{sch} = k_0 + (1 - \cos \theta_1)^5 (1 - k_0)$$

Ray tracing

- Refraction/Reflection

- Each ray is split (if conditions are met)

- Without control, for each emitted ray, a large number of rays is generated recursively
 - We may limit that number by counting the number of reflections / refractions and stop after a certain value is attained
 - We may limit by "killing" the rays whose attenuation is above a certain threshold.



Ray tracing

- Plane that is slightly reflective
+ Lambert model



Ray tracing

- Justification of the Lambertian model

- Kubelka and al. 1931: model layers and does not consider the surface reflection, $\alpha(\lambda)$, $\beta(\lambda)$ are the fractions of the incident and reflected wavelength absorbed per unit length in the medium.

$$\rightarrow R_{\infty}(\lambda) = \frac{2 - \omega(\lambda) - 2\sqrt{1 - \omega(\lambda)}}{\omega(\lambda)} \quad \text{with } \omega(\lambda) = \frac{\beta(\lambda)}{\alpha(\lambda) + \beta(\lambda)}$$

- Reichmann 1971: includes the term of the surface reflection

$$\rightarrow R_B(\theta, \lambda) = (1 - R_s) \frac{C(\theta, \lambda)(1 - r_i(\lambda))(R_{\infty}(\lambda) - D(\theta))}{2(1 - r_i(\lambda)R_{\infty}(\lambda))\cos\theta}$$

$$\text{with } C(\theta, \lambda) = \frac{\omega(\lambda)\cos\theta(2\cos\theta + 1)}{1 - 4(1 - \omega(\lambda))\cos^2\theta} \quad \text{and } D(\theta) = \frac{2\cos\theta - 1}{2\cos\theta + 1}$$

- $r_i(\lambda)$ is the internal reflectance of the material (cf Orchard.1969)

- Complex ? Assume that the incident wave is not absorbed by the medium : so $\alpha(\lambda) \rightarrow 0$, $\beta(\lambda) \rightarrow 1$ and then $R_B(\theta, \lambda) = (1 - R_s)$: what is not reflected from the surface is sent in all directions independently of the orientation.

Reflectance : « ratio between the incident and reflected luminance »