Bump Mapping for Triangle Meshes

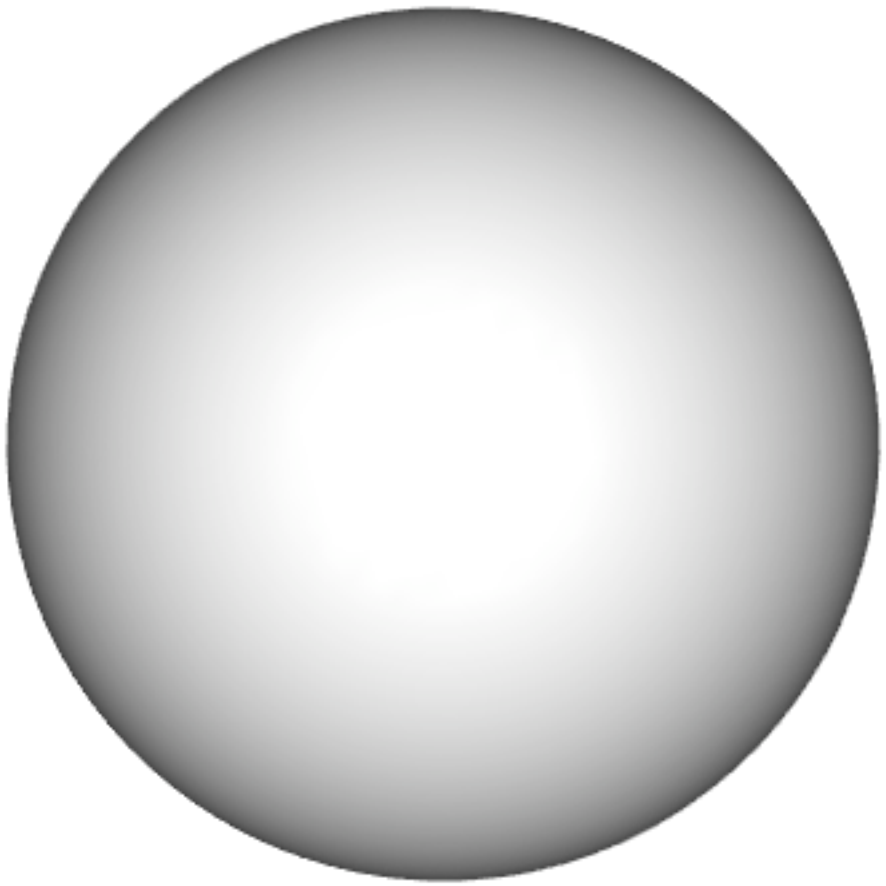
by Hao Li

A long time ago, in 1978...

... bump mapping was born

courtesy by ZBrush

So far, for meshes

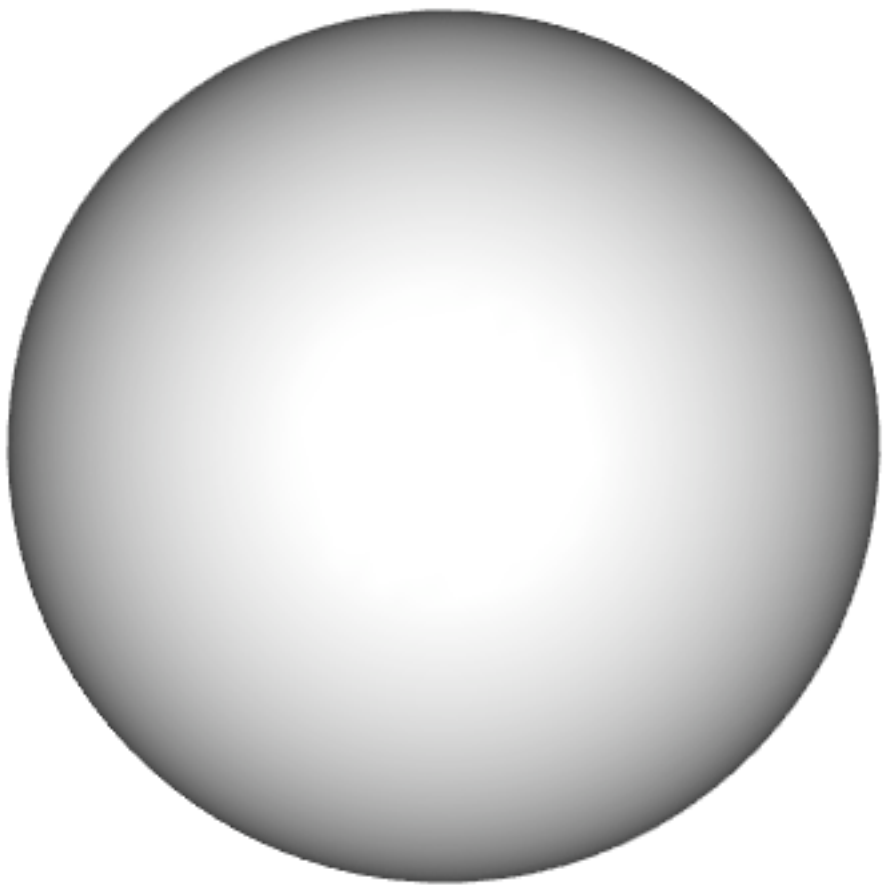


vertex normal interpolation

↓

smooth shading

So far, for meshes



vertex normal interpolation

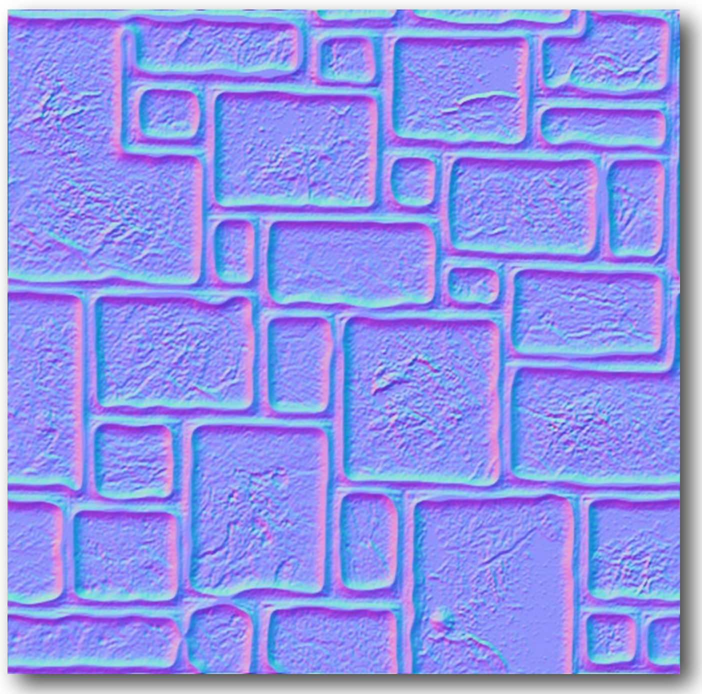
↓

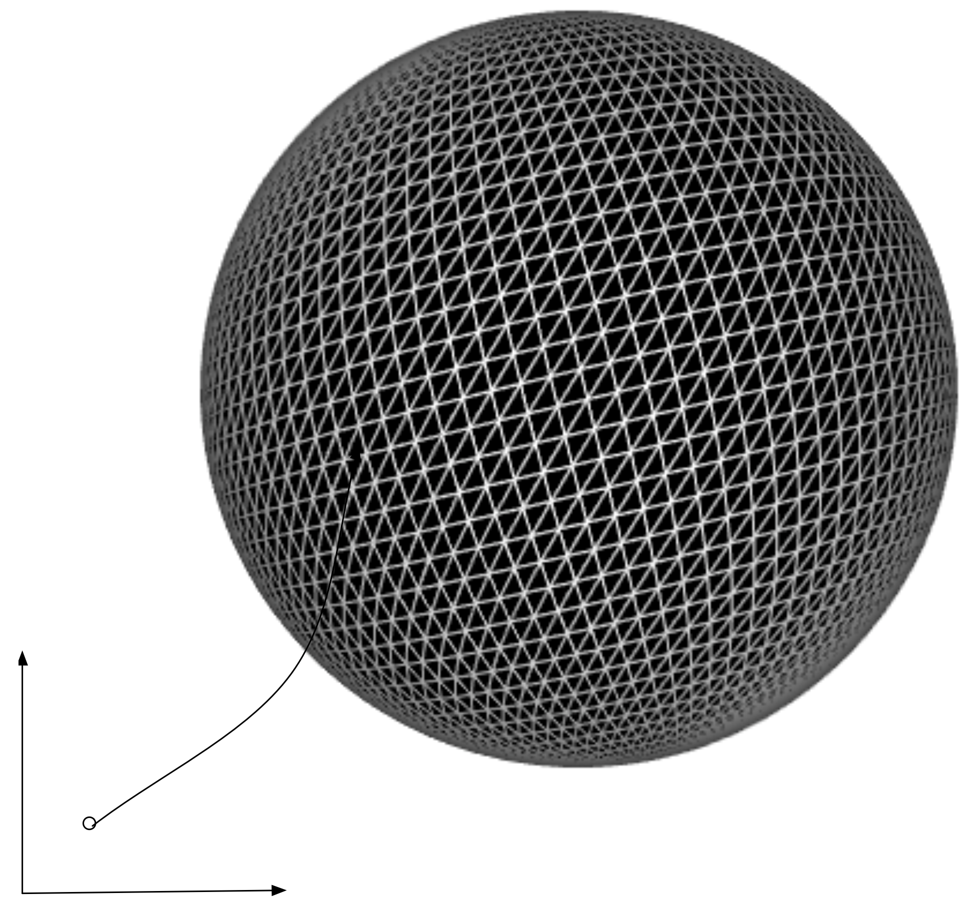
smooth shading

What about
accessing **textures** to modify **surface normals**...

Task

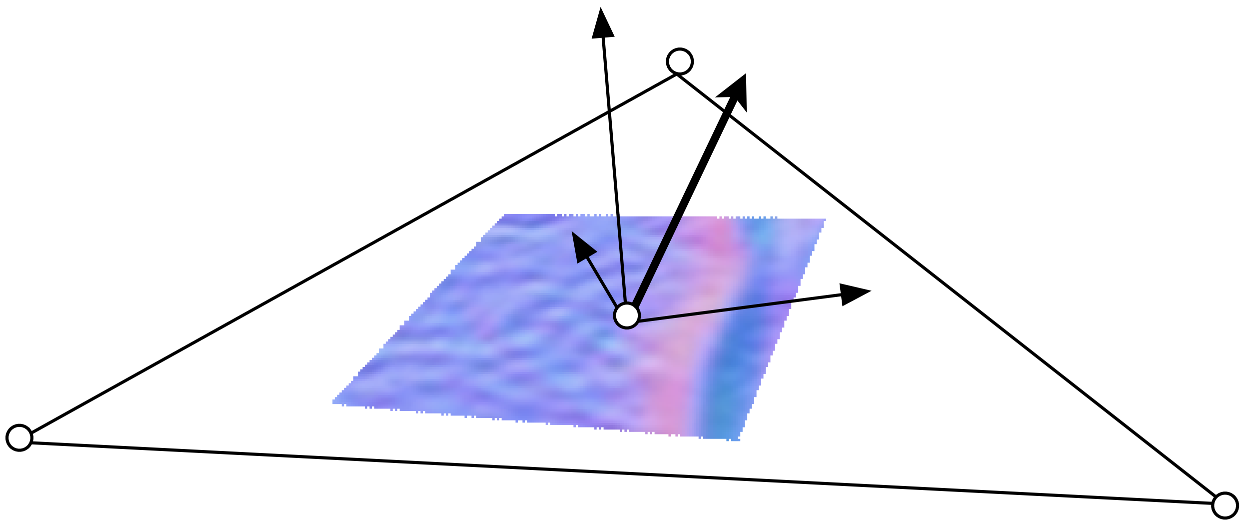
Use **bump map normals** given a **parametrized mesh**



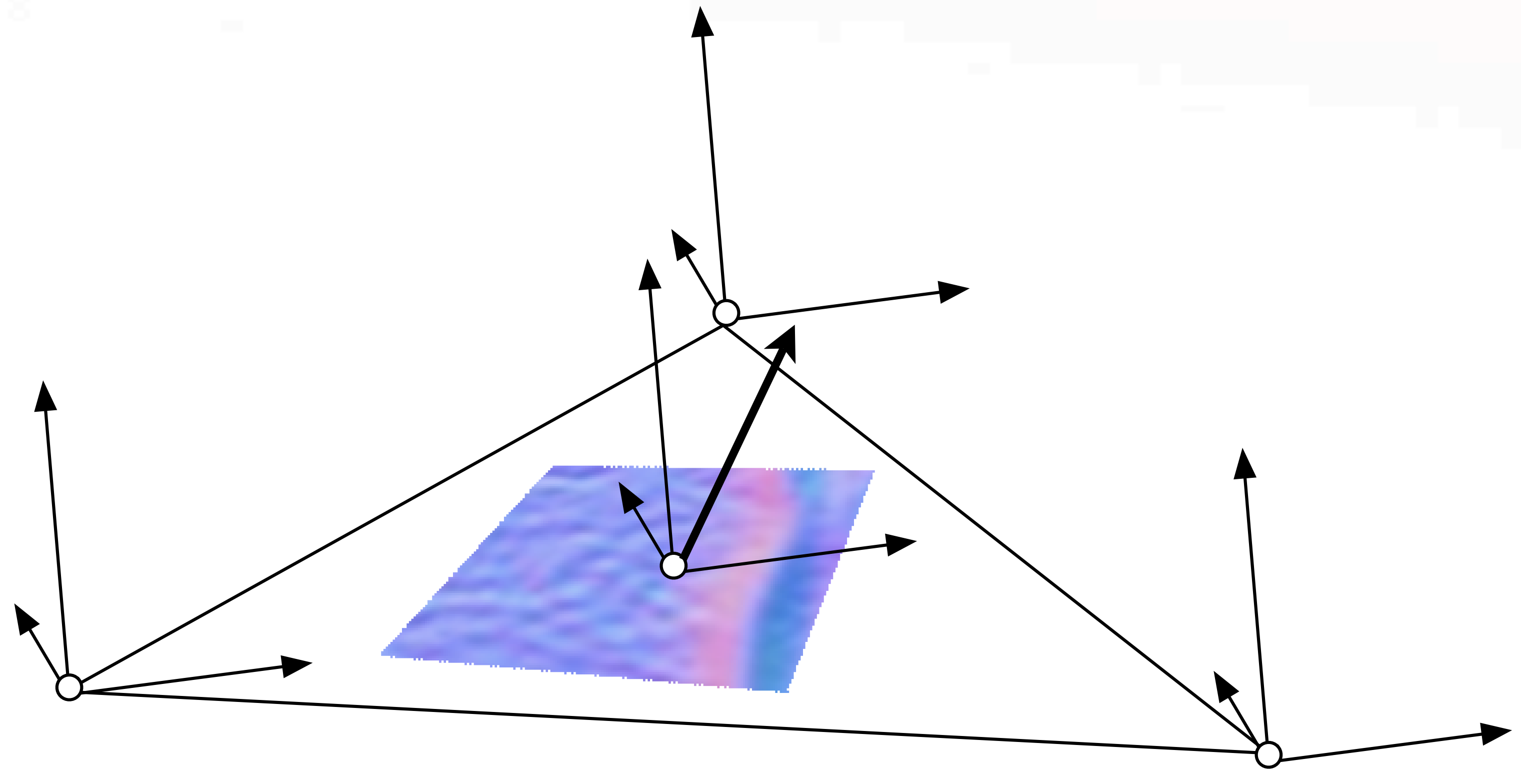


$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} \in \mathbb{R}^2$$

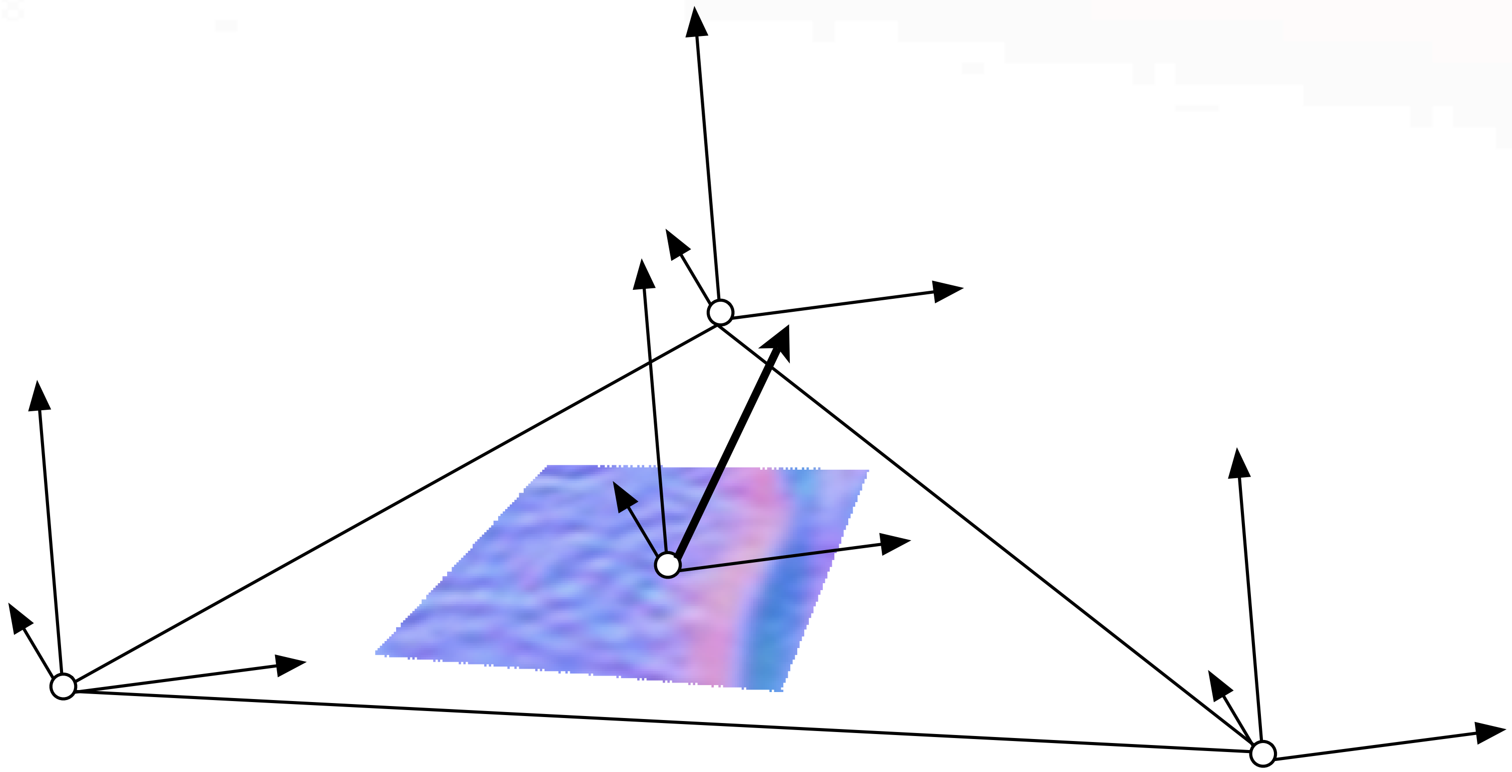
Bump map normals
are defined in a **local coordinate frame**
inside a triangle



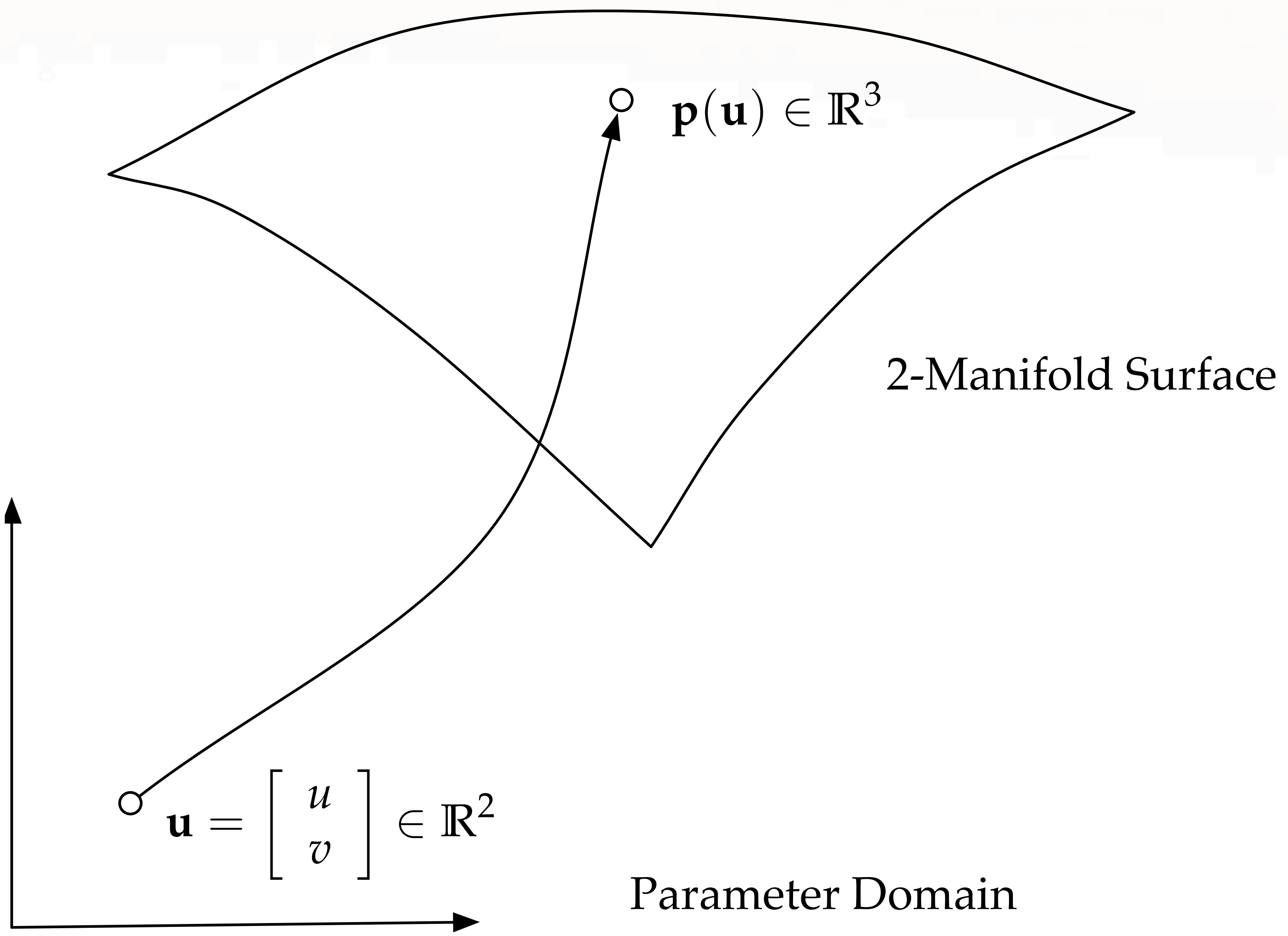
We have **positions, normals** and **parameters** of the triangle corners



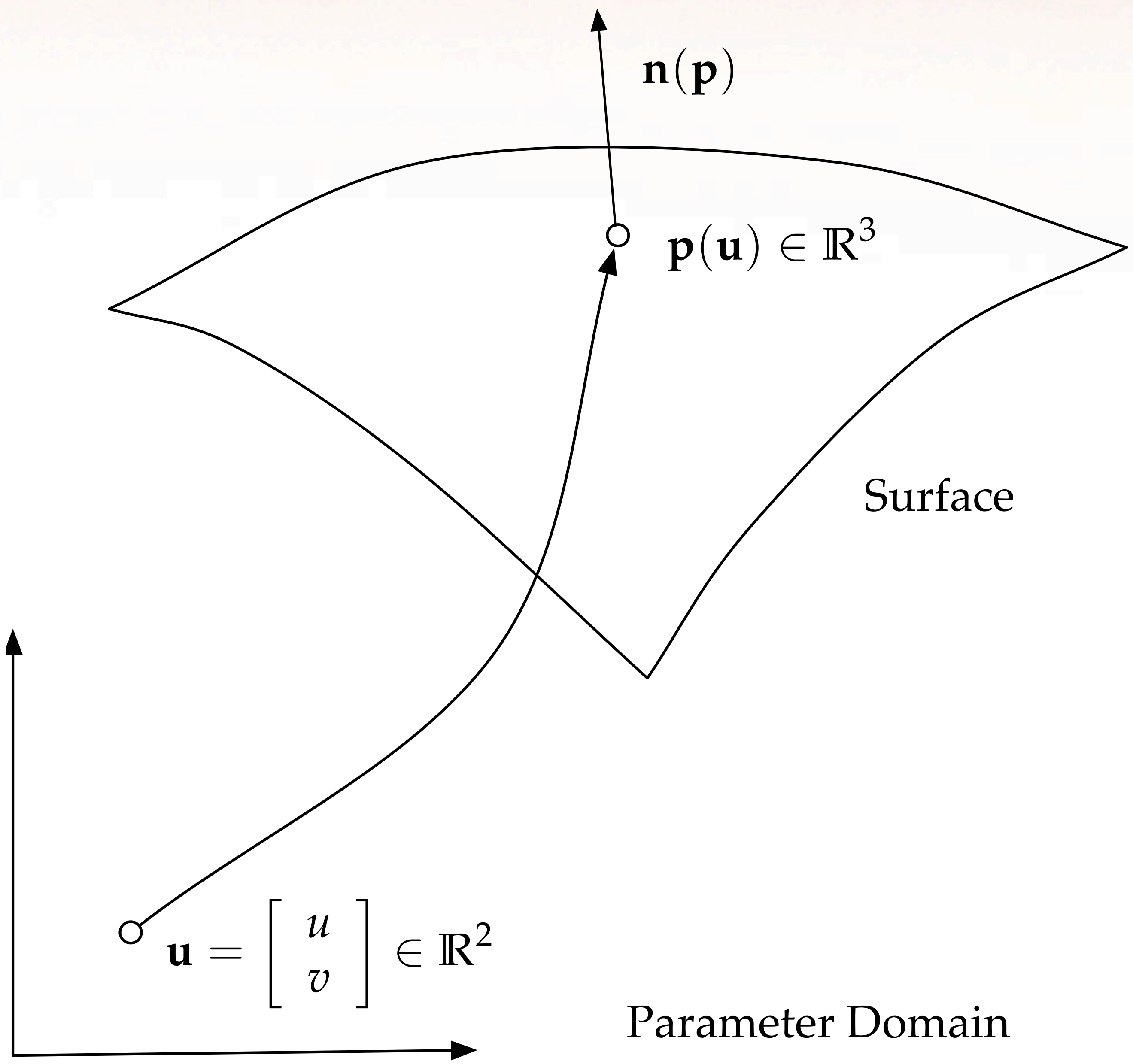
How do we obtain the **coordinate frames**?



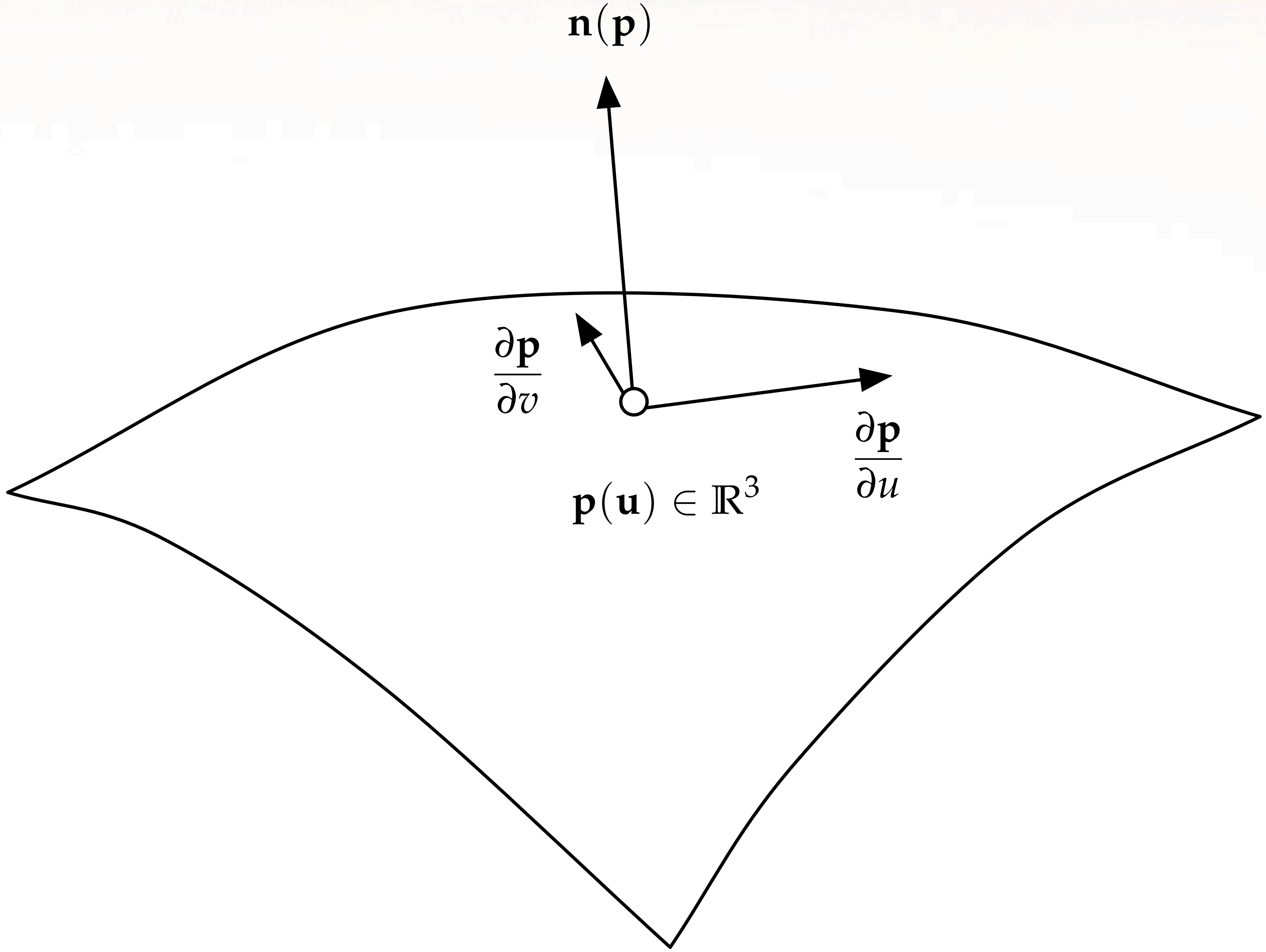
Some differential geometry



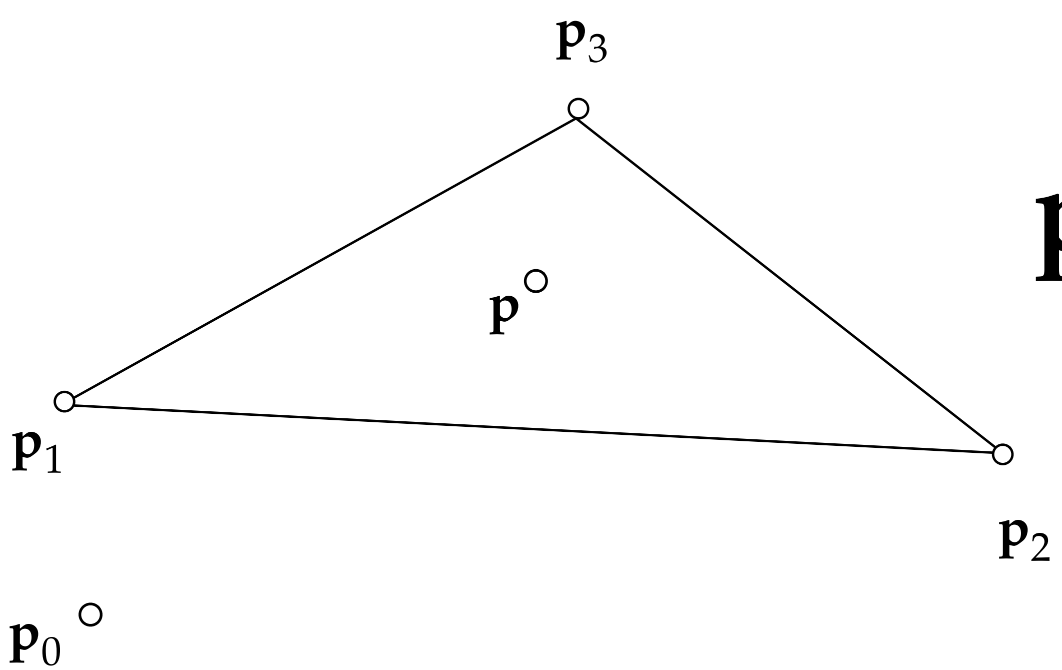
Surface normals used **for shading**



Surface normals obtained **from tangent space**



Tangent vectors inside a triangle



$$\mathbf{p}_i = \mathbf{p}_0 + u_i \frac{\partial \mathbf{p}}{\partial u} + v_i \frac{\partial \mathbf{p}}{\partial v}$$

Fully determined from **positions** and **parameters**

we are not interested in $\mathbf{p}_0$, thus

$$\mathbf{p}_2 - \mathbf{p}_1 = (u_2 - u_1)\frac{\partial \mathbf{p}}{\partial u} + (v_2 - v1)\frac{\partial \mathbf{p}}{\partial v}$$

$$\mathbf{p}_3 - \mathbf{p}_1 = (u_3 - u_1)\frac{\partial \mathbf{p}}{\partial u} + (v_3 - v1)\frac{\partial \mathbf{p}}{\partial v}$$

2x2 Matrix inversion

$$\mathbf{p}_2 - \mathbf{p}_1 = (u_2 - u_1)\frac{\partial \mathbf{p}}{\partial u} + (v_2 - v1)\frac{\partial \mathbf{p}}{\partial v}$$

$$\mathbf{p}_3 - \mathbf{p}_1 = (u_3 - u_1)\frac{\partial \mathbf{p}}{\partial u} + (v_3 - v1)\frac{\partial \mathbf{p}}{\partial v}$$

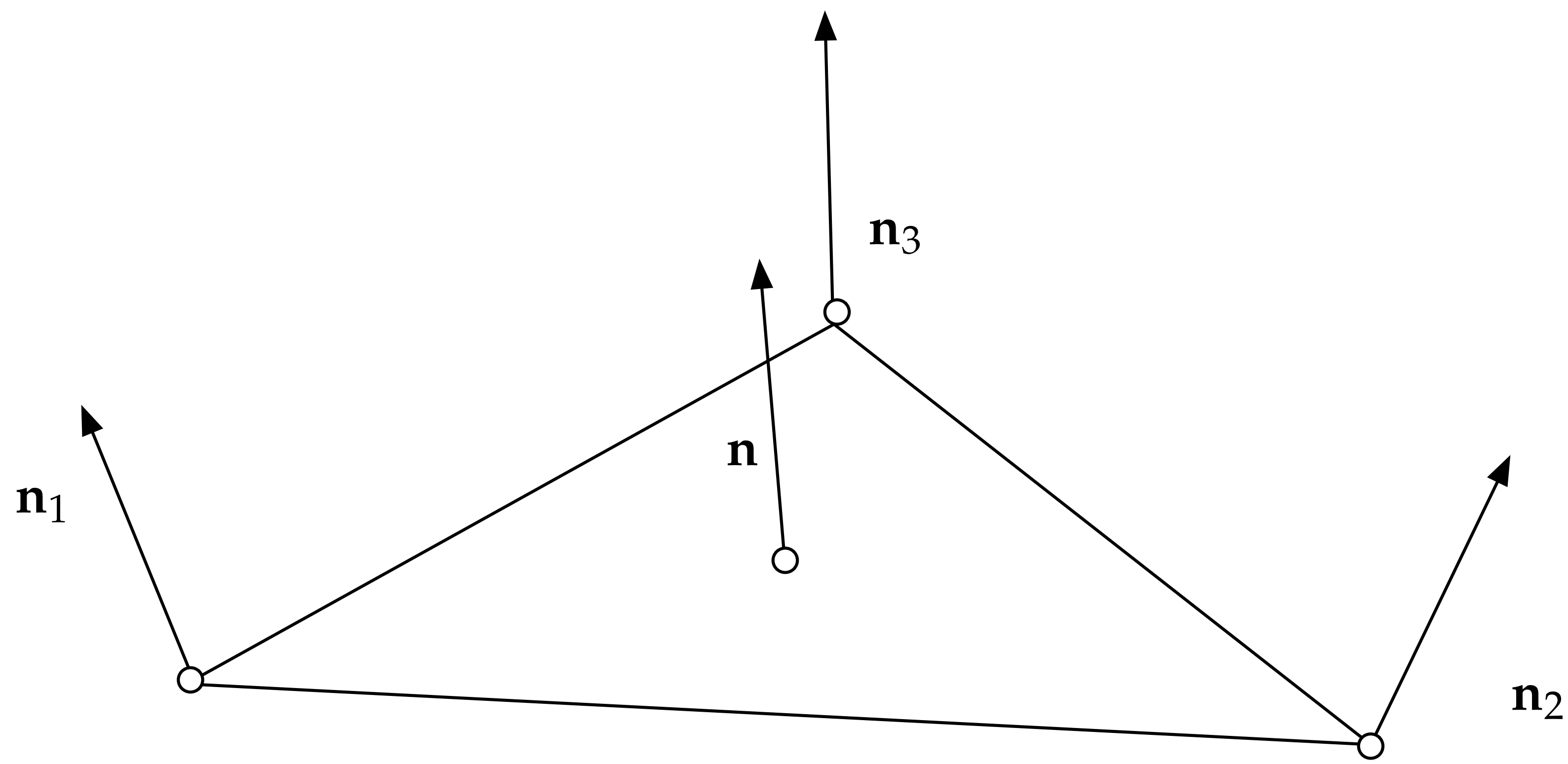
↓

$$\begin{bmatrix} \mathbf{p}_2 - \mathbf{p}_1 & \mathbf{p}_3 - \mathbf{p}_1 \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathbf{p}}{\partial u} & \frac{\partial \mathbf{p}}{\partial v} \end{bmatrix} \begin{bmatrix} (u_2 - u_1) & (u_3 - u_1) \\ (v_2 - v_1) & (v_3 - v_1) \end{bmatrix}$$

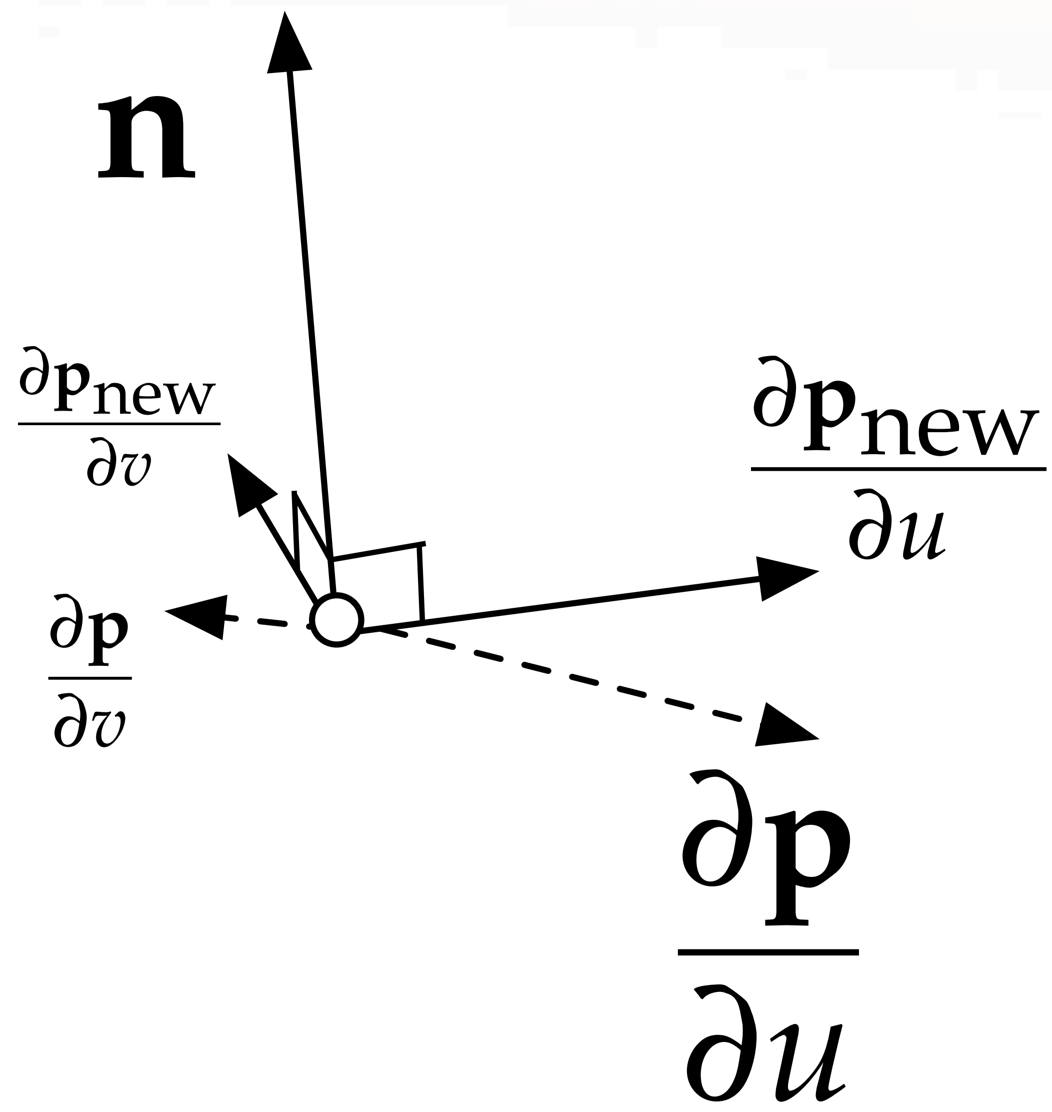
correct if mesh is planar

Normals interpolation

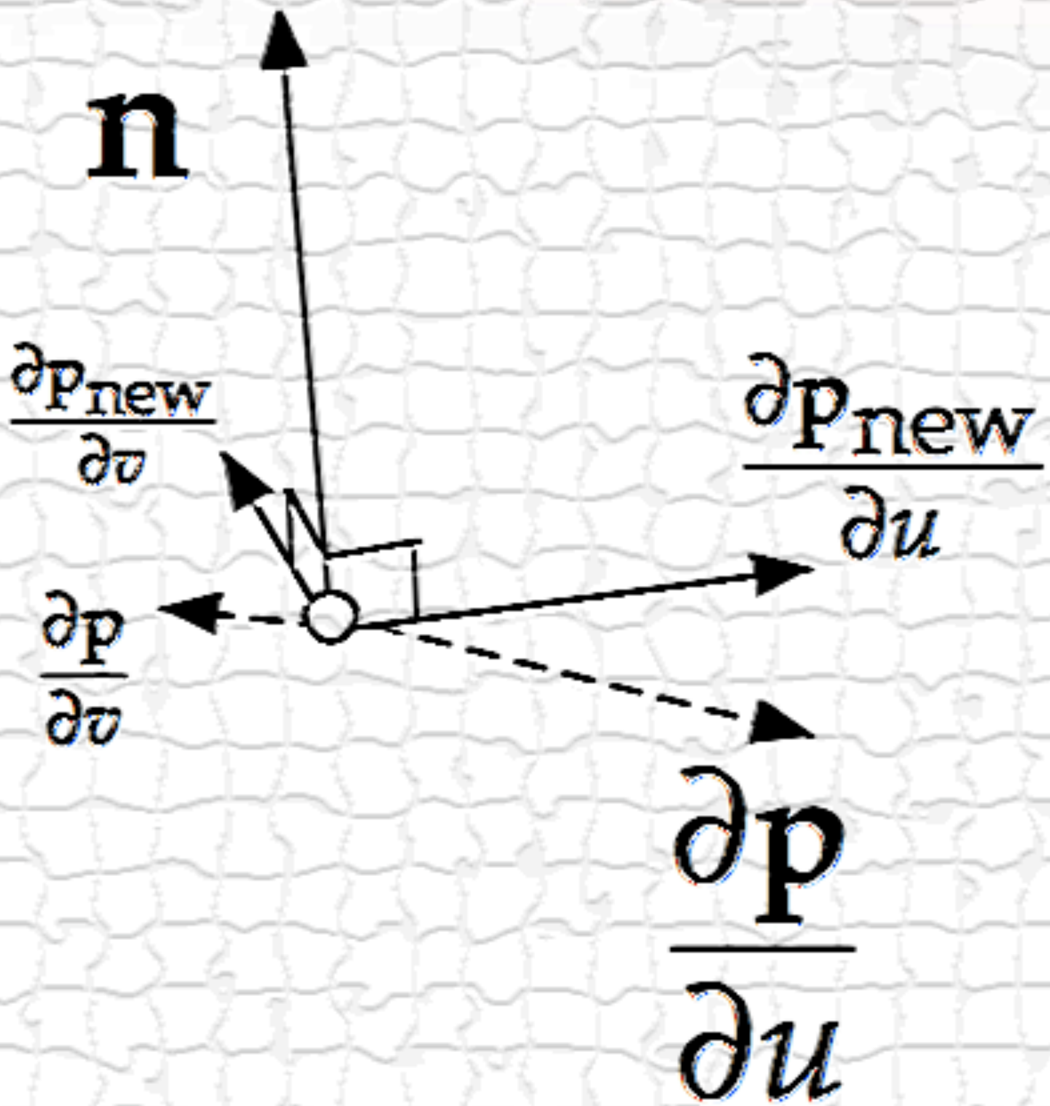
$$\mathbf{n} = \alpha_1 \mathbf{n}_1 + \alpha_2 \mathbf{n}_2 + \alpha_3 \mathbf{n}_3 \quad \text{from} \quad \mathbf{p} = \alpha_1 \mathbf{p}_1 + \alpha_2 \mathbf{p}_2 + \alpha_3 \mathbf{p}_3$$



Tangent vectors **orthogonal** to normal



Tangent vectors **orthogonal** to normal



We now have an **inexpensive way** to add **geometric details**

We now have an **inexpensive way** to add **geometric details**

Other bump mapping techniques exist

Jim Blinn today...

Further Readings

- “Simulation of Wrinkled Surfaces” [Blinn 1978]

- “Real-Time Rendering” [Akenine-Möller and Haines 2002] p.166 – 177

?

hao@inf.ethz.ch