Discrete Differential Geometry

Paolo Cignoni
paolo.cignoni@isti.cnr.it
http://vcg.isti.cnr.it/~cignoni

Normal

- Let's consider 2 manifold surface S in R^3

- Suppose to have a mapping $R^2 \rightarrow R^3$

  $S(u,v) \rightarrow R^3$

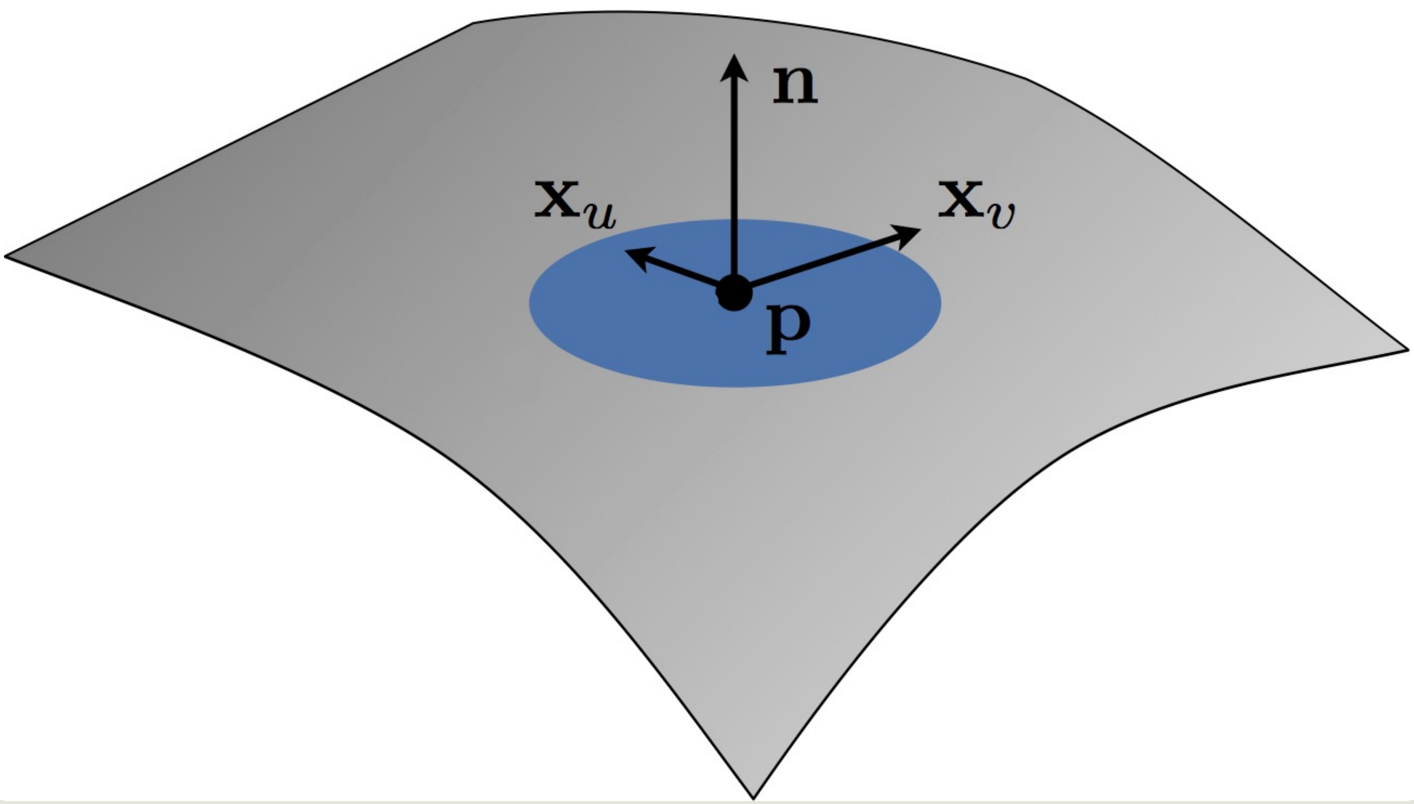
- Then we can define the normal for each point of the surface as:

$$\mathbf{n} = (\mathbf{x}_u \times \mathbf{x}_v)/\|\mathbf{x}_u \times \mathbf{x}_v\|$$

Where Xu and Xv are vectors on tangent space

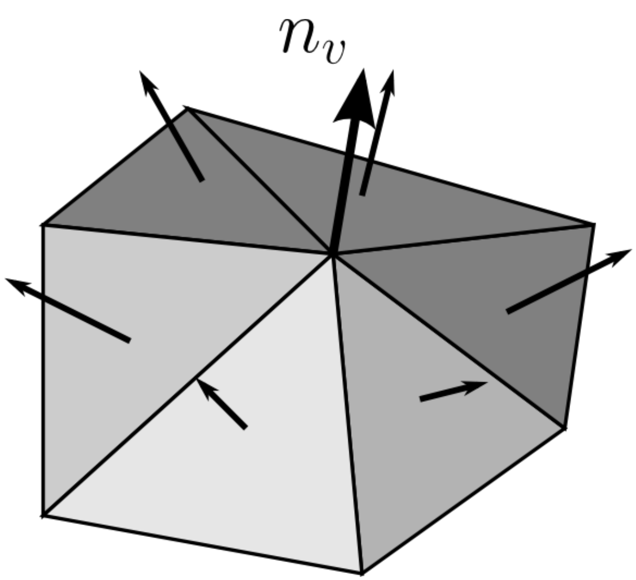
Normal

- Normal $\mathbf{n} = (\mathbf{x}_u \times \mathbf{x}_v)/\|\mathbf{x}_u \times \mathbf{x}_v\|$



Normals on triangle meshes

- Computed per-vertex and interpolated over the faces
- Common: consider the tangent plane as the average among the planes containing all the faces incident on the vertex

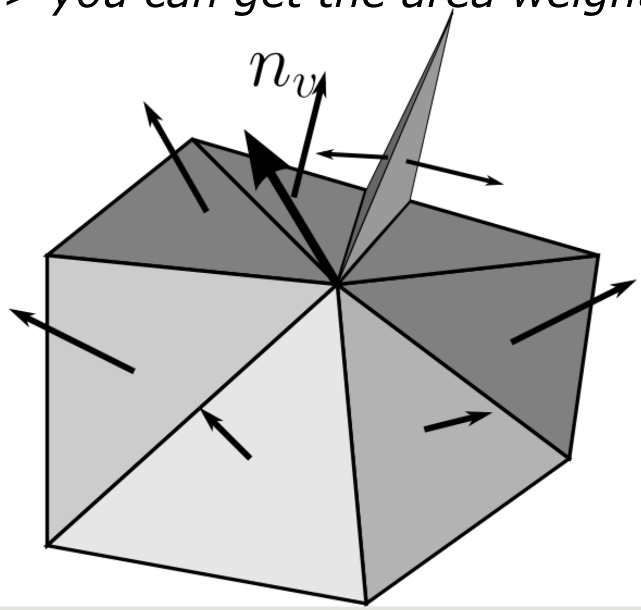


$$n_v = \frac{1}{\#N(v)} \sum_{f \in N(v)} n_f$$

$$N(v) = \{f : f \ coface \ of \ v\}$$

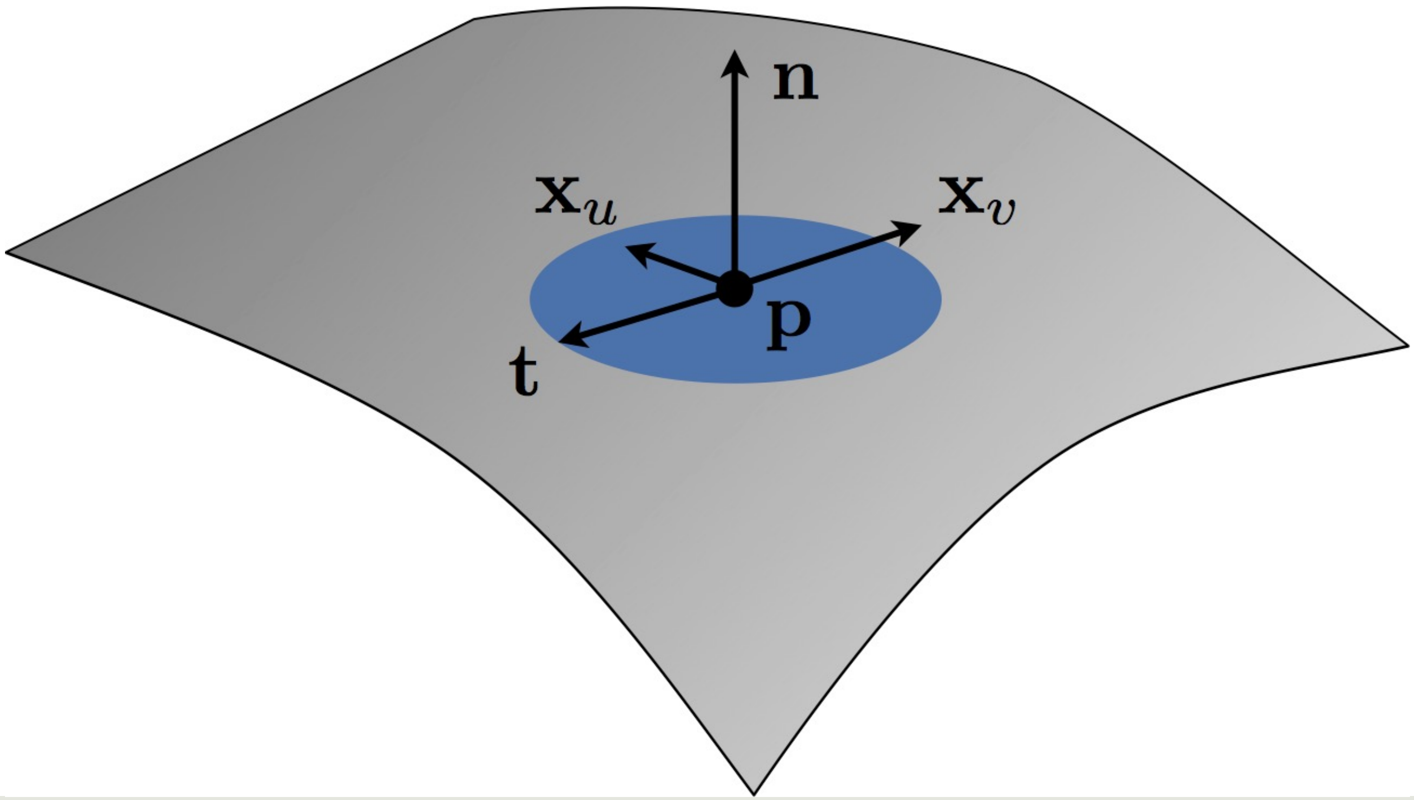
Normals on triangle meshes

- Does it work? Yes, for a "good" tessellation
  - Small triangles may change the result dramatically
  - Weighting by area, angle, edge len helps
    - Note: if you get the normal as cross product of adj edges, if you leave it un-normalized its length is twice the area of the triangle -> *you can get the area weighting for free*



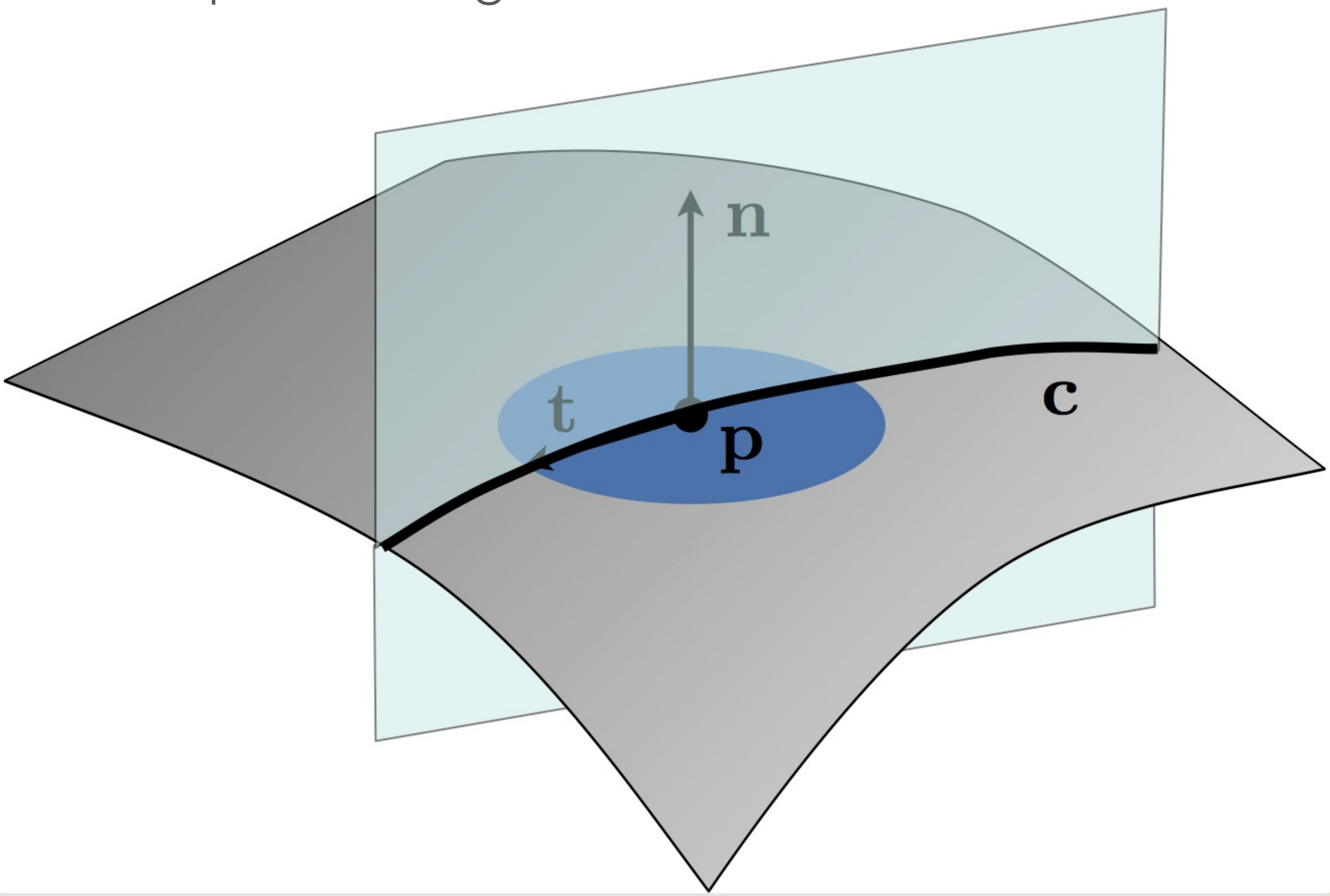
Curvature

- Define a tangent vector $\mathbf{t} = \cos\phi \frac{\mathbf{x}_u}{\|\mathbf{x}_u\|} + \sin\phi \frac{\mathbf{x}_v}{\|\mathbf{x}_v\|}$



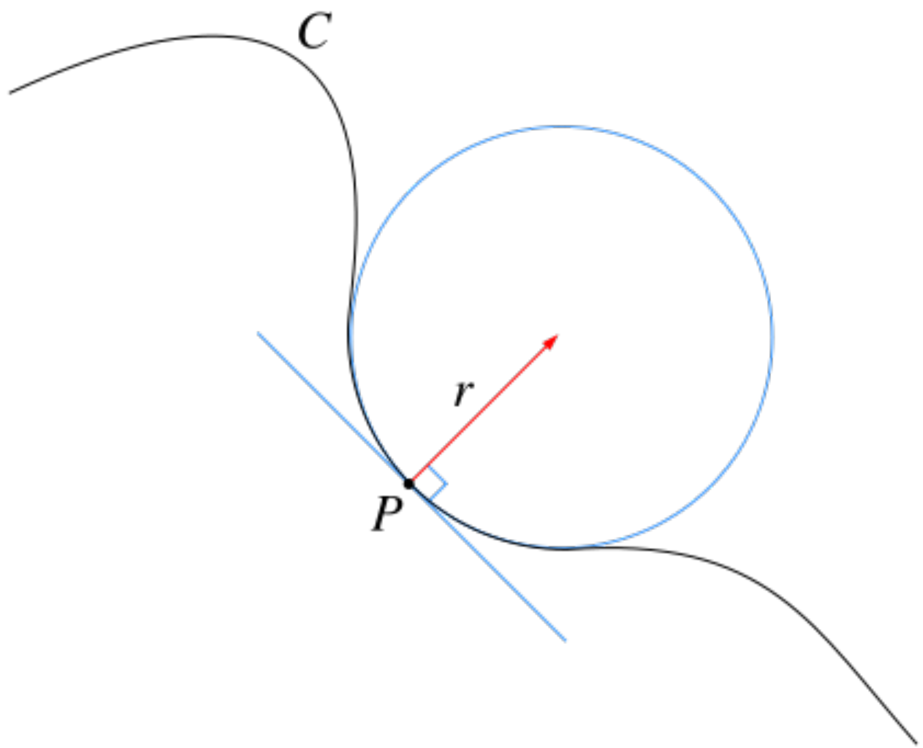
Curvature

- Consider the plane along n,t and the 2D curve defined on it



Curvature in 2D

- The curvature of C at P is then defined to be the reciprocal of the radius of osculating circle at point P.



The osculating circle of a curve C at a given point P is the circle that has the same **tangent** as C at point P as well as the same **curvature**.

Just as the tangent line is the line best approximating a curve at a point P, the osculating circle is the best circle that approximates the curve at P

Main curvature directions

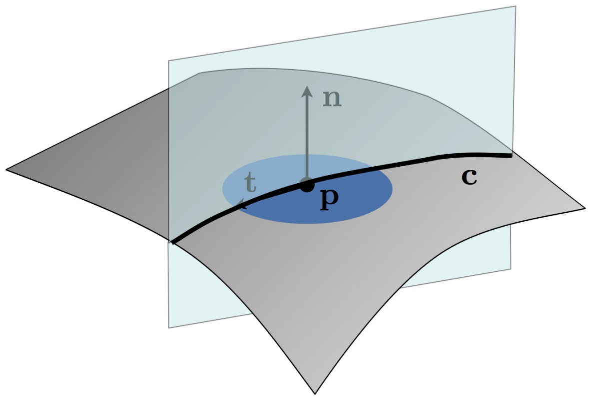
- For each direction **t** , we define a curvature value k.

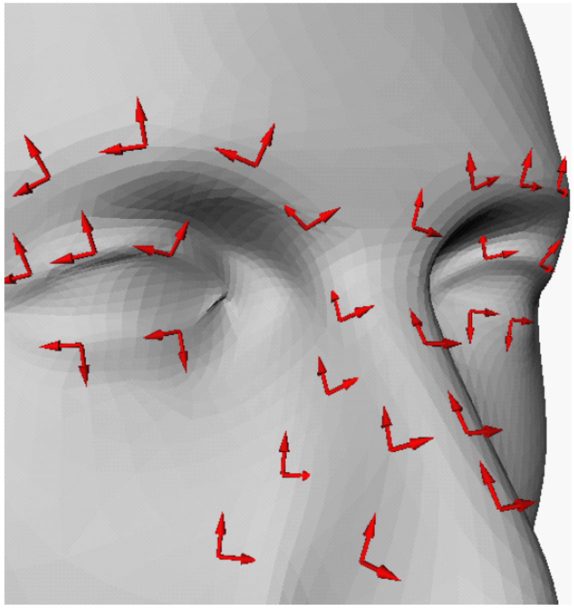
- Let's consider the two directions $\mathbf{k_1}$ and $\mathbf{k_2}$ where the curvature values k_1 and k_2 are **maximum** and **minimum**

- **Euler theorem**

$\mathbf{k_1}$ and $\mathbf{k_2}$ are perpendicular and curvature along a direction t making an angle θ with $\mathbf{k_1}$ is:

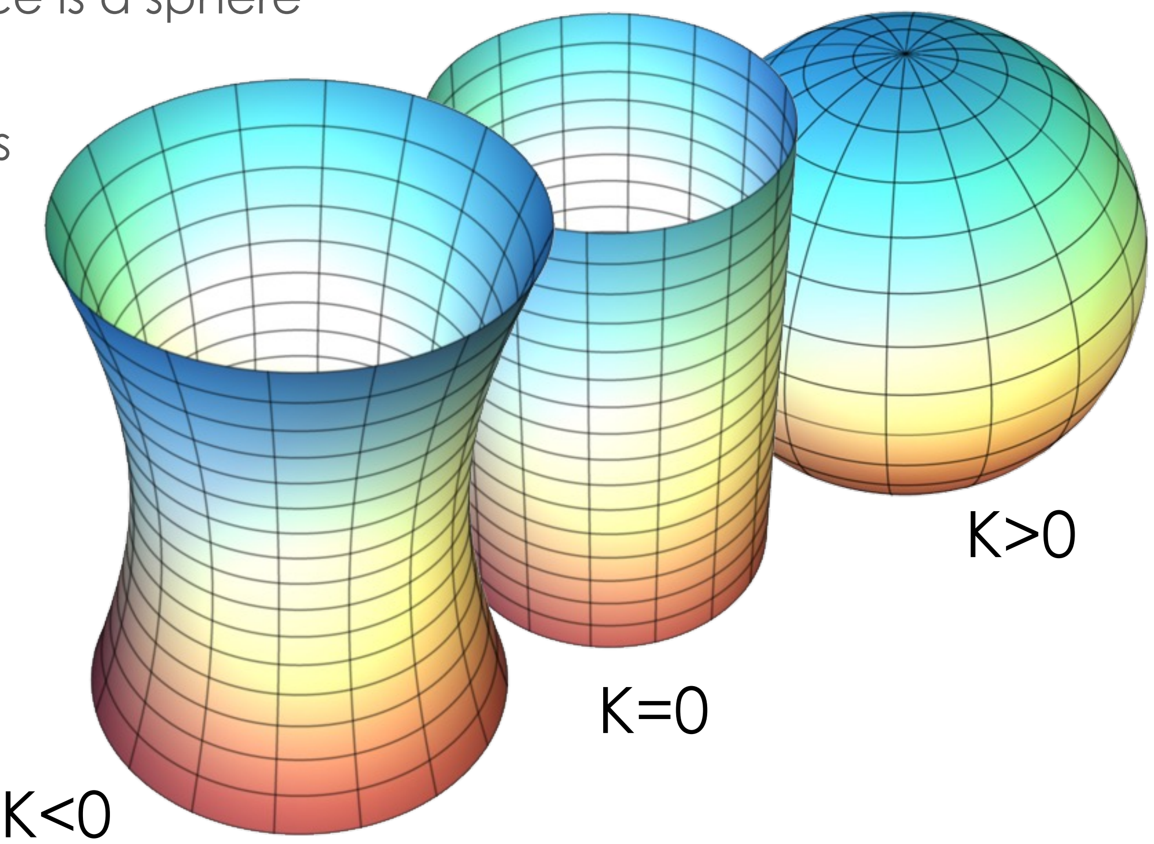
$$k_\theta = k_1 \cos^2\theta + k_2 \sin^2\theta$$





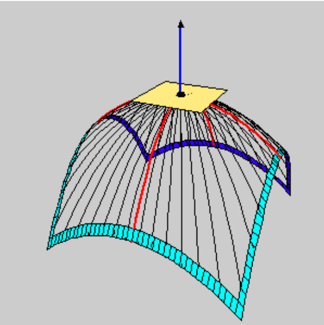
Gaussian curvature

- Defined as $K = k_1 \cdot k_2$
  - >0 when the surface is a sphere
  - 0 if locally flat
  - <0 for hyperboloids

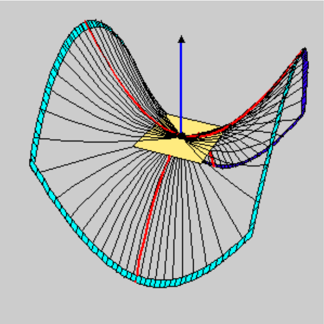


Gaussian curvature

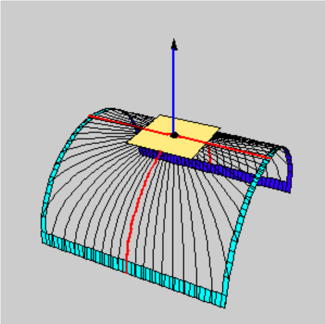
- A point x on the surface is called:
  - **elliptic** if $K > 0$
    (k_1 and k_2 have the same sign)
  - **hyperbolic** if $K < 0$
    (k_1 and k_2 have opposite sign)
  - **parabolic** if $K = 0$
    (exactly one of k_1 and k_2 is zero)
  - **planar** if $K = 0$
    (equivalently $k_1=k_2=0$).



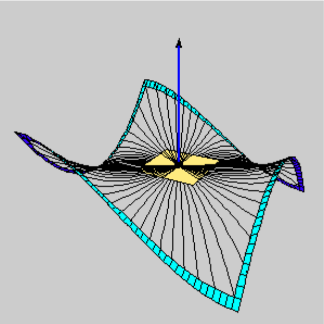
elliptic



hyperbolic

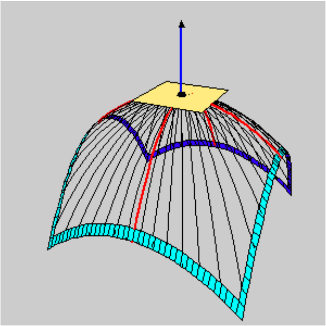


parabolic

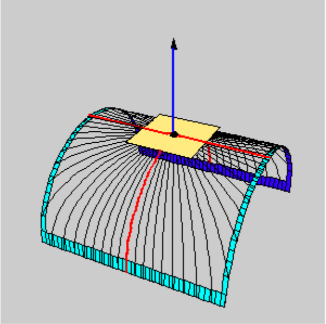


planar

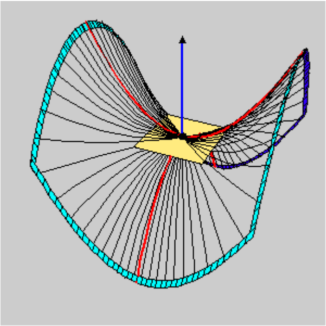
Different classes distributed on the surface



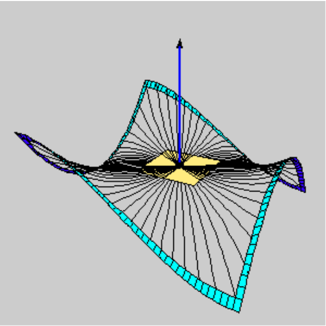
elliptic



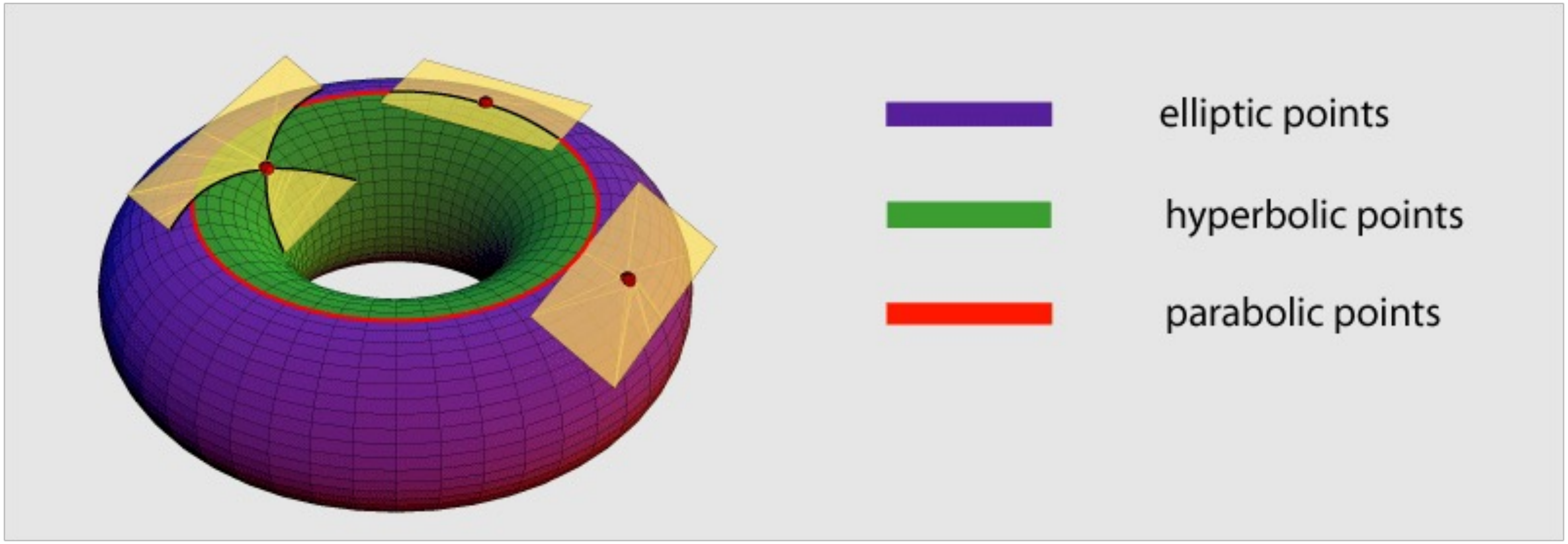
parabolic



hyperbolic

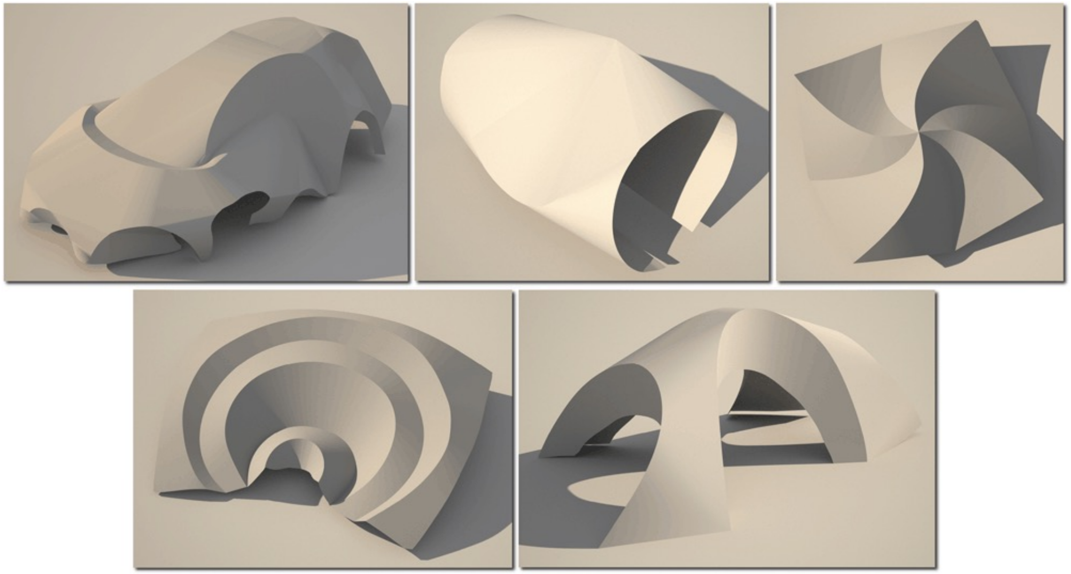


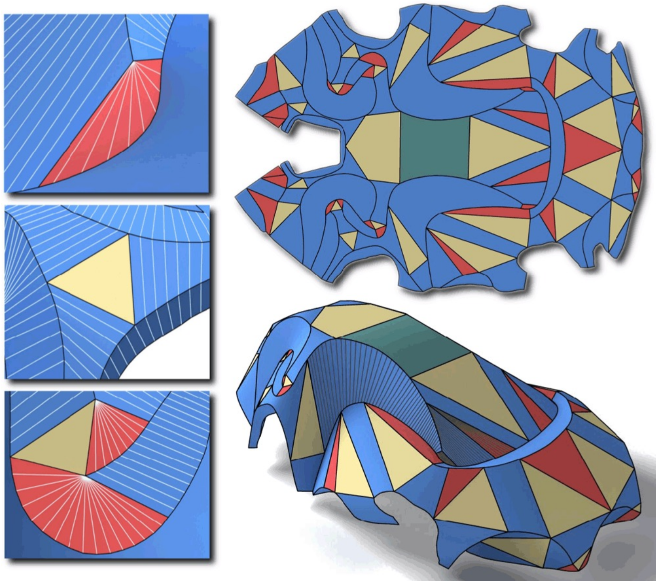
planar



Developable surfaces

- Developable surface ⇔ $K = 0$
- Flattening introduce no distortion



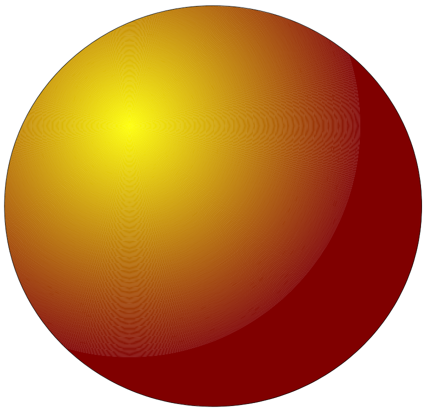


Gaussian Curvature: intrinsic / extrinsic

- Gaussian curvature is an **intrinsic** properties of the surface (even if we defined in an extrinsic way)

- It is possible to determine it by moving on the surface keeping the geodesic distance constant to a radius r and measuring the circumference C(r) :

$$K = \lim_{r \to 0} \frac{6\pi r - 3C(r)}{\pi r^3}$$



K>0

K<0

Mean Curvature

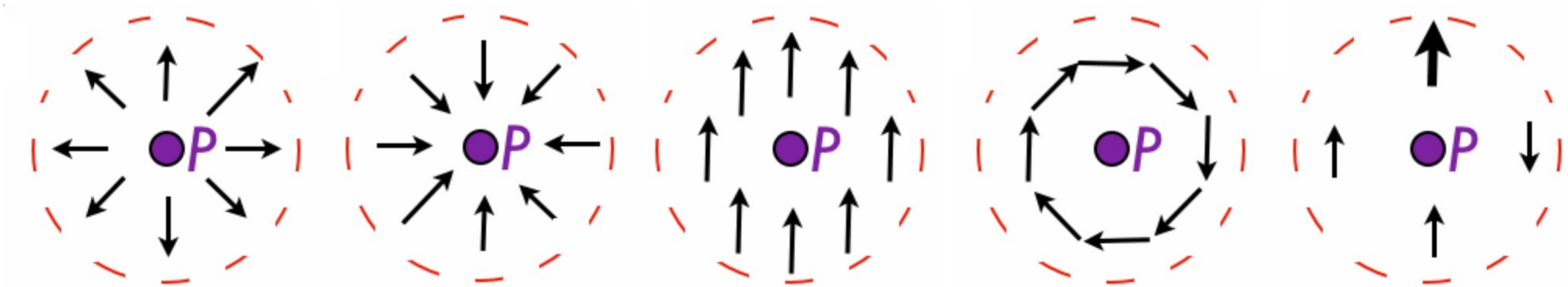
- $H=(k_1+k_2)/2$

- Measure the **divergence** of the normal in a local neighborhood of the surface

- The **divergence** **div_s** is an operator that measures a vector field's tendency to originate from or converge upon a given point

Divergence

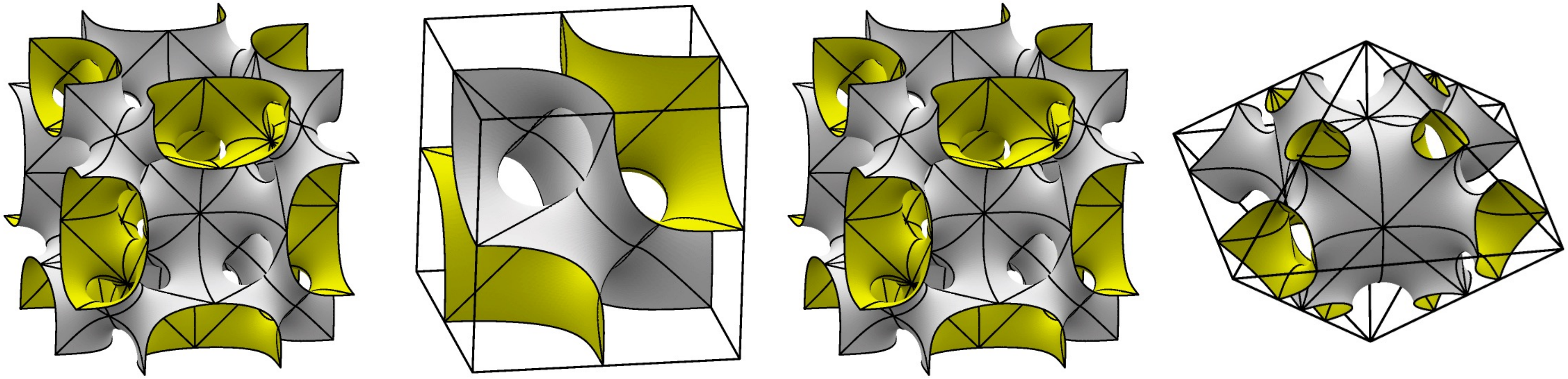
- Imagine a vector field represents water flow:
  - If **div**$_s$ is a **positive** number, then water **is flowing out** of the point.
  - If **div**$_s$ is a **negative** number, then water is **flowing into** the point.



$div_s > 0$ $div_s > 0$ $div_s = 0$ $div_s = 0$ $div_s > 0$

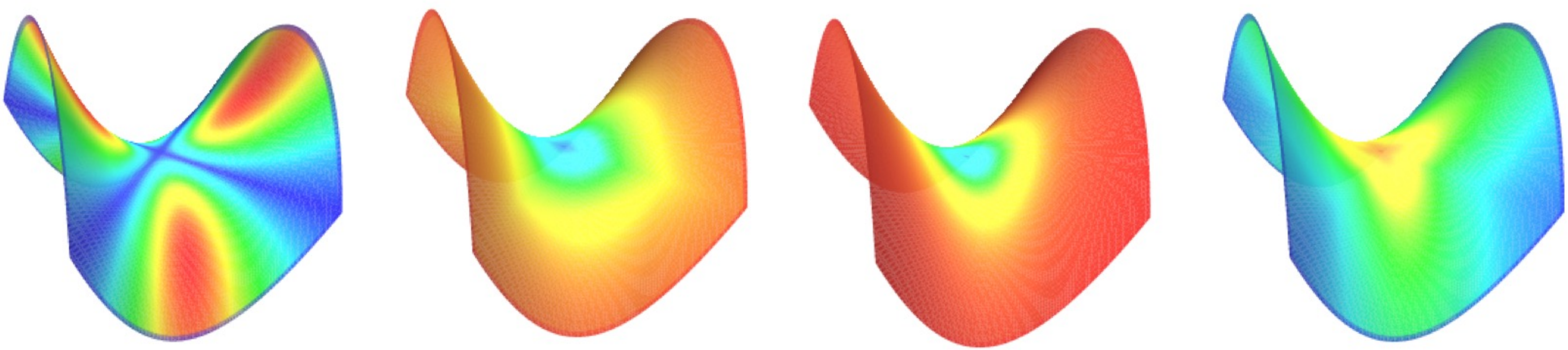
Minimal surface and minimal area surfaces

- A surface is **minimal** iff H=0 everywhere
- All surfaces of minimal AREA (subject to boundary constraints) have H= 0 (not always true the opposite!)
- The surface tension of an interface, like a soap bubble, is proportional to its mean curvature



Then… finally…

- Red > 0 Blue < 0 , not the same scale



mean gaussian min max

Some math…. Gradient and divergence

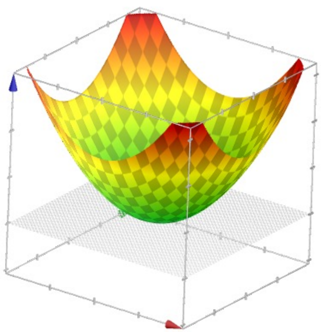
- Given a function **F: R^2→R** (our surface) the **gradient** of **F** is the vector field **∇F:R^2→R^2** defined by the partial derivatives:

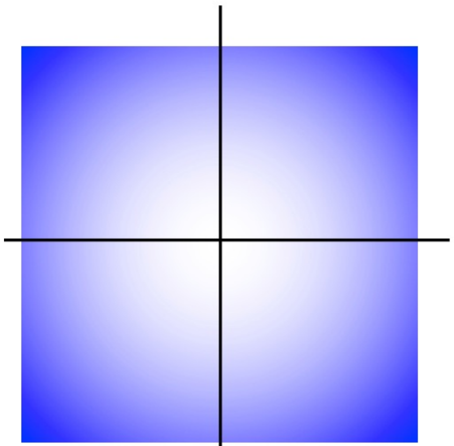
$$\nabla F(x, y) = \left(\frac{\partial F}{\partial x}, \frac{\partial F}{\partial y} \right)$$

- **Intuitively**: At the point p_0, the vector $\nabla F(p_0)$ points in the **direction of greatest change of F**.

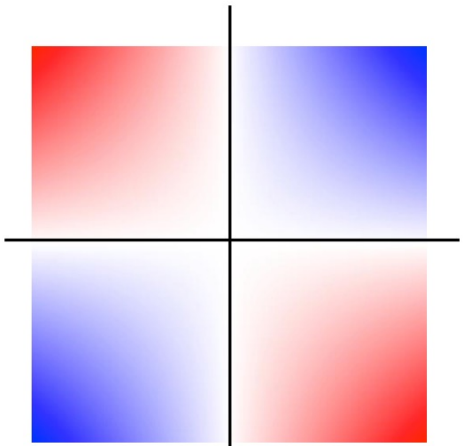
Some math…. Gradient and divergence

- Example ：

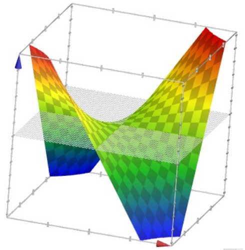


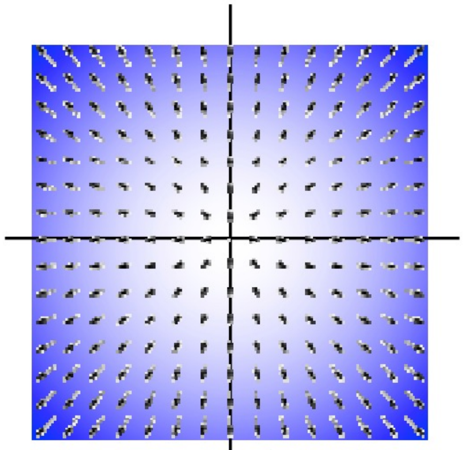


$F(x,y) = x^2 + y^2$

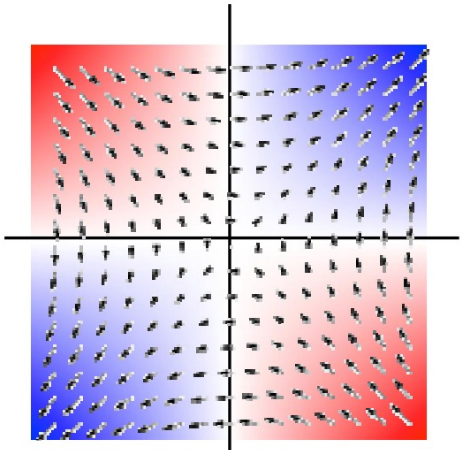


$F(x,y) = xy$





$\nabla F(x,y) = (2x, 2y)$



$\nabla F(x,y) = (y, x)$

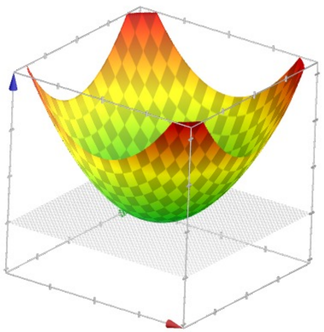
Some math…. Gradient and divergence

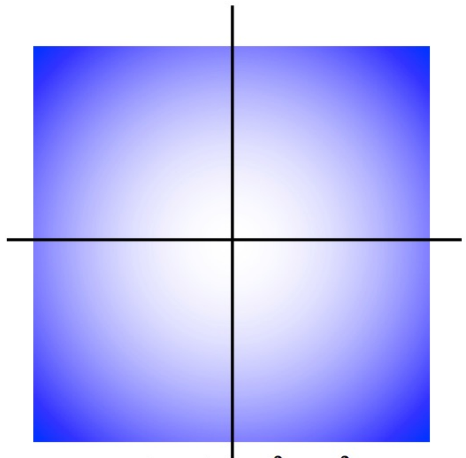
- Given a function **$F(F_1,F_2)$: $R^2 \rightarrow R^2$** the **divergence** of **F** is the function **div:$R^2 \rightarrow R$** defined as:

$$\text{div } F(x,y) = \partial F_1/\partial x + \partial F_2/\partial y$$

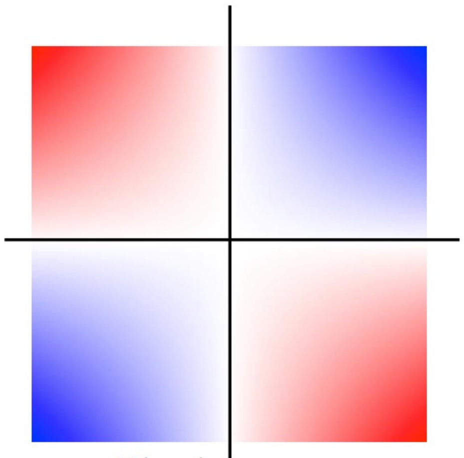
- **Intuitively**: At the point p_0, the divergence div $F(p_0)$ is a measure of the extent to which the flow (de)compresses at p_0.

Some math…. Gradient and divergence

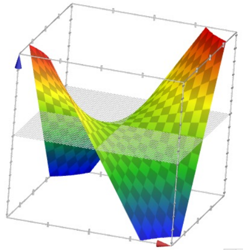


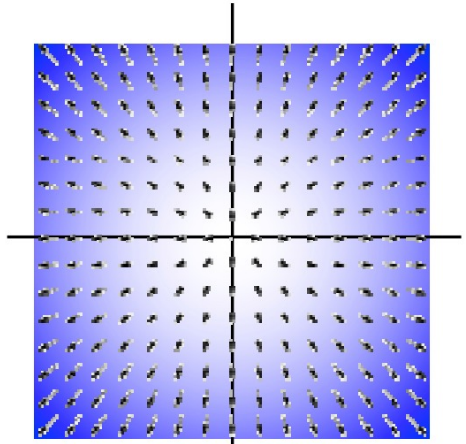


$F(x,y) = x^2 + y^2$



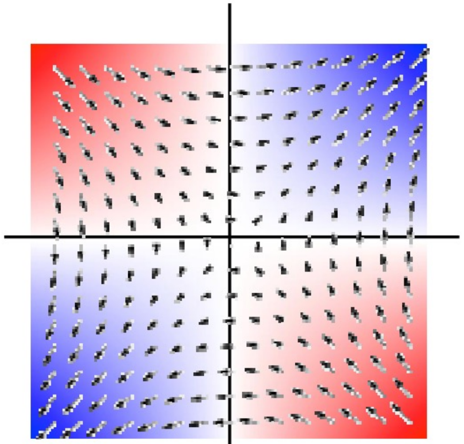
$F(x,y) = xy$





$\nabla F(x,y) = (2x, 2y)$

div $\nabla F(x,y) = 4$



$\nabla F(x,y) = (y, x)$

div $\nabla F(x,y) = 0$

Some math…. Laplacian

- Given a function $\mathbf{F(F_1,F_2): R^2 \rightarrow R}$
the **Laplacian** of **F** is the function $\mathbf{\Delta F: R^2 \rightarrow R}$ defined by the divergence of the gradient of the partial derivatives:

$$\Delta F = \mathrm{div}(\nabla F(x,y)) = \partial^2 F/\partial x^2 + \partial^2 F/\partial y^2$$

- **Intuitively**: The Laplacian of F at the point p_0 measures the extent to which the value of F at p_0 differs from the average value of F its neighbors.

Discrete Differential Operators

- Assumption: Meshes are piecewise linear approximations of smooth surfaces

- Approach: Approximate differential properties at point x as spatial average over local mesh neighborhood $N(x)$, where typically
  - x = mesh vertex
  - $N(x)$ = n-ring neighborhood (or local geodesic ball)

Discrete Laplacian

- Uniform discretization

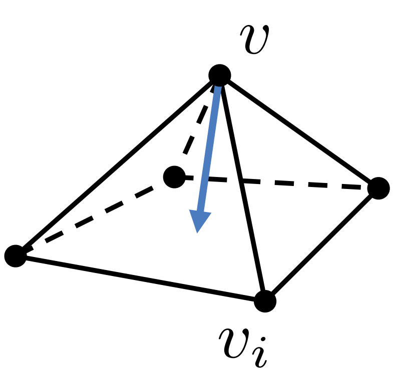
$$\Delta_{uni} f(v) := \frac{1}{|\mathcal{N}_1(v)|} \sum_{v_i \in \mathcal{N}_1(v)} \left(f(v_i) - f(v) \right)$$

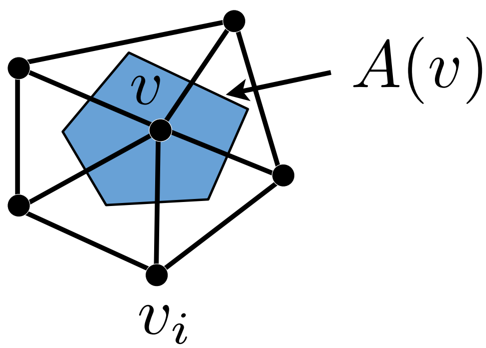
  - depends only on connectivity → simple and efficient
  - bad approximation for irregular triangulations

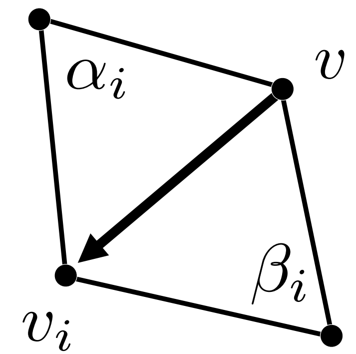
Discrete Laplacian

- Cotangent formula

$$\Delta_S f(v) := \frac{2}{A(v)} \sum_{v_i \in \mathcal{N}_1(v)} \left(\cot \alpha_i + \cot \beta_i\right)\left(f(v_i) - f(v)\right)$$



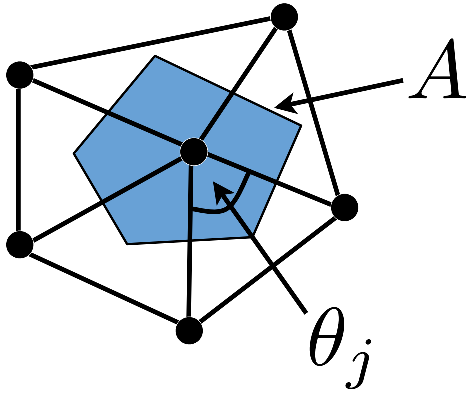




Discrete Curvatures

- Mean Curvature $H = \|\Delta_{\mathcal{S}} \mathbf{x}\|$

- Gaussian Curvature $G = (2\pi - \sum_j \theta_j)/A$



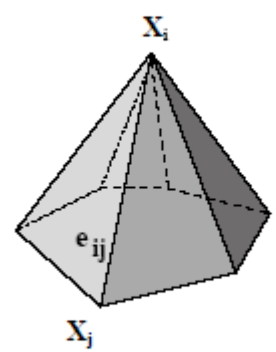
- Principal Curvatures

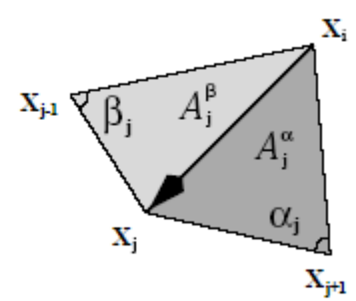
$$\kappa_1 = H + \sqrt{H^2 - G} \qquad \kappa_2 = H - \sqrt{H^2 - G}$$

Mean curvature on a triangle mesh

$$H(p) = \frac{1}{2A}\sum(cot\alpha_i + cot\beta_i)\,\|p - p_i\|$$

where α_j and β_j are the two angles opposite to the edge in the two triangles having the edge e_{ij} in common
A is the sum of the areas of the triangles





Gaussian curvature on a triangle mesh

- It's the *angle defect* over the area
- $$\kappa_G(v_i) = \frac{1}{3A}\left(2\pi - \sum_{t_j\, adj\, v_i} \theta_j\right)$$

- **Gauss-Bonnet Theorem:** The integral of the Gaussian Curvature on a closed surface depends on the Euler number

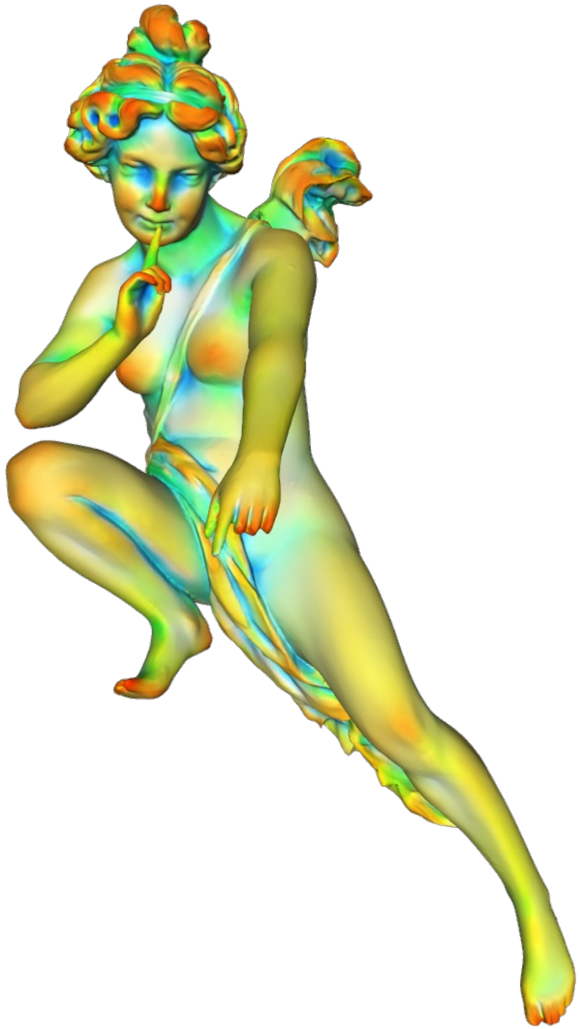
$$\int_S \kappa_G = 2\pi\chi$$

Discrete Curvatures

- Problems:
  - Depends on triangulation!
  - Very sensitive to Noise…

Curvature via Surface Fitting

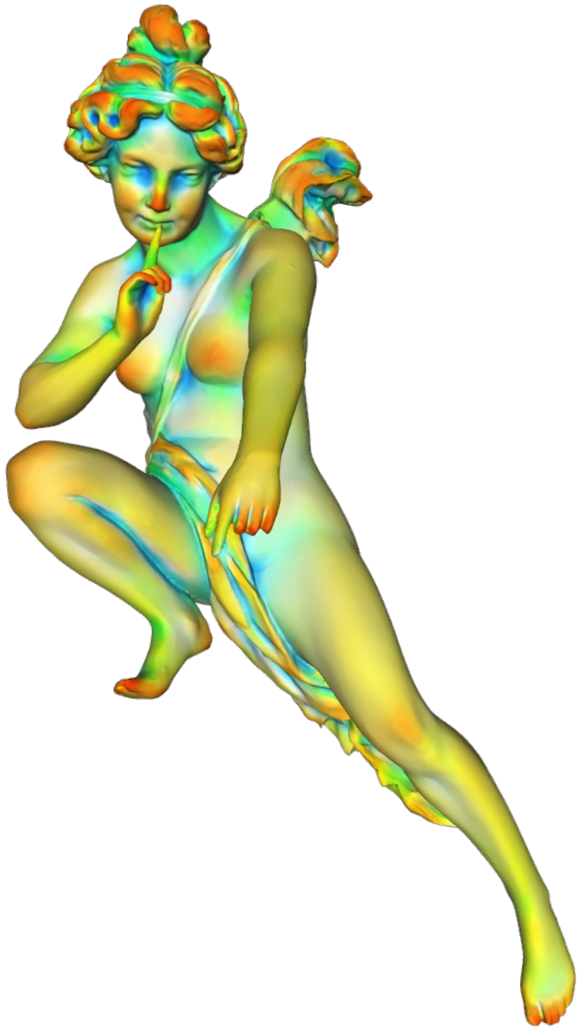
- The radius r of the neighborhood of each point p is used as a scale parameter
  - 1. gather all faces in a local neighborhood of radius r
  - 2. set an axis $\mathbf{w} = \frac{1}{n_v}\sum_{i=1}^{n}\mathbf{n}_i$
- where n_v is the number of vertices gathered and n_i is the surface normal at each such vertex



Curvature via Surface Fitting

- 3. discard all vertices v_i such that $n_i \cdot w < 0$
- 4. set a local frame (u,v,w) where u and v are any two orthogonal unit vectors lying on the plane orthogonal to w, and such that the frame is right-handed
- 5. express all vertices of the neighborhood in such a local frame with origin at $\mathbf{p}$
- 6. fit to these points a polynomial of degree two through $\mathbf{p}$ (least squares fitting)

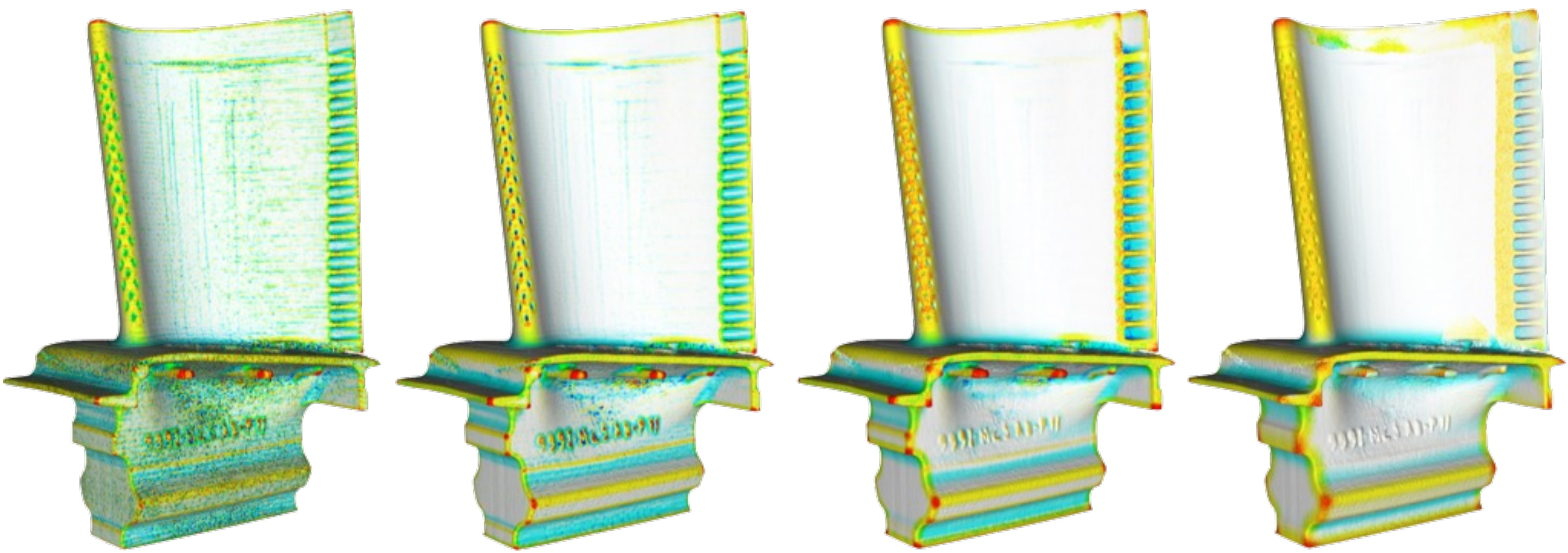
$$f(u,v) = au^2 + bv^2 + cuv + du + ev$$



- Curvatures at $\mathbf{p}$ are computed **analytically via first and second fundamental forms** of f at the origin

curvature via surface fitting

- Curvatures extracted at different scales



Screen Space Mean Curvature

```
// License: CC0 (http://creativecommons.org/publicdomain/zero/1.0/)
#extension GL_OES_standard_derivatives : enable

varying vec3 normal;
varying vec3 vertex;

void main() {
  vec3 n = normalize(normal);

  // Compute curvature
  vec3 dx = dFdx(n);
  vec3 dy = dFdy(n);
  vec3 xneg = n - dx;
  vec3 xpos = n + dx;
  vec3 yneg = n - dy;
  vec3 ypos = n + dy;
  float depth = length(vertex);
  float curvature = (cross(xneg, xpos).y - cross(yneg, ypos).x) * 4.0 / depth;

  // Compute surface properties
  vec3 light = vec3(0.0);
  vec3 ambient = vec3(curvature + 0.5);
  vec3 diffuse = vec3(0.0);
  vec3 specular = vec3(0.0);
  float shininess = 0.0;

  // Compute final color
  float cosAngle = dot(n, light);
  gl_FragColor.rgb = ambient +
    diffuse * max(0.0, cosAngle) +
    specular * pow(max(0.0, cosAngle), shininess);
}
```

- Known effect as Cavity Shading