Games with Texture Mapping

CS 4620 Lecture 13

Recall first definition…

Texture mapping: a technique of defining surface properties (especially shading parameters) in such a way that they vary as a function of position on the surface.

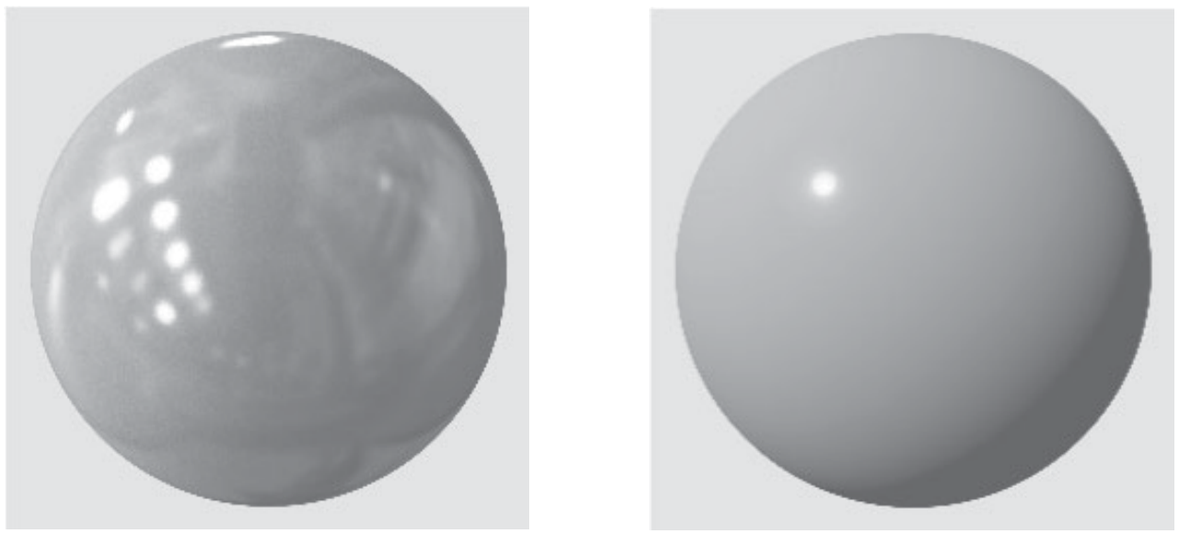
A refined definition

Texture mapping: a set of techniques for defining functions on surfaces, for a variety of uses.

- **Let's look at some examples of more general uses of texture maps.**

Reflection mapping

- **Early (earliest?) non-decal use of textures**
- **Appearance of shiny objects**
  - Phong highlights produce blurry highlights for glossy surfaces.
  - A polished (shiny) object reflects a sharp image of its environment.
- **The whole key to a shiny-looking material is providing something for it to reflect.**



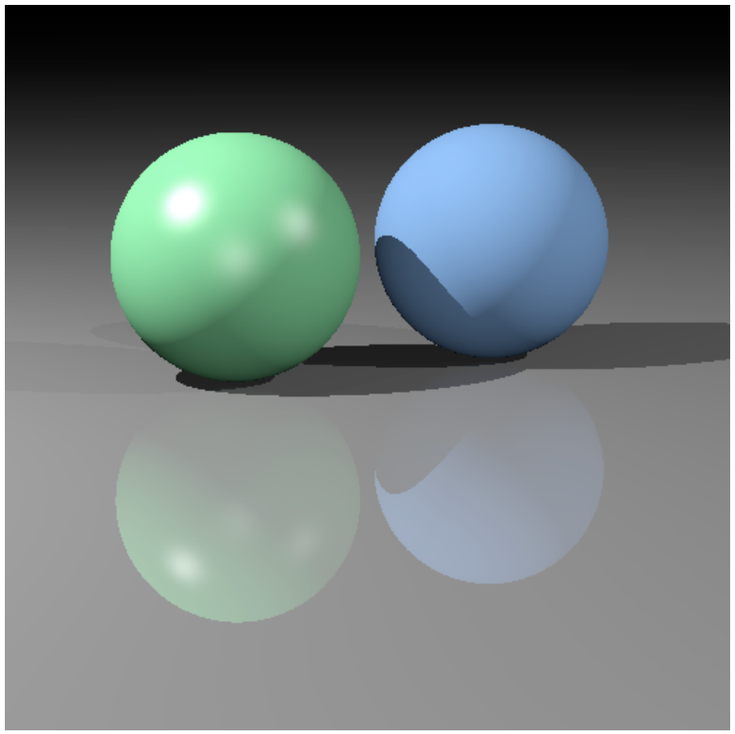
[Dror, Willsky, & Adelson 2004]

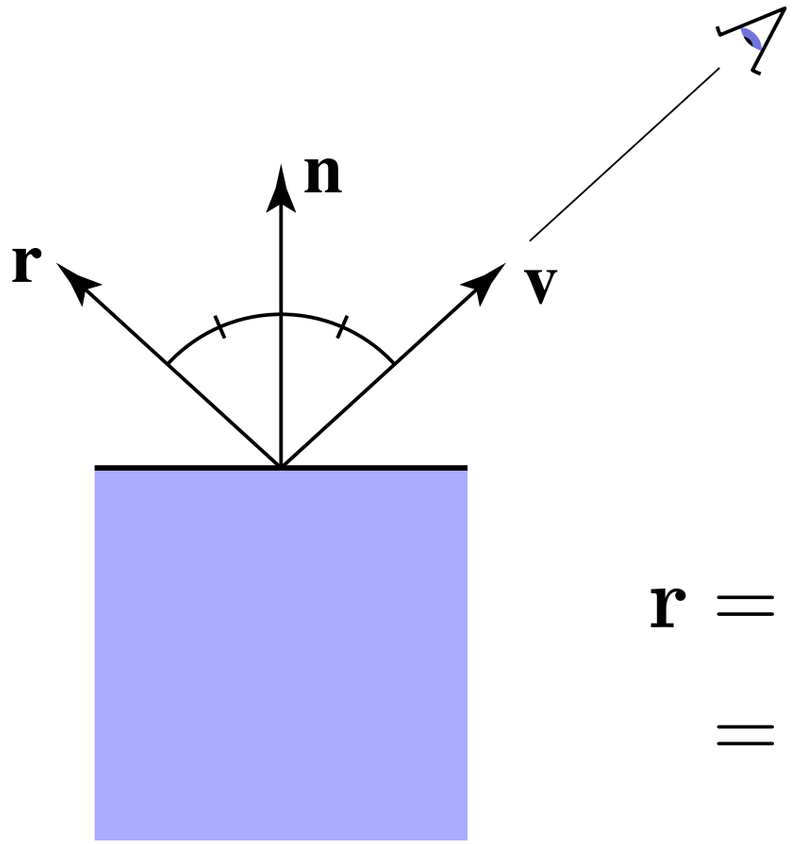
(a) (b)

Figure 2. (a). A shiny sphere rendered under photographically acquired real-world illumination. (b). The same sphere rendered under illumination by a point light source.

Reflections in ray tracing

- **Recall how we can make mirror reflections in ray tracing**





$$\begin{aligned}\mathbf{r} &= \mathbf{v} + 2((\mathbf{n}\cdot\mathbf{v})\mathbf{n} - \mathbf{v}) \\ &= 2(\mathbf{n}\cdot\mathbf{v})\mathbf{n} - \mathbf{v}\end{aligned}$$

Reflection mapping

- **If scene is infinitely far away, the color seen by the reflection ray depends only on the direction of the ray**
  - a two-dimensional function
  - represent it with a texture!
- **Environment map: texture that maps directions to colors**
  - one option: axes are (theta, phi)
  - better option: cube map

straight up

straight up

straight up

due East

due South

due West

due North

due East

straight down

straight down

straight down

A spherical panorama, aka. enironment map

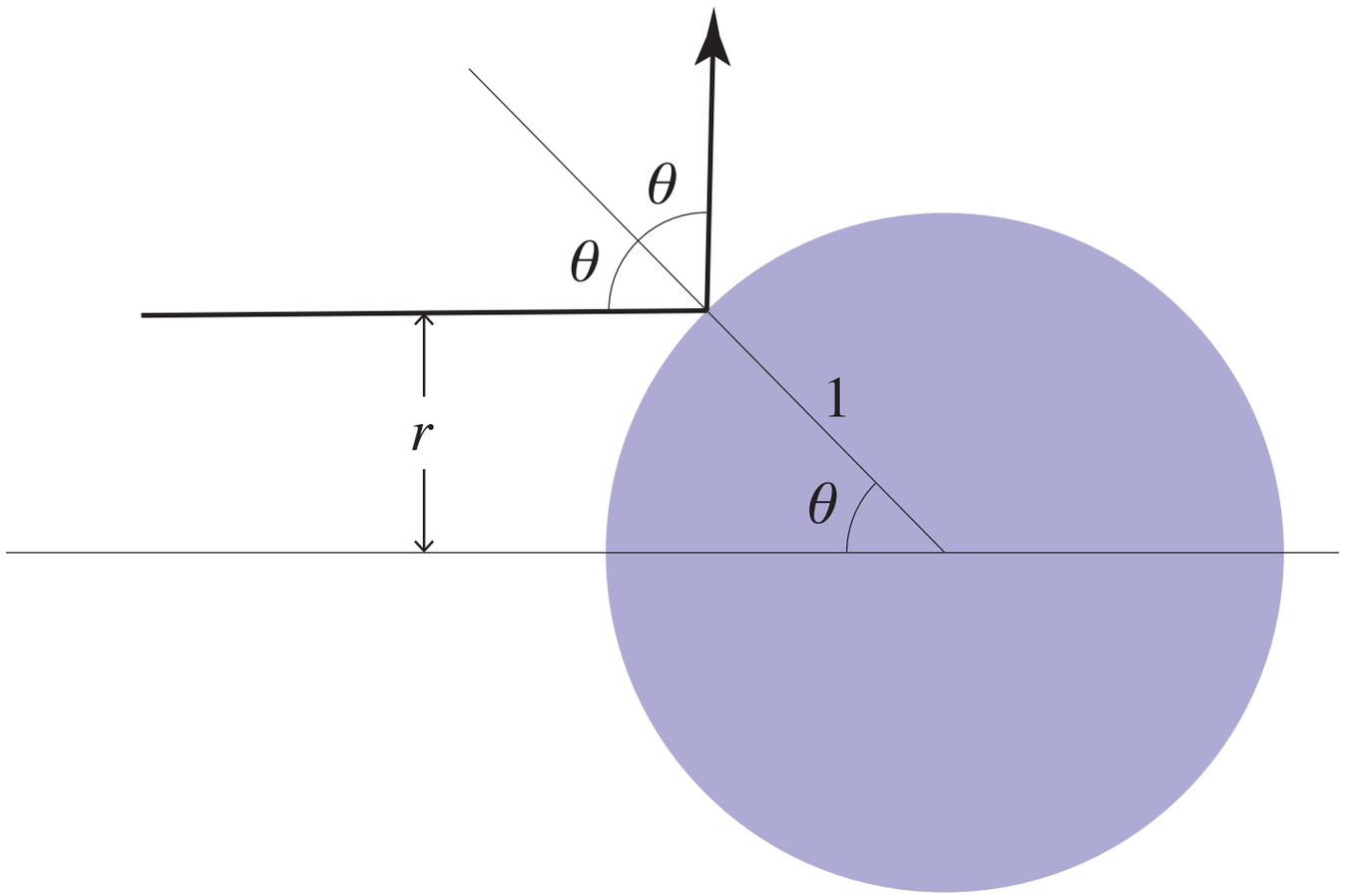
openfootage.net

Environment map

- **A function from the sphere to colors, stored as a texture.**

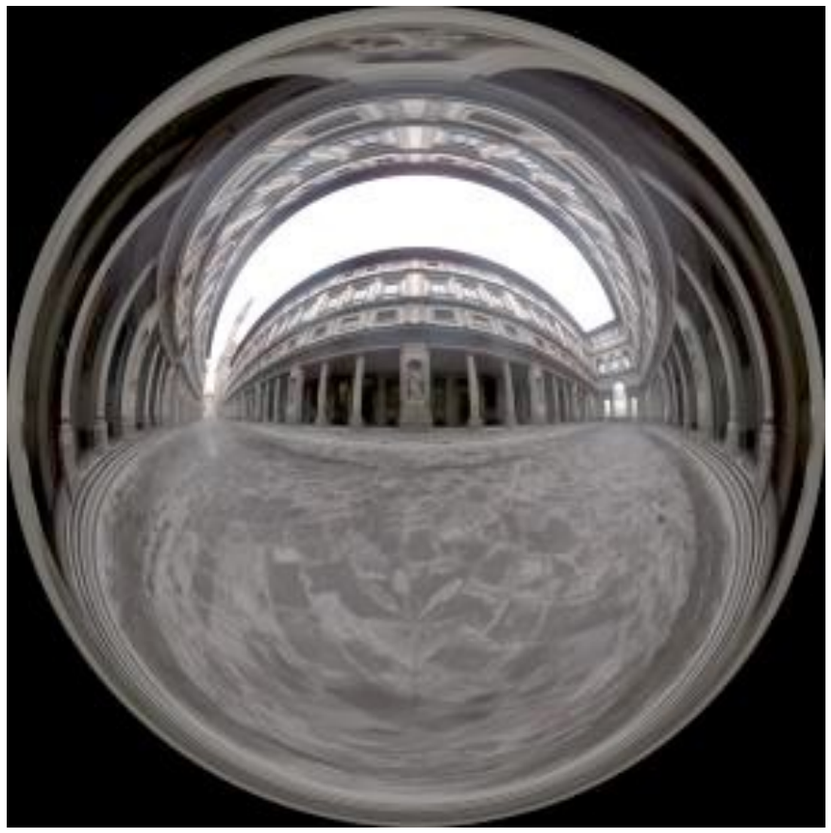
[Blinn & Newell 1976]

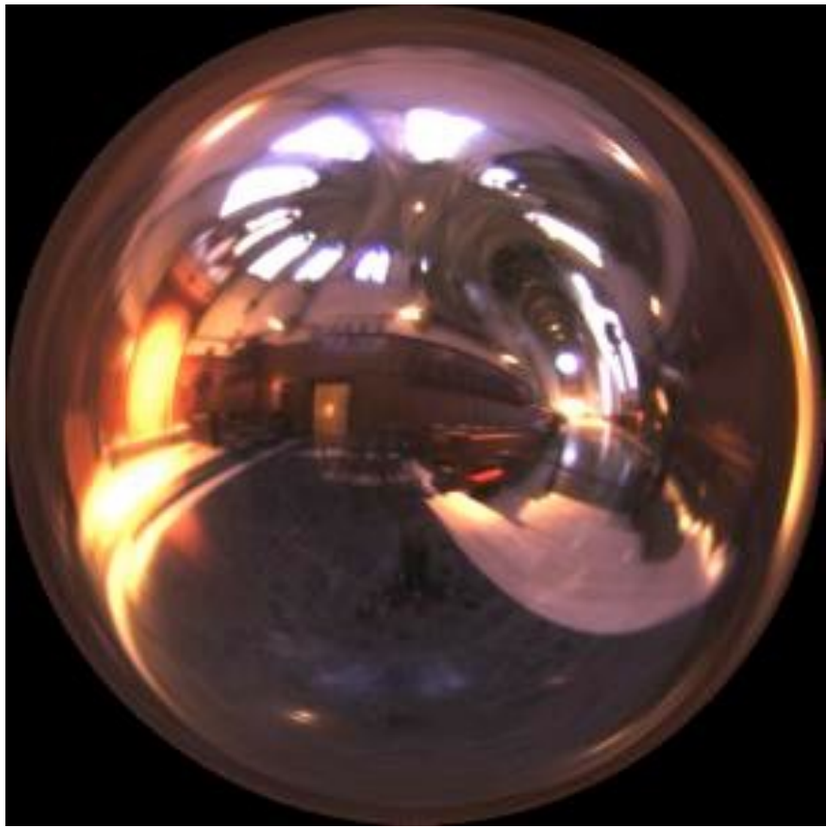
Spherical environment map



Hand with Reflecting Sphere. M. C. Escher, 1935. lithograph

Environment Maps

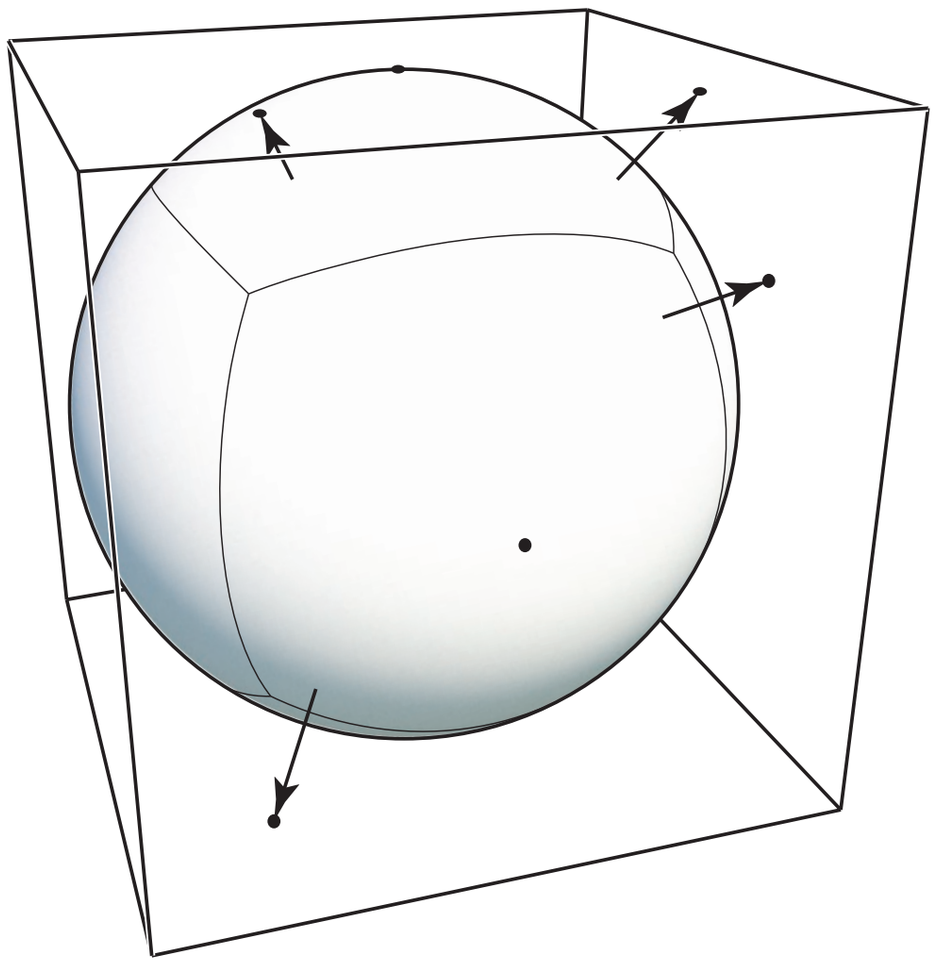


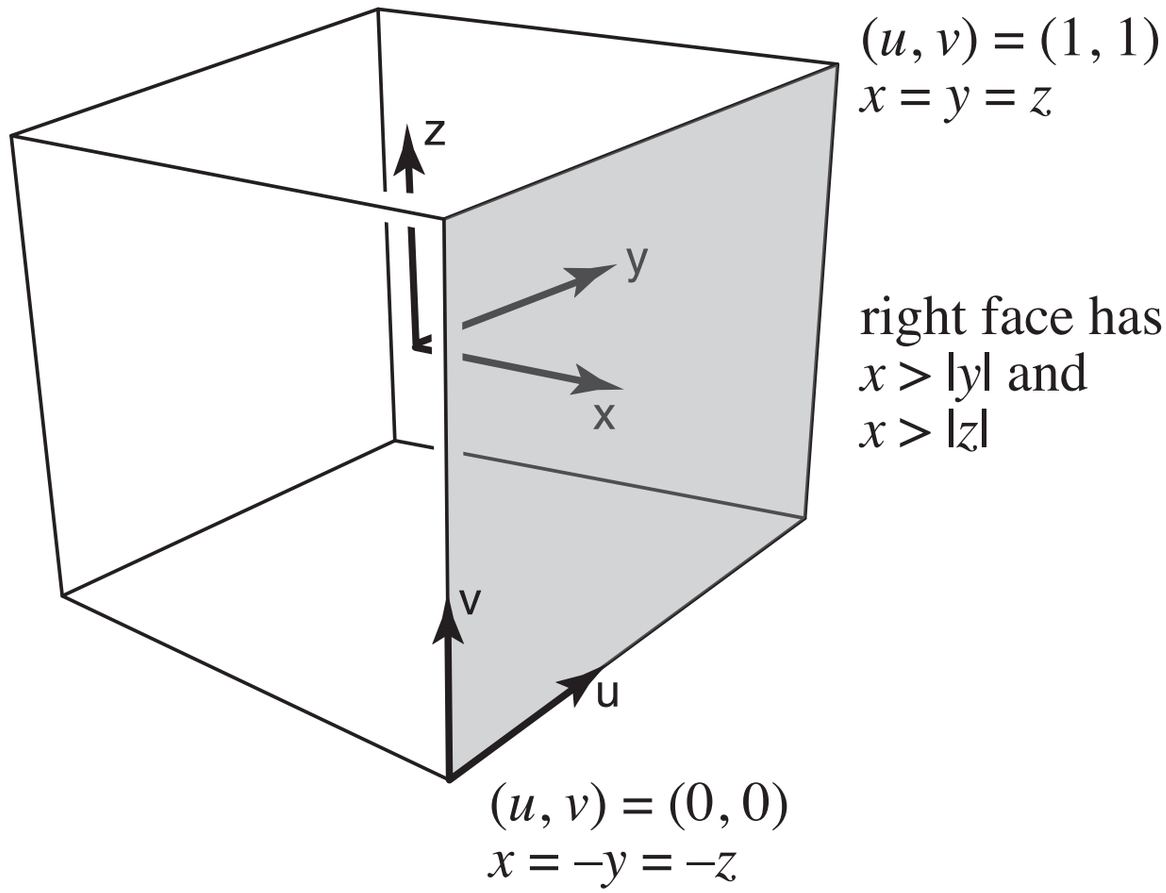


[Paul Debevec]

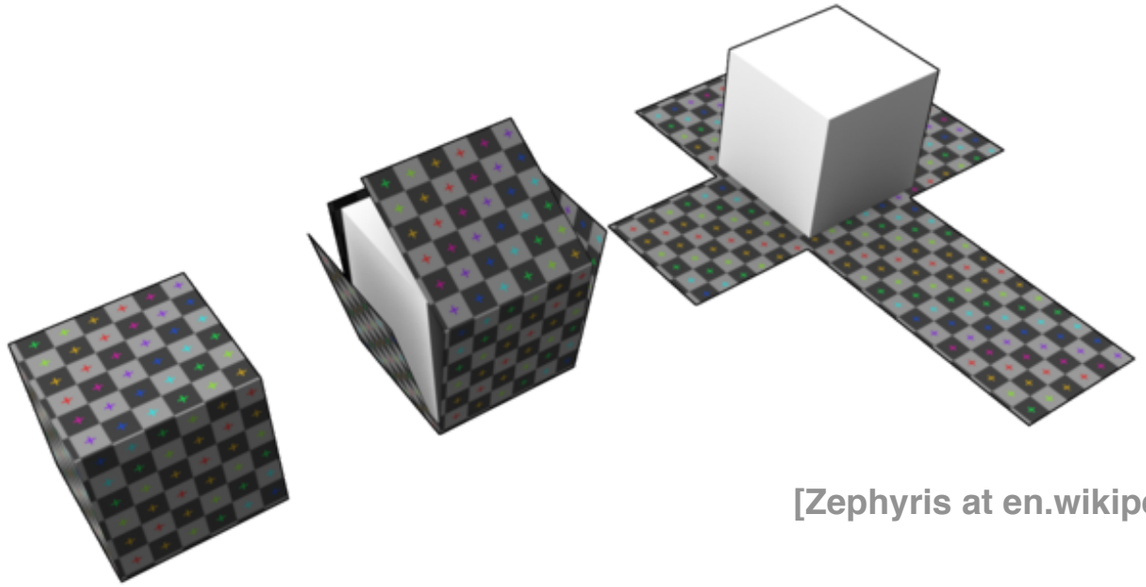
[CS467 slides]

Cube map





a direction vector maps to the point on the cube that is along that direction.
The cube is textured with 6 square texture maps.



[Zephyris at en.wikipedi

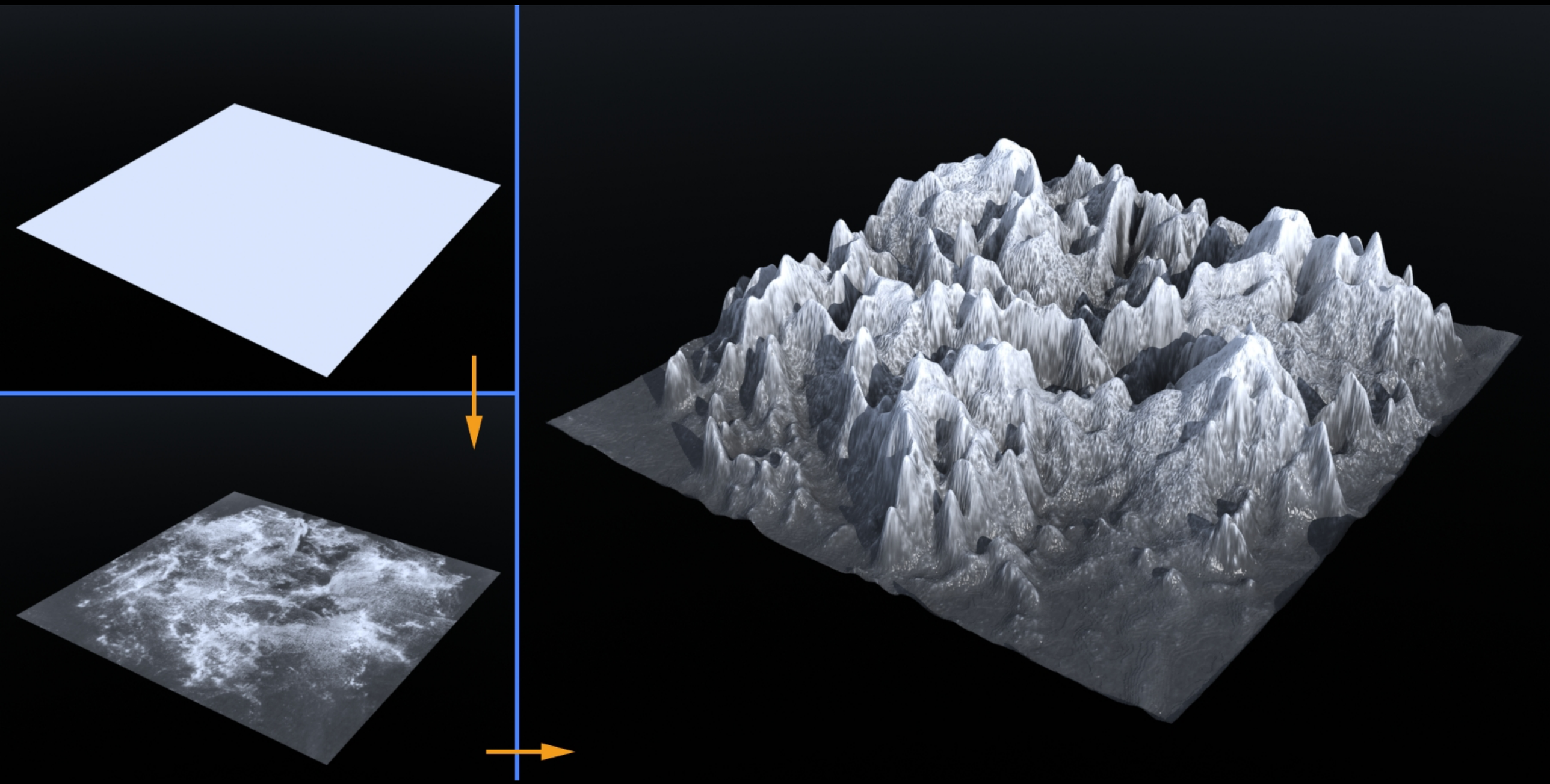
[Emil Persson]

Reflection mapping in GLSL

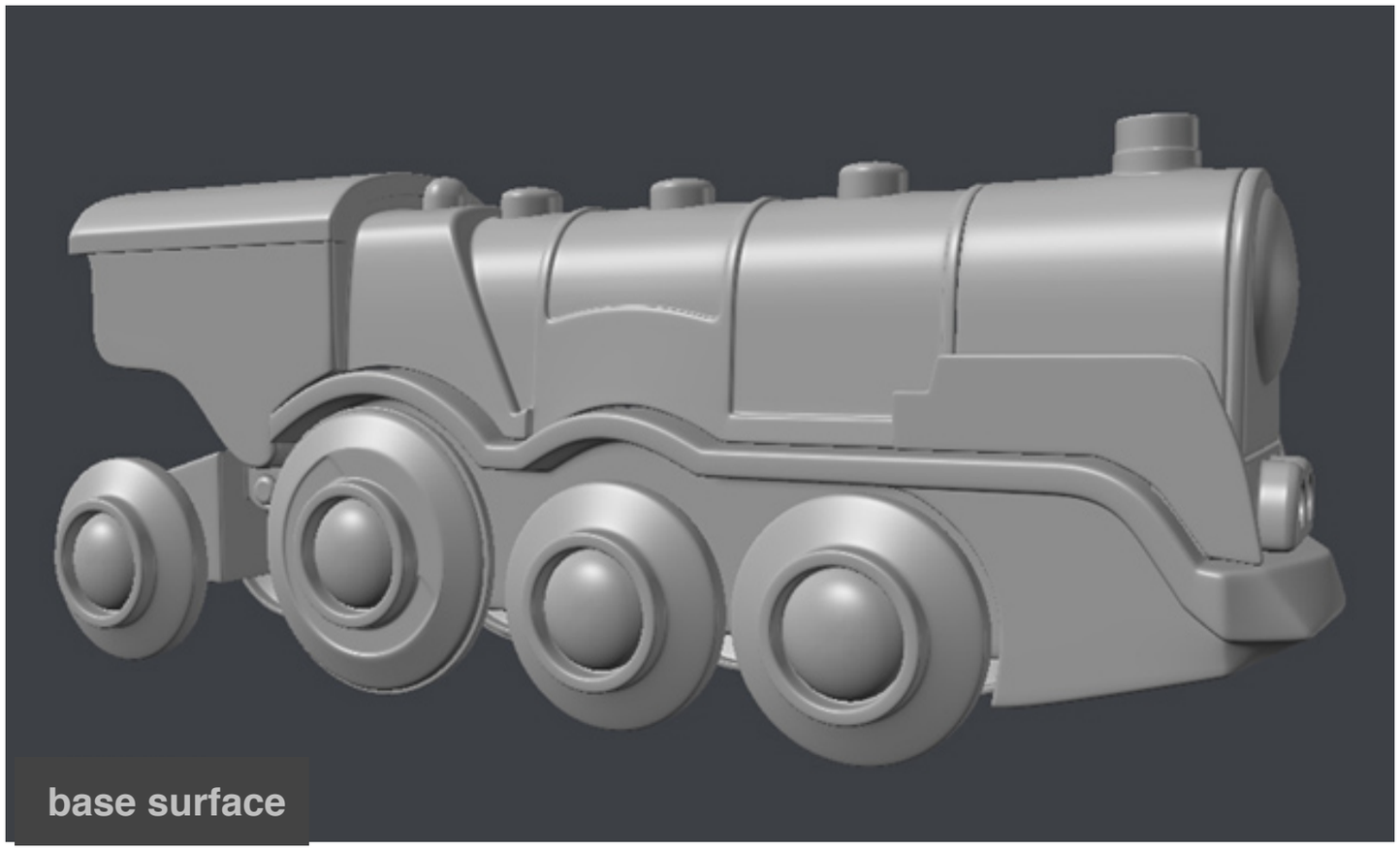
- **A fragment operation**
  - requires surface normal and a way to get the view direction
- **GLSL handles cubemaps by itself**
  - you just give it the reflection vector and it figures out where to sample and on which face
  - sample using `textureCube()`
- **Don't overlook built-in functions**
  - e.g. `reflect()`

Geometry

Displacement mapping

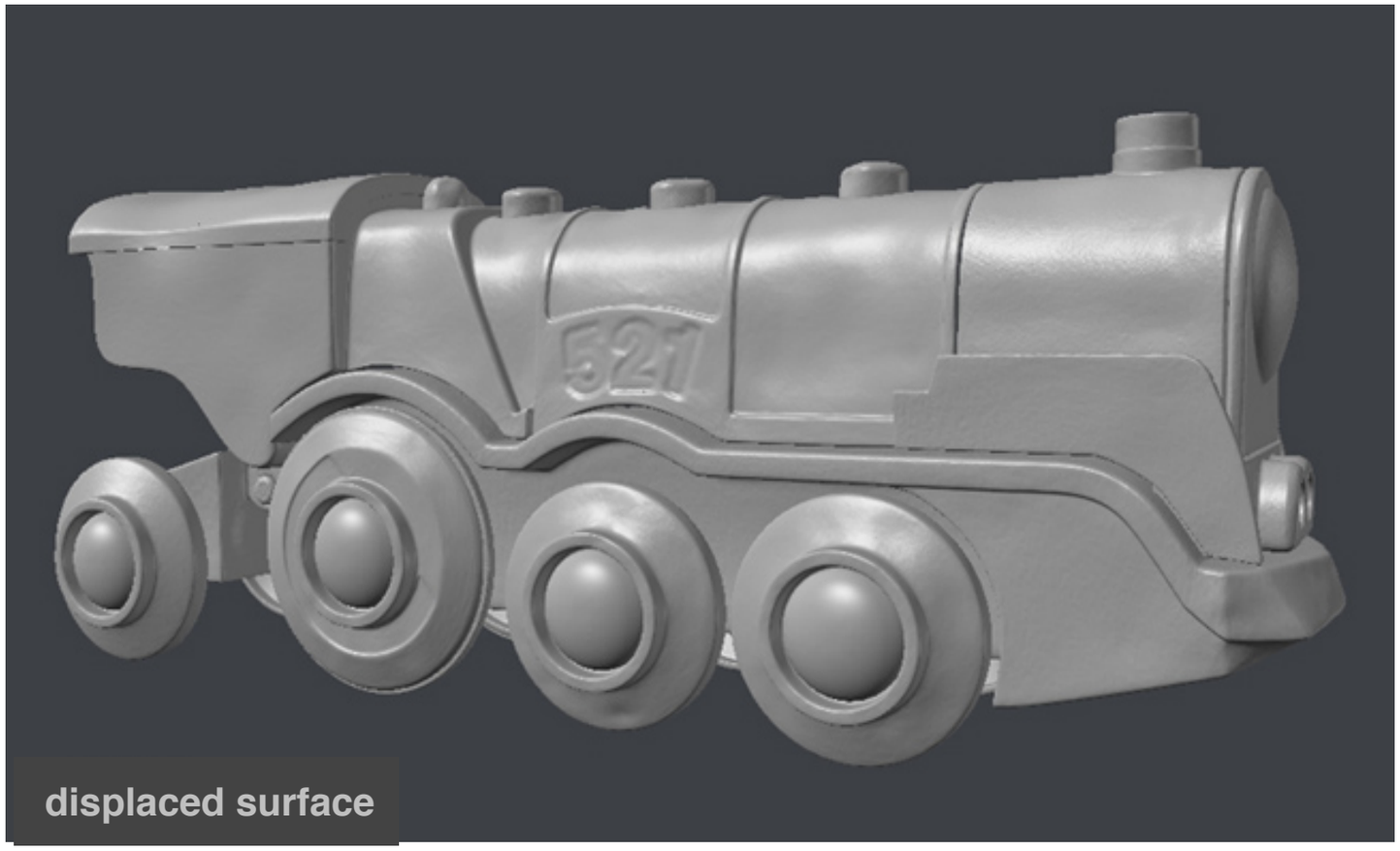


[wikiwand]



base surface

hand-painted displacement map (detail)



displaced surface

Paweł Filip
tolas.wordpress.com

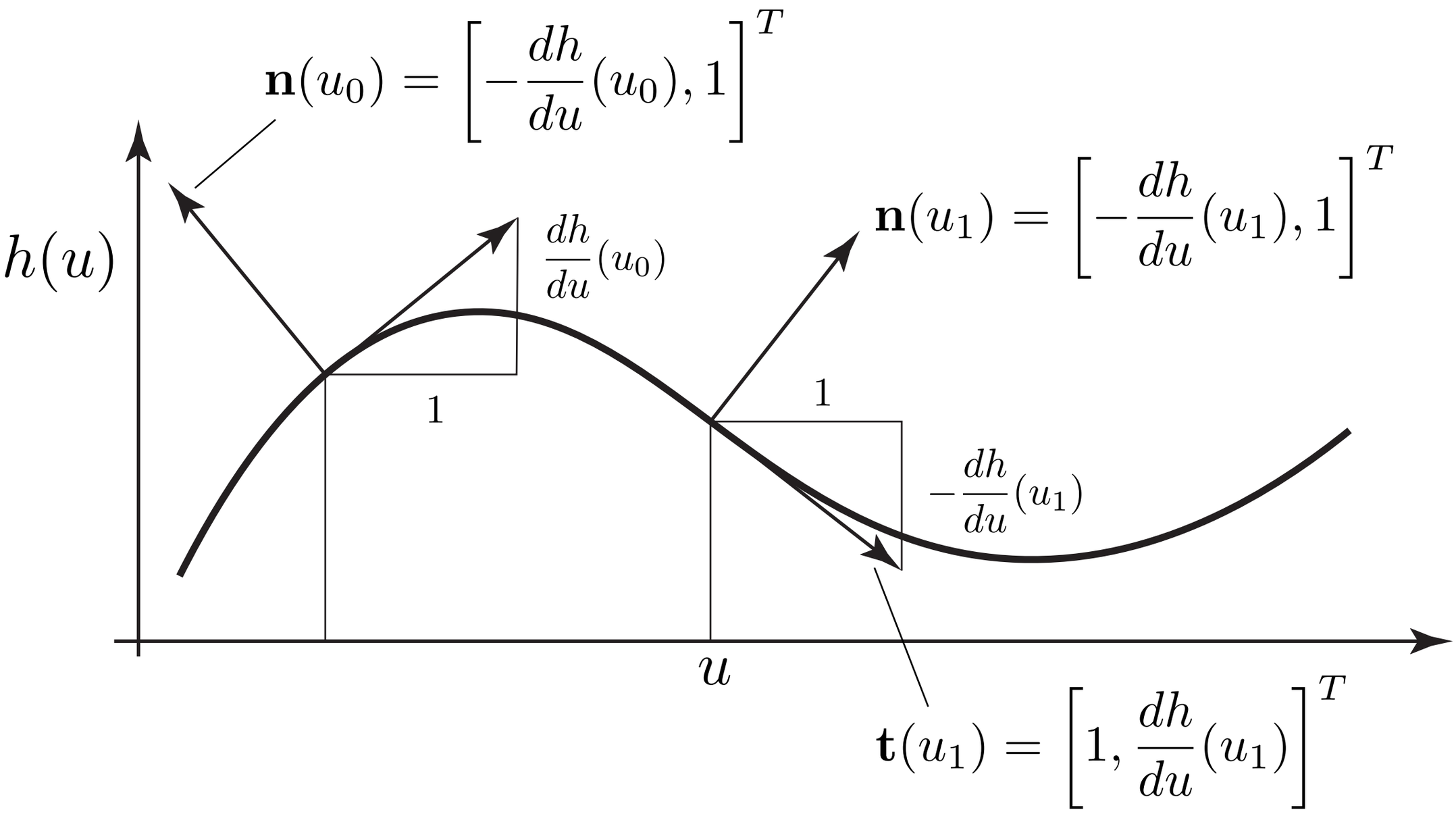

521
fryrender
physically-based render engine
©2007 Paweł Filip

Displacement mapping

- **A powerful tool for modeling detail**
  - used heavily in film production
- **Geometric prerequisites**
  - texture map representing height field
  - smooth normals
  - dense triangulation
- **In GLSL**
  - a vertex operation (because it moves geometry)
  - displace vertices along normal vectors
  - displacement distance proportional to texture map value

Normals in displacement mapping

- **Displacing the surface changes the surface normal**



Normals in displacement mapping

- **In the 3D case the correct normal is**

$$\mathbf{n}(u,v) = \left[-\frac{\partial h}{\partial u}, -\frac{\partial h}{\partial v}, 1\right]^T$$

  - normalized, of course!
  - this is measured in the coordinates of the height field
- **This needs to be computed and transformed to object space**
  - the necessary transformation is the *tangent frame* of the surface
  - a basis defined by the three vectors:
    - tangent in *u* direction
    - tangent in *v* direction
    - surface normal

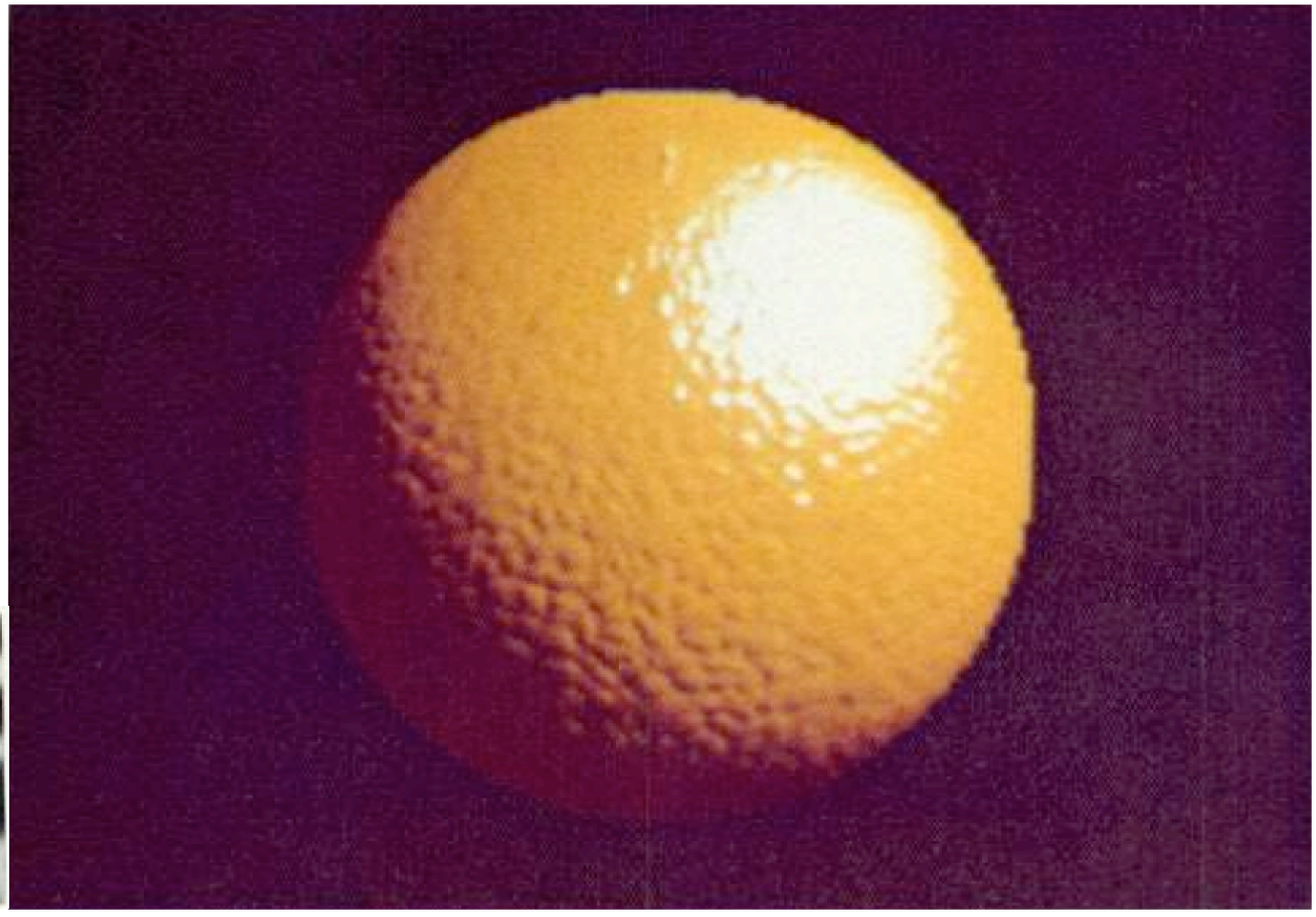
Geometry for displacement

- **geometric data**
  - normal
  - *u* tangent
  - *v* tangent
  - height field texture

in A4 we don't worry about normals in displacement mapping
(since normal mapping is its own part)

- **geometric logic**
  - look up displacement value from texture
  - compute normal to displaced surface
    - compute derivatives of height by finite differences
    - or precompute these normals (see normal mapping in a moment)
  - transform normal to (tangent-u, tangent-v, normal) space

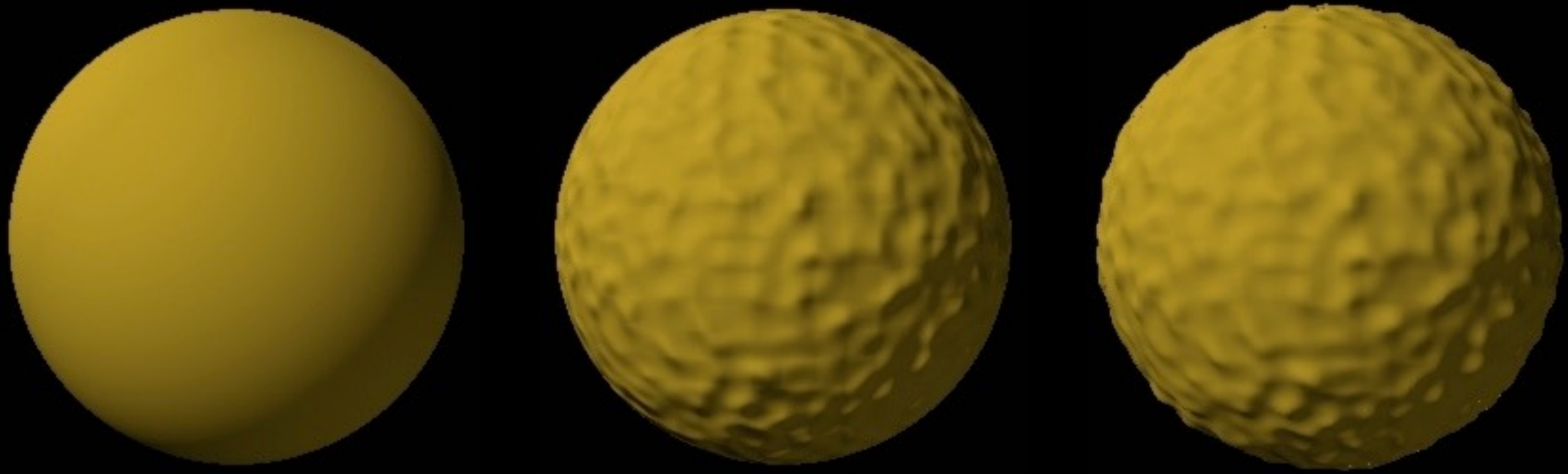
Bump mapping



[Blinn 1978]

Bump mapping

- **Displacement mapping is expensive**
  - requires densely tesselated geometry
  - many triangles to rasterize
- **For small displacements, the most important effect is on the normal**
  - so just do that part; don't displace the surface
- **Bump mapping is then a fragment operation**
  - doesn't require dense tesselation
  - doesn't actually displace the surface
  - gives shading that looks just like displaced surface



Geometry

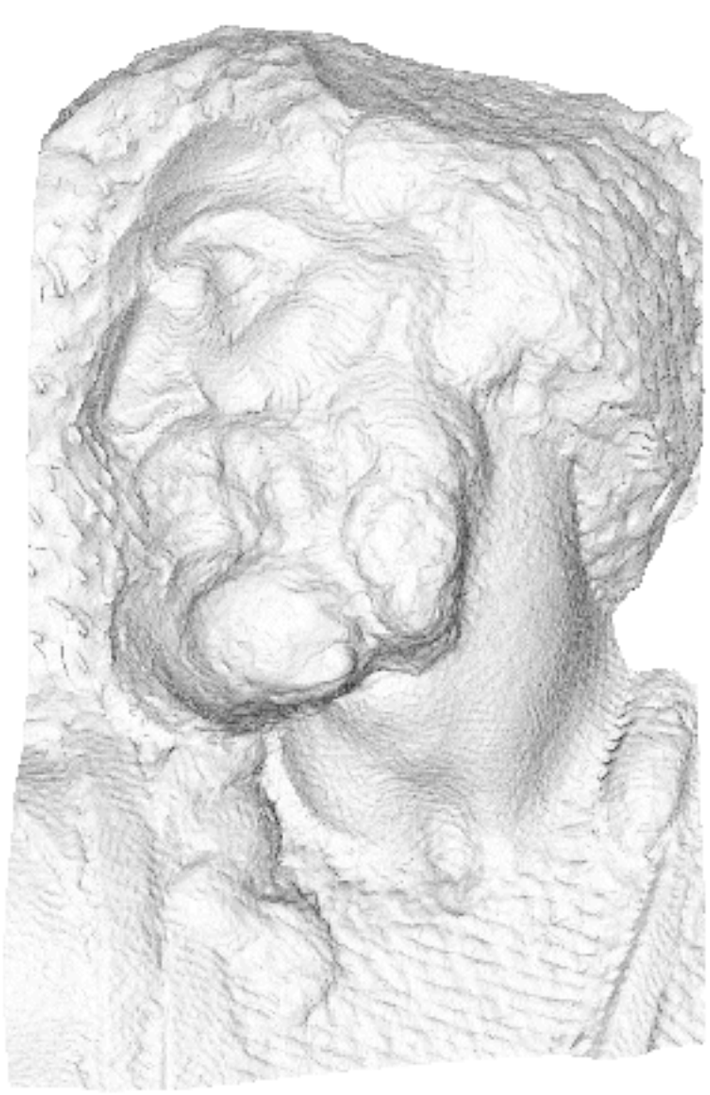
Bump mapping

Displacement mapping

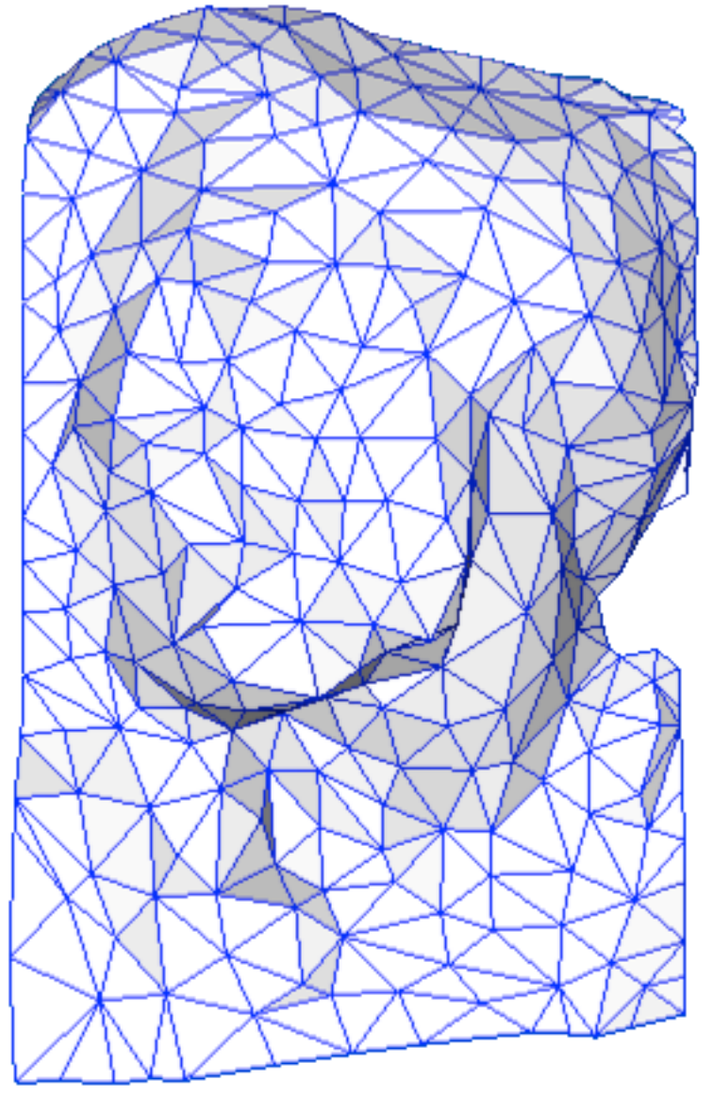
Bump mapping

- **Geometric prerequisites**
  - texture map representing height field
  - smooth normals
  - tangent vectors
  - no dense triangulation needed
  - **Geometric logic**
    - compute normal to displaced surface
      - compute derivatives of height by finite differences
    - transform normal to (tangent-u, tangent-v, normal) space

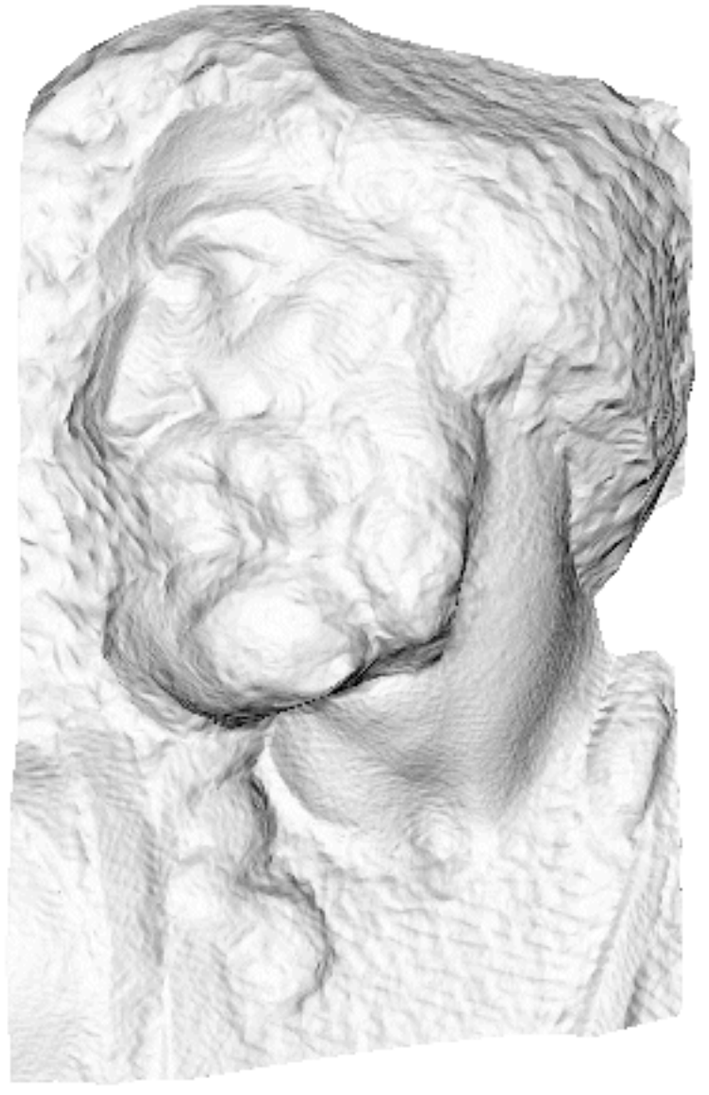
Normal mapping



original mesh
4M triangles



simplified mesh
500 triangles



simplified mesh
and normal mapping
500 triangles

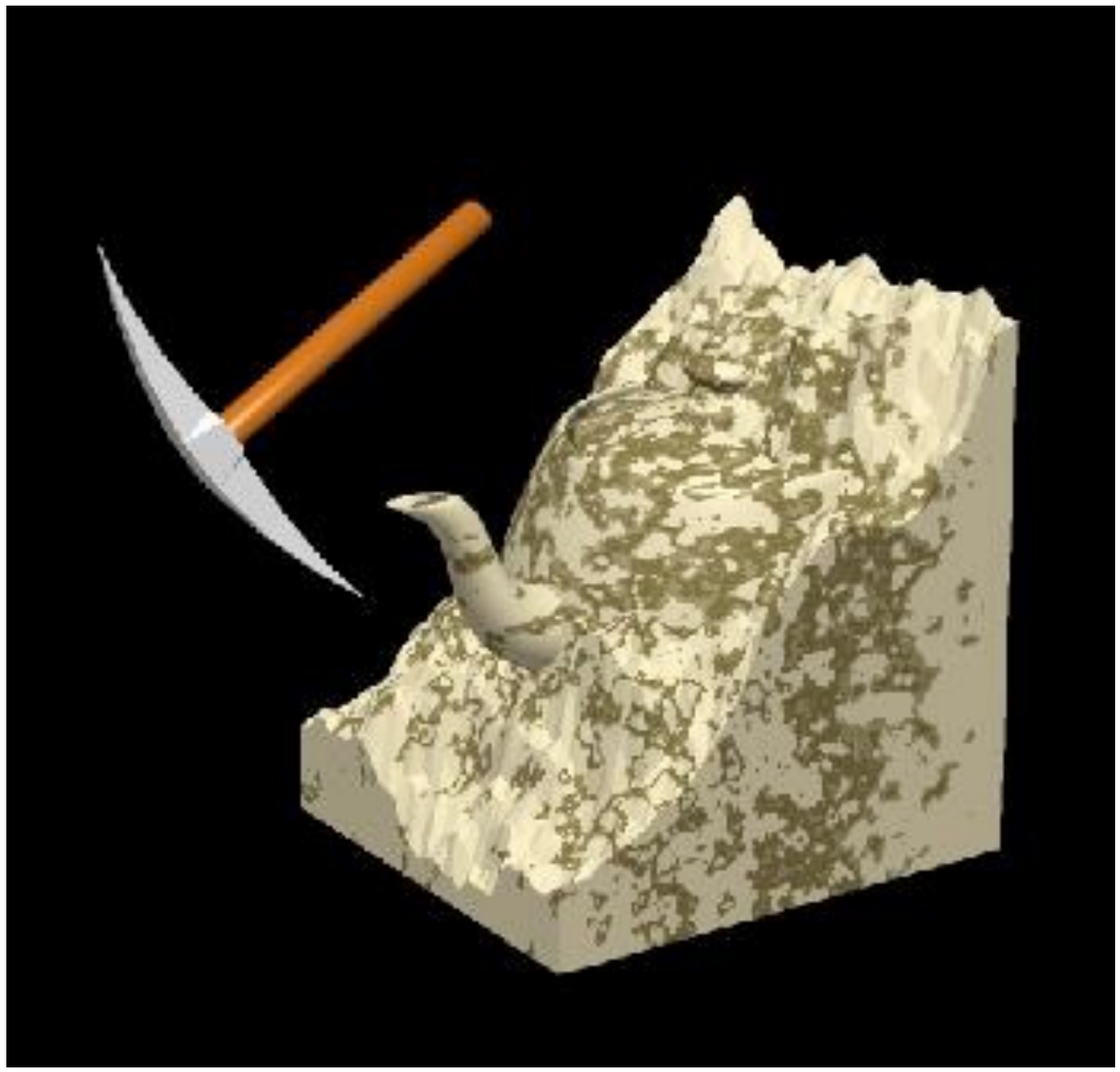
[Paolo Cignoni]

Normal mapping

- **Geometric prerequisites**
  - Texture map (3 channels) representing normal field
    - single lookups into normal map required
  - Smooth normals
  - Tangent vectors
    - if you want to store normals in tangent space (and you do)
  - No dense triangulation needed
  - No finite differencing needed
- **Geometric logic**
  - look up normal from map
  - transform into (tangent-u, tangent-v, normal) space

3D textures

- **Texture is a function of (*u*, *v*, *w*)**
  - can just evaluate texture at 3D surface point
  - good for solid materials
  - often defined procedurally
  - see book for more!



[Wolfe / SG97 Slide set]

