CS100433
Texture Mapping

Junqiao Zhao 赵君峤

Department of Computer Science and Technology

College of Electronics and Information Engineering

Tongji University

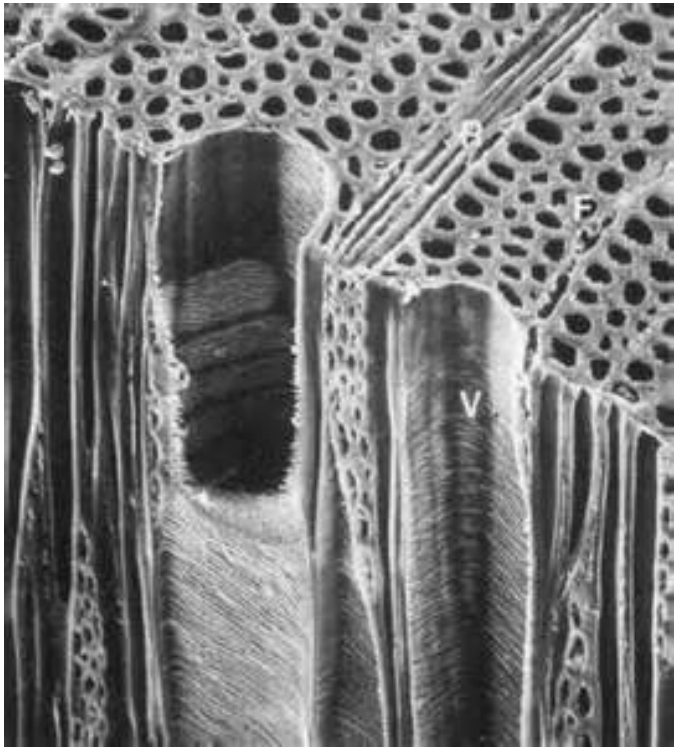
Appearance property

- Objects have geometry as well as appearance properties

[2013 Steve Marschner]

Appearance property

- In most scenarios, those properties are not modeled geometrically
- Why?



http://civcal.media.hku.hk/materials/wood/structure/microstructure/images/tests07.htm

Appearance properties

- In many case such properties are demanded

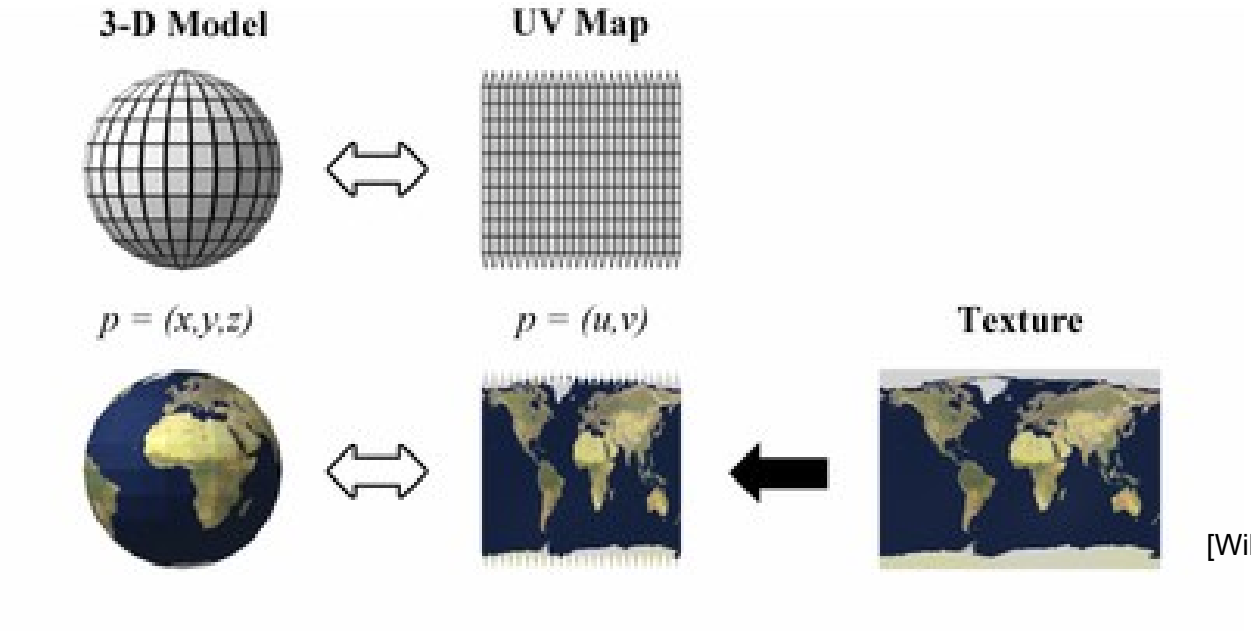
[Tom Thorne, Edinburgh]

What is 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 - Steve Marschner

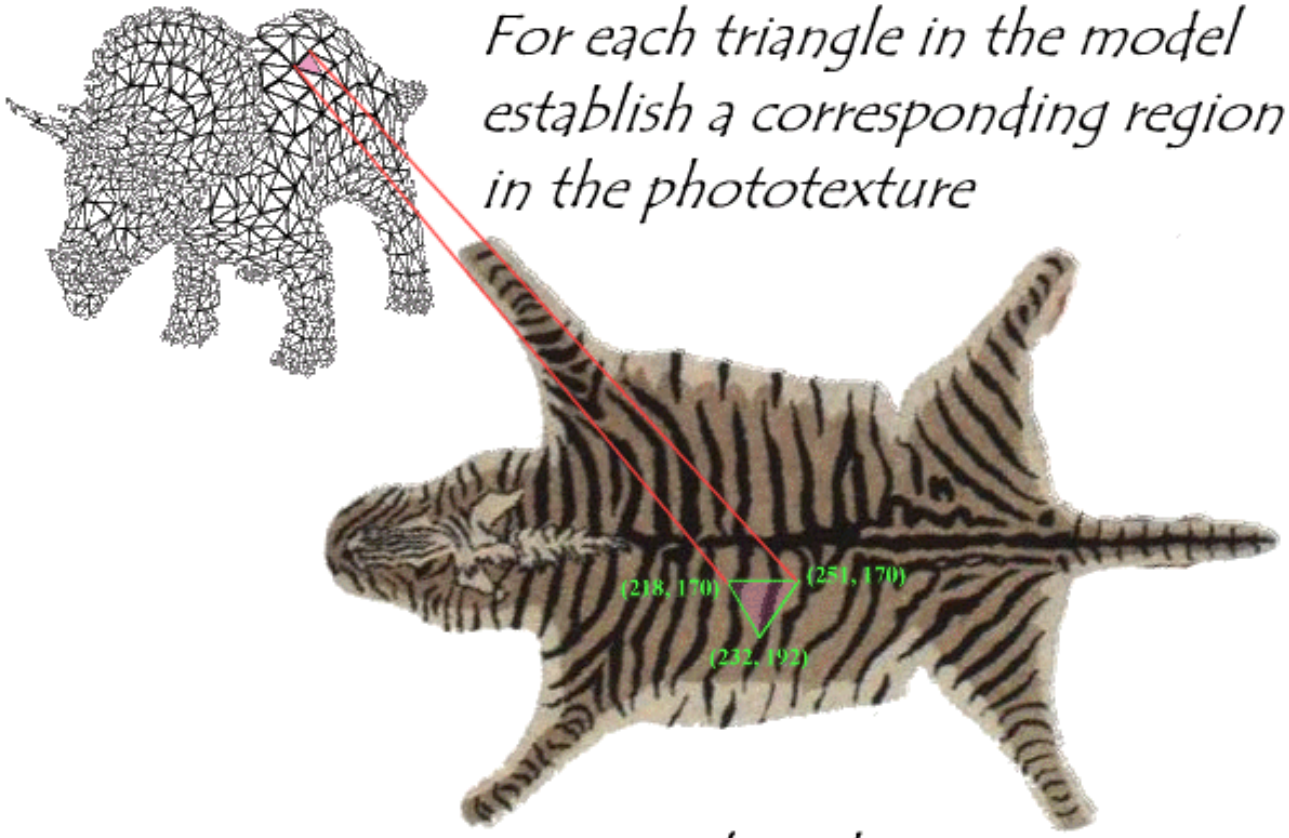
What is texture mapping?

- An “image” is pasted onto the 3D model
- The “image” is called a **texture map**, the pixels of the image are referred as **texels** and referenced using **UV** coordinates



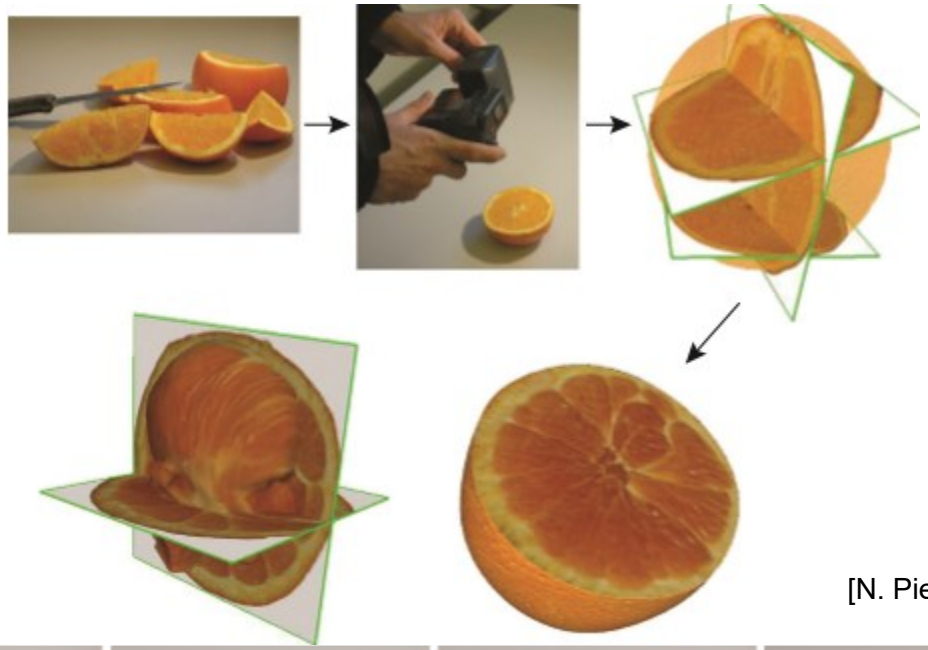
[Wikipedia]

What is texture mapping?



[Tom Thorne, Edinburgh]

What is texture mapping?

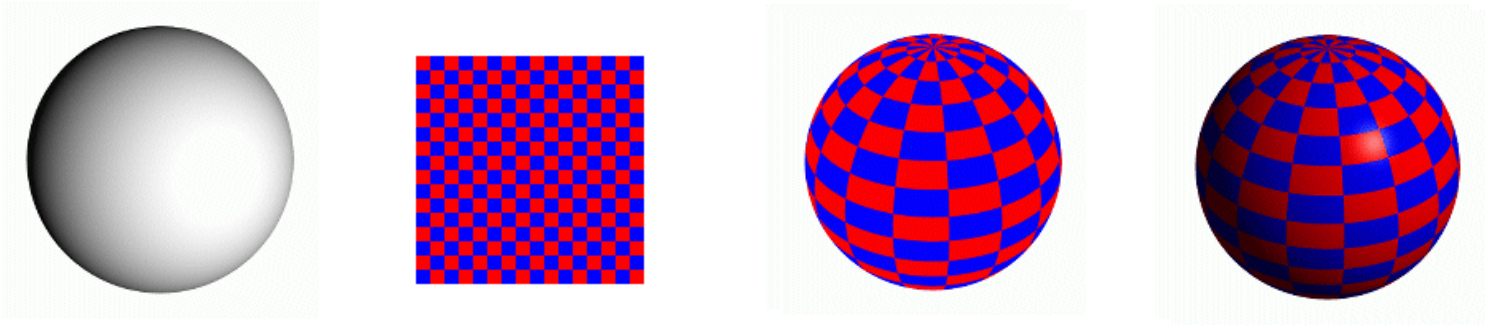


[N. Pietroni et al., 2007]

[B. Cutler et al., 2002]

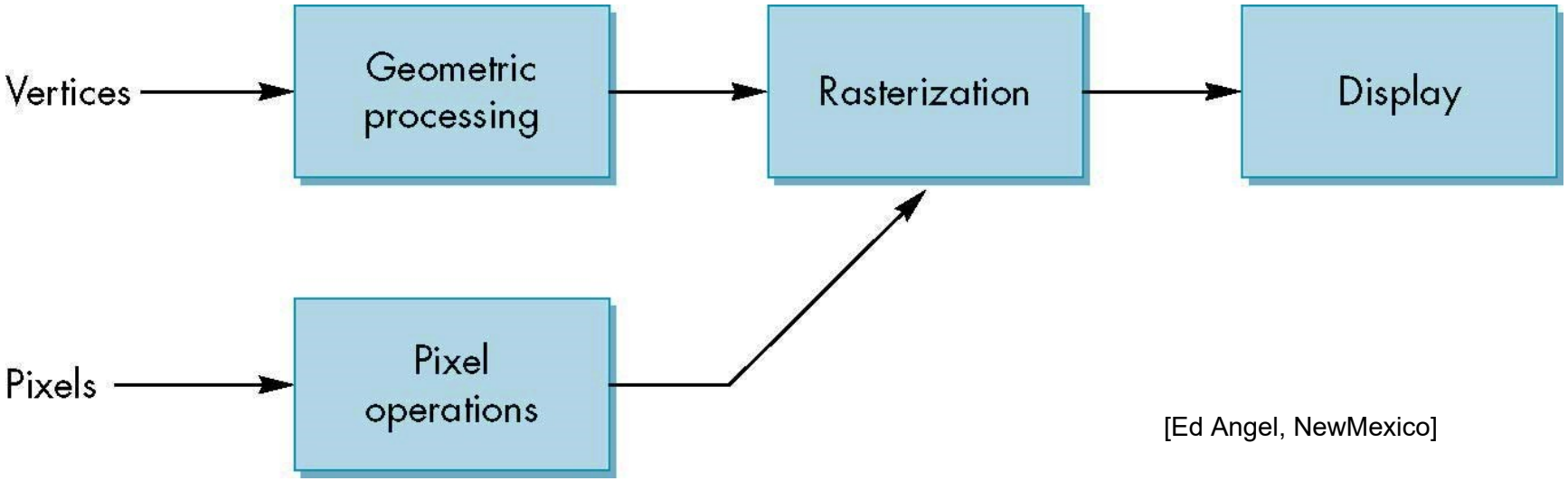
Texture vs Material

- Material is the intrinsic properties of a surface defining how the lighting interacts with the surface
  - per vertex (or per polygon)
- Texture is an “image-based” data describing appearance property
  - per texel/pixel
- Blended together



Where is texture mapping?

- Mapping techniques are implemented at the end of the rendering pipeline
  - Very efficient because polygons already passed the clipper



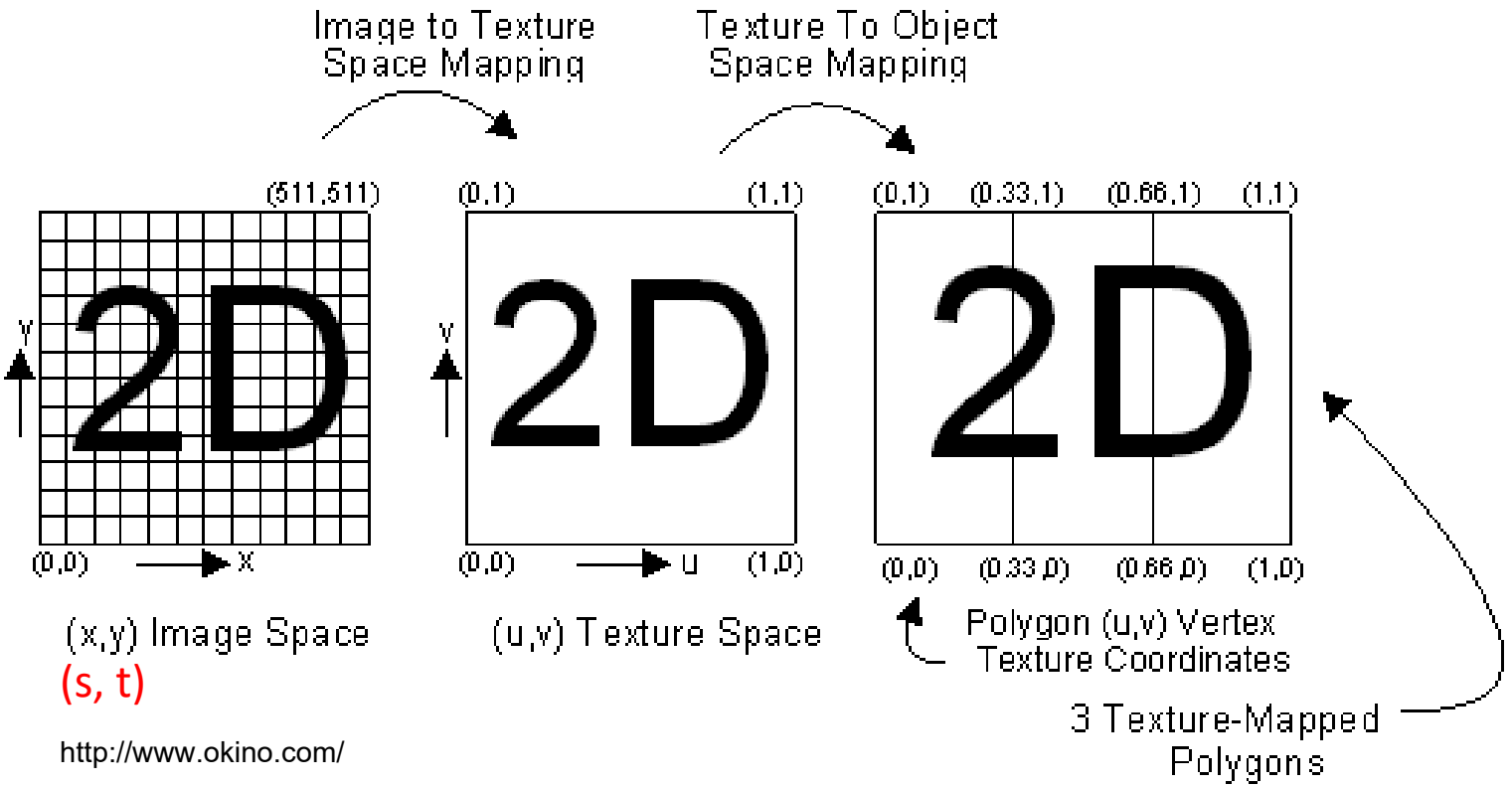
[Ed Angel, NewMexico]

1D, 2D and 3D texture

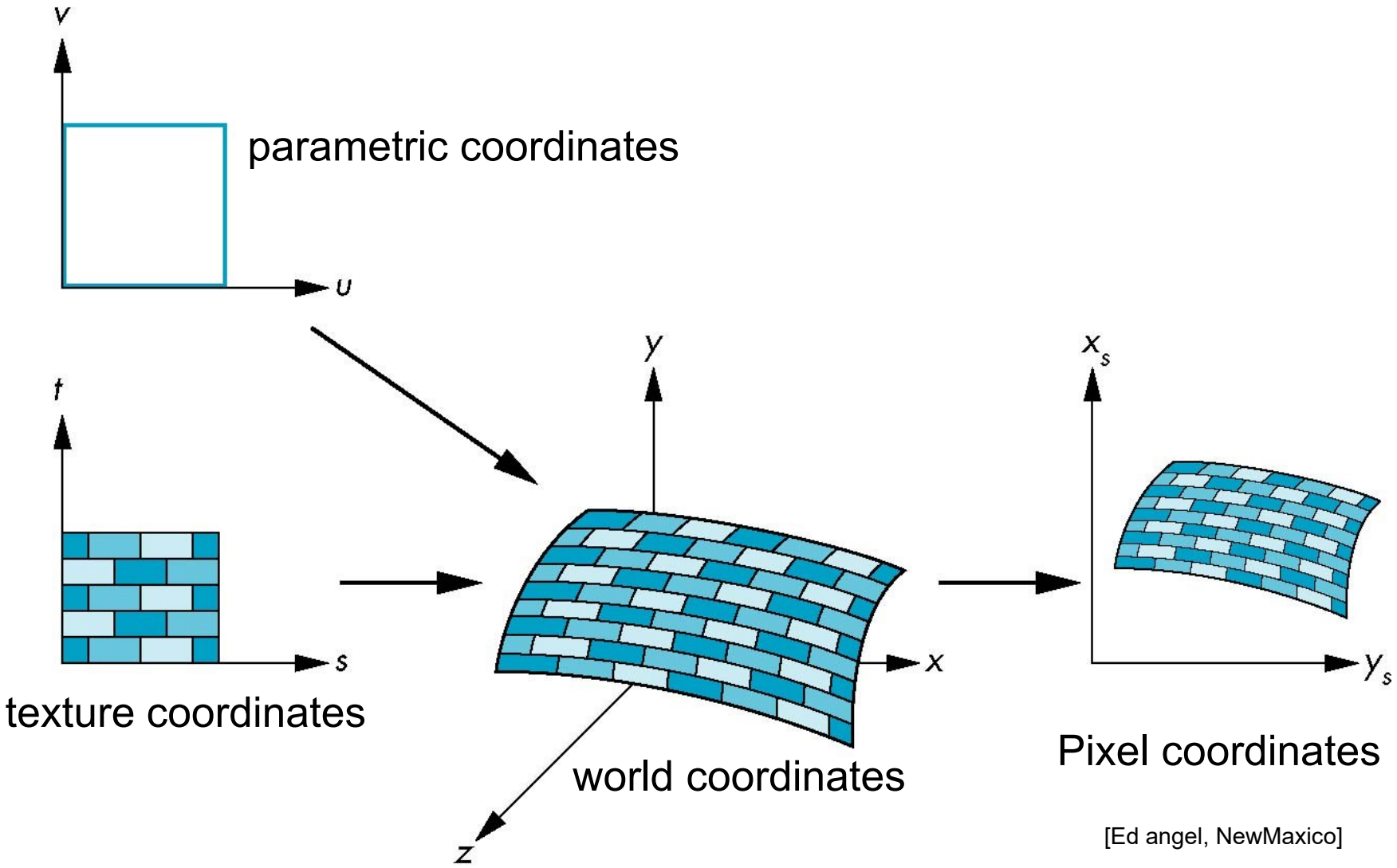
- 1D/Linear texture
- 2D/Surface texture
- 3D/Volume texture

Surface texture mapping

- Texture coordinates (u, v) for each vertex



Surface texture mapping



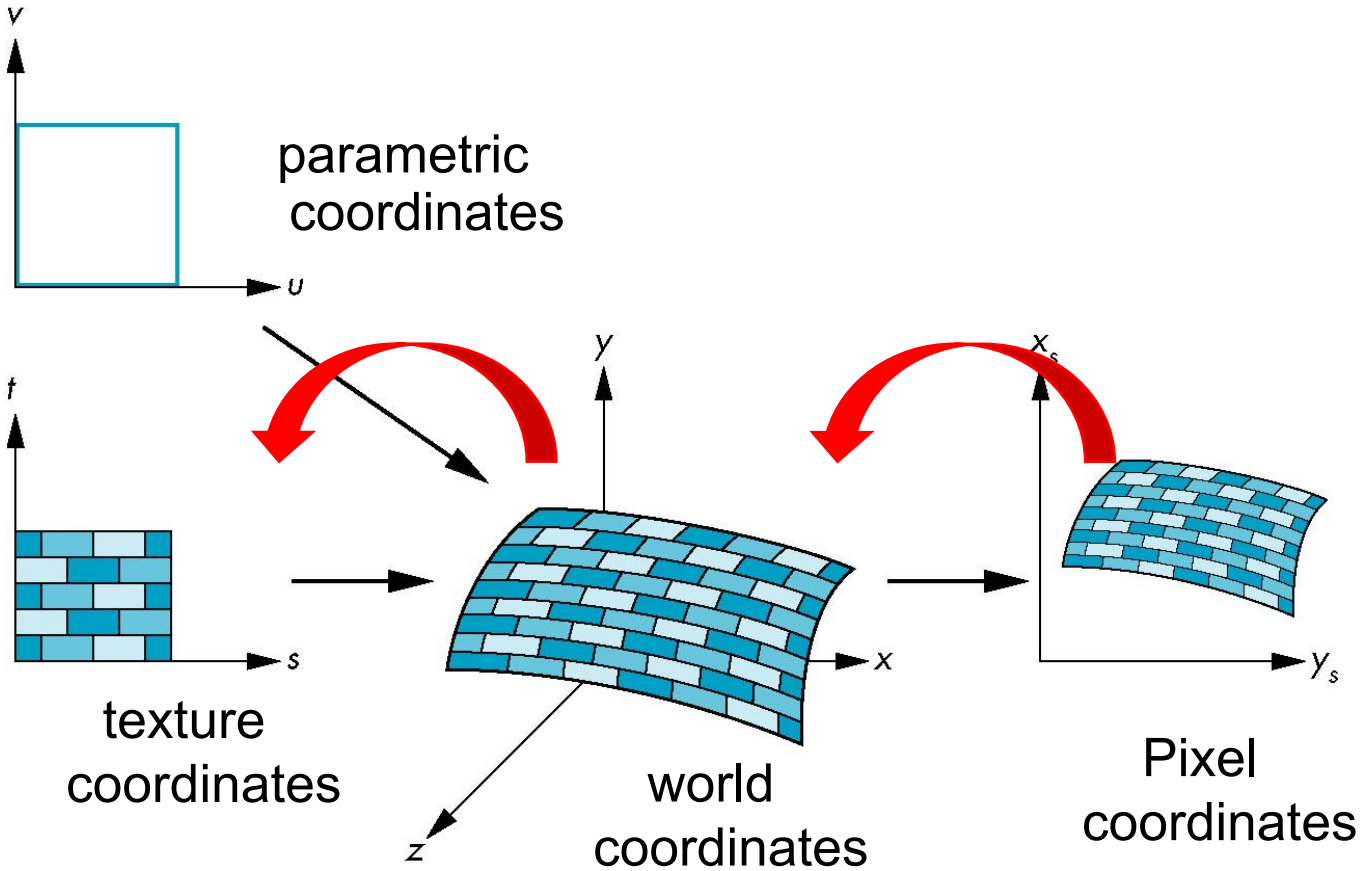
[Ed angel, NewMaxico]

Mapping functions

- We need to find the mapping from texture coordinate to parametric coordinate
  - u = u(s, t)
  - v = v(s, t)
- We also need to find the mapping from the parametric coordinate to object coordinates
  - u = u(x, y, z)
  - v = v(x, y, z)

Mapping functions

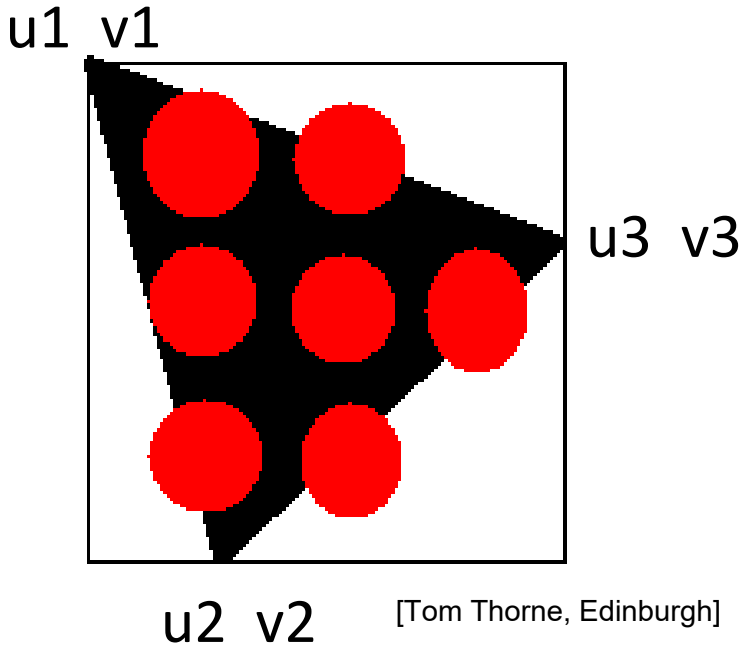
- In common rendering framework, the **backward mapping** is adopted
  - Why?



[Ed angel, NewMexico]

UV Interpolation

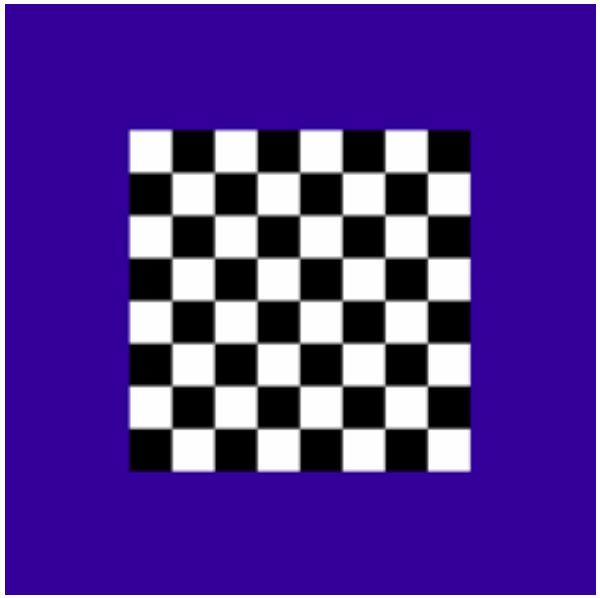
- Pixel coordinate to uv coordinates (2D to 2D)
- We can using barycentric coordinates or scanning conversion
- However uv cannot be interpolated linearly in screen space
  - Why?
  - Perspective distortion



[Tom Thorne, Edinburgh]

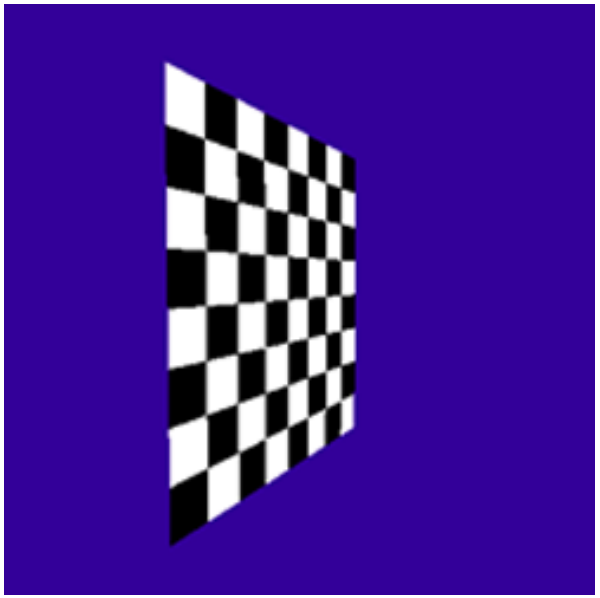
UV Interpolation

- Linear interpolation in screen space:



texture source

what we get

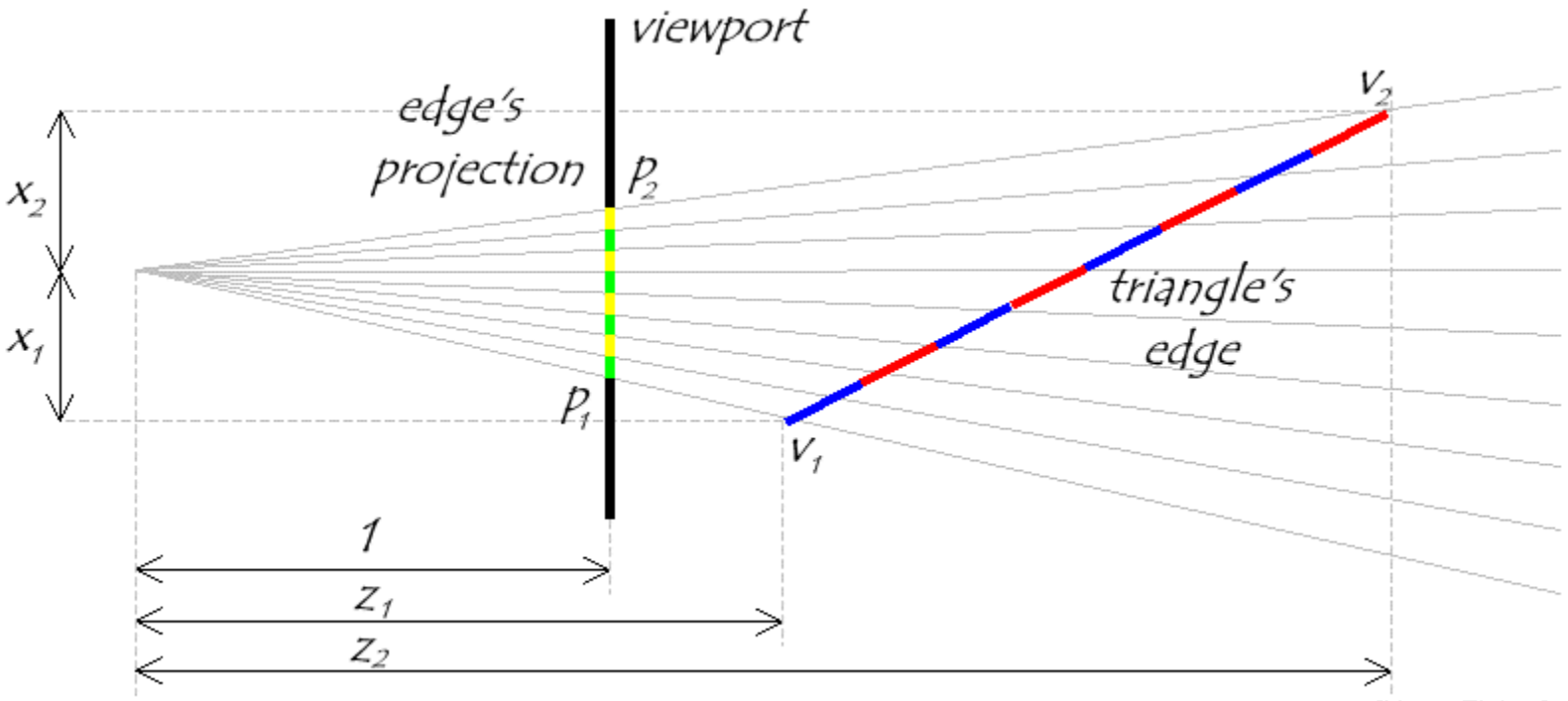


what we want

[Tom Thorne, Edinburgh]

Perspective correction

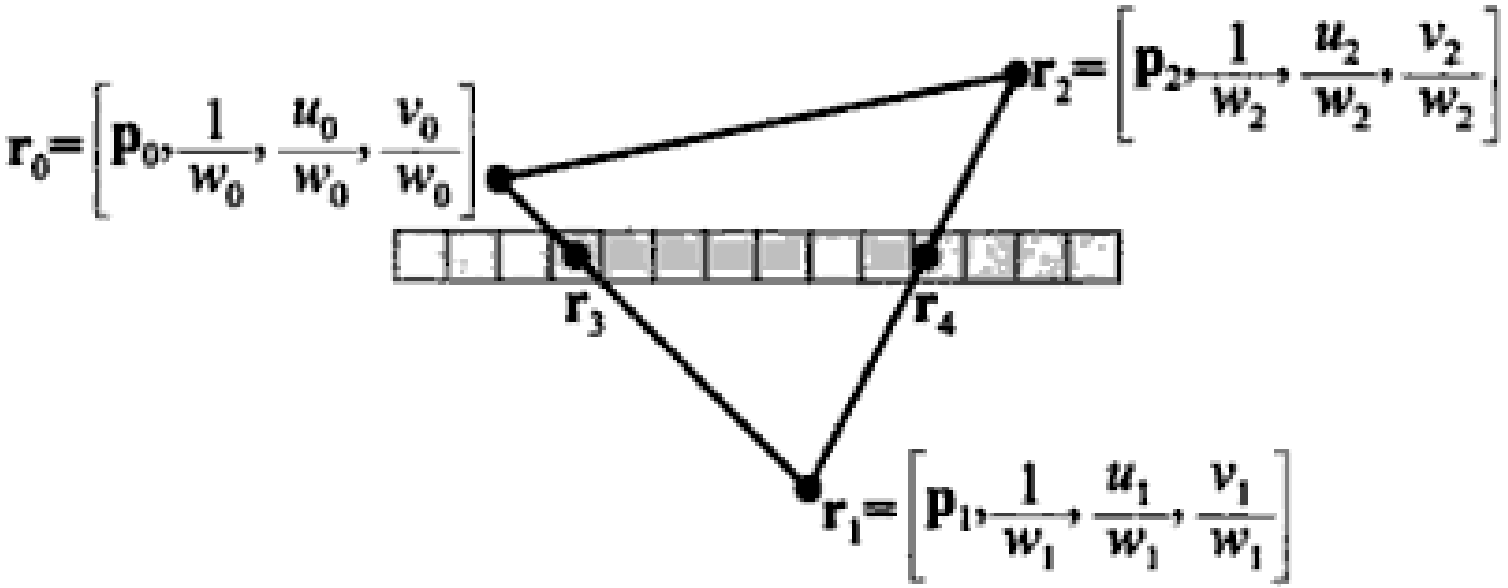
- Uniform steps on the image plane does not correspond to uniform steps along the edge



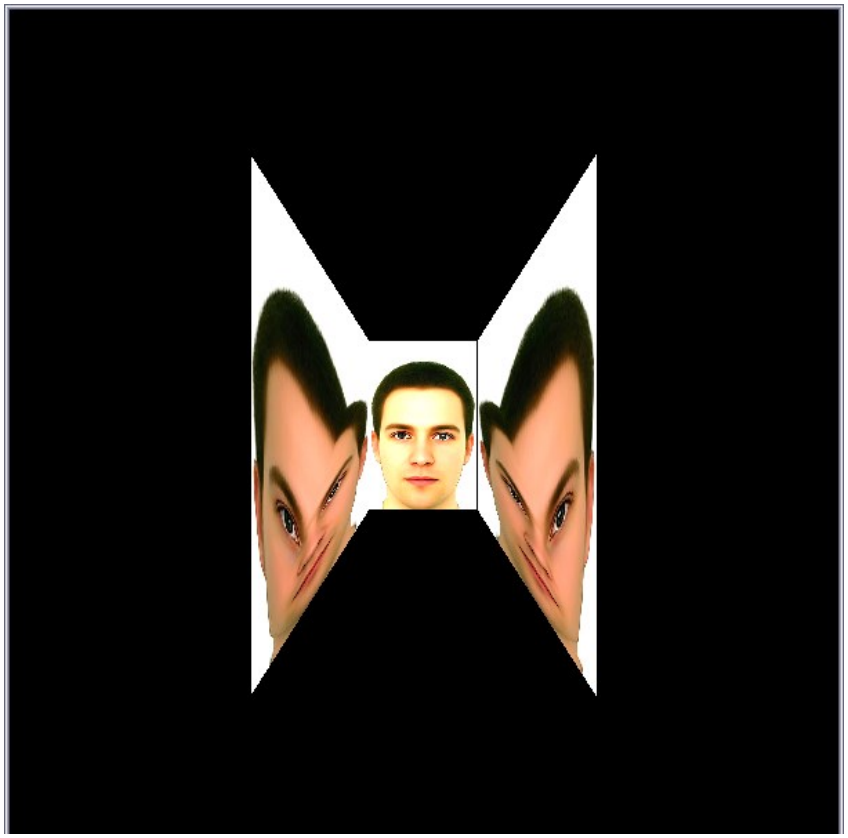
[Hugo Fisher]

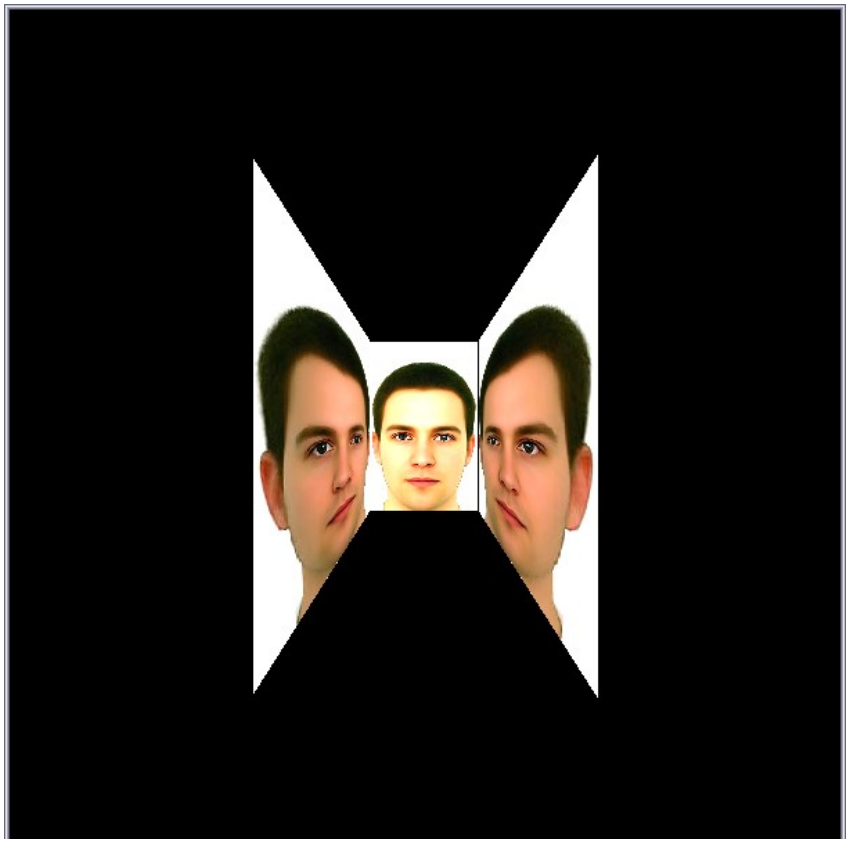
Perspective correction

- (u, v) should be divided by the depth
- using (u/w, v/w) in homogeneous coordinates



[Tom Thorne, Edinburgh]

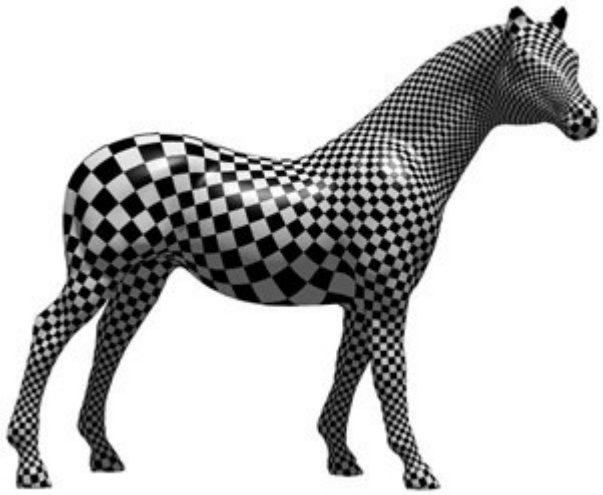




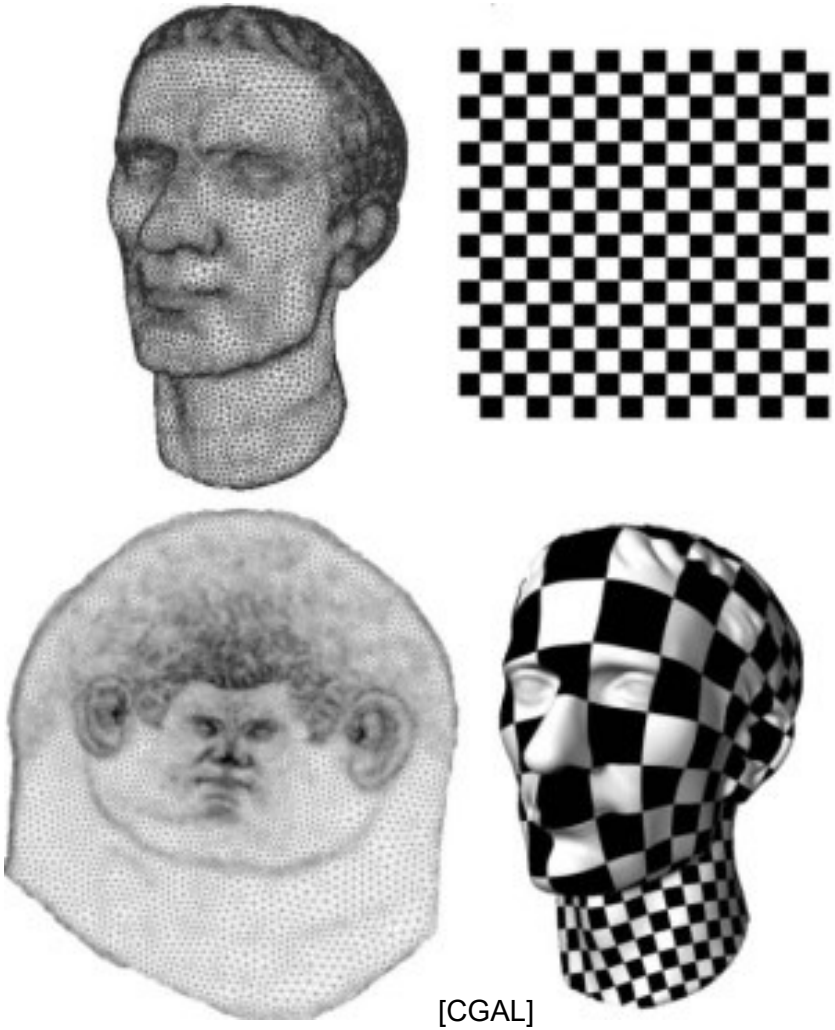
[Tom Thorne, Edinburgh]

Mesh parameterization (UVMapping)

- u = u(x, y, z)
- v = v(x, y, z)



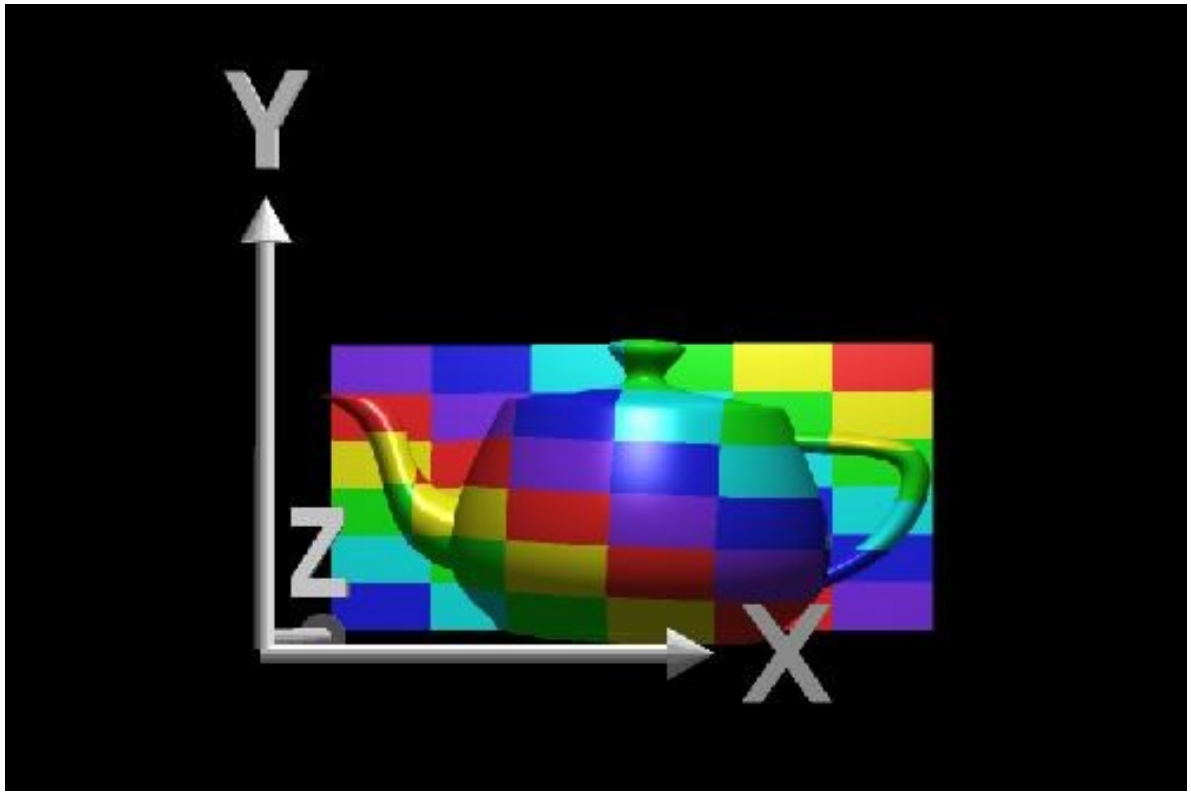
[RiemannMapper]



[CGAL]

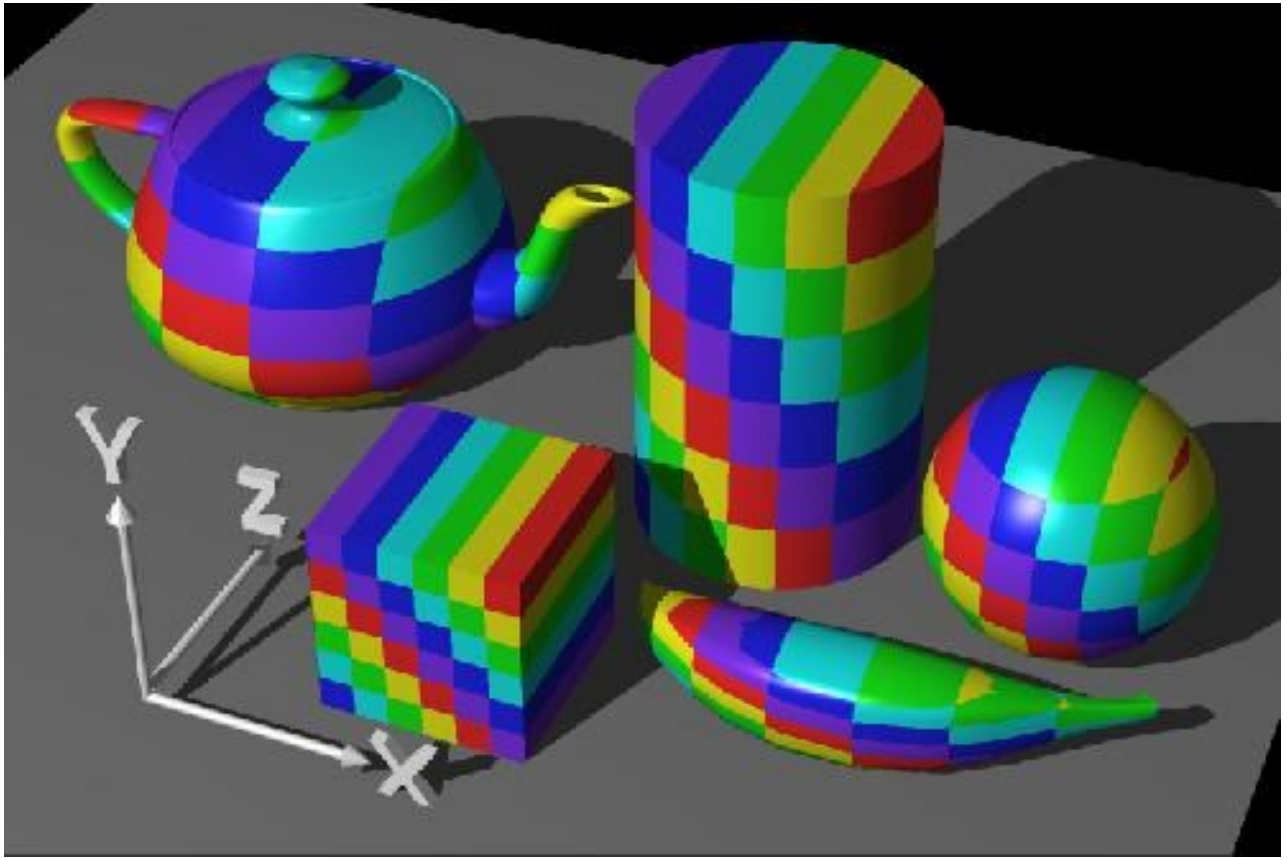
UVMapping - How to generate texture coordinates

- Where do we find the location in the texture map?



[Rosalee Wolfe / DePaul University]

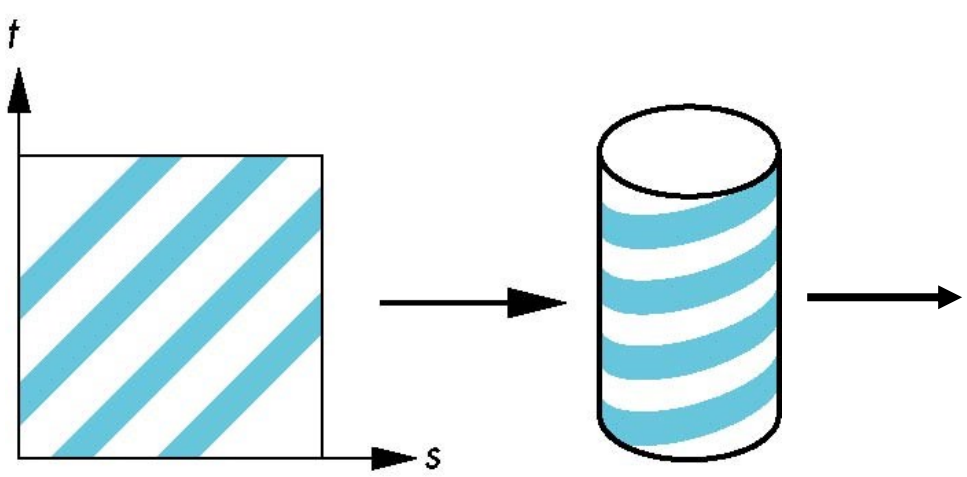
Orthogonal mapping



[Rosalee Wolfe / DePaul University]

Cylindrical mapping

- One solution is to first map the texture to an intermediate e.g. a cylinder or a sphere



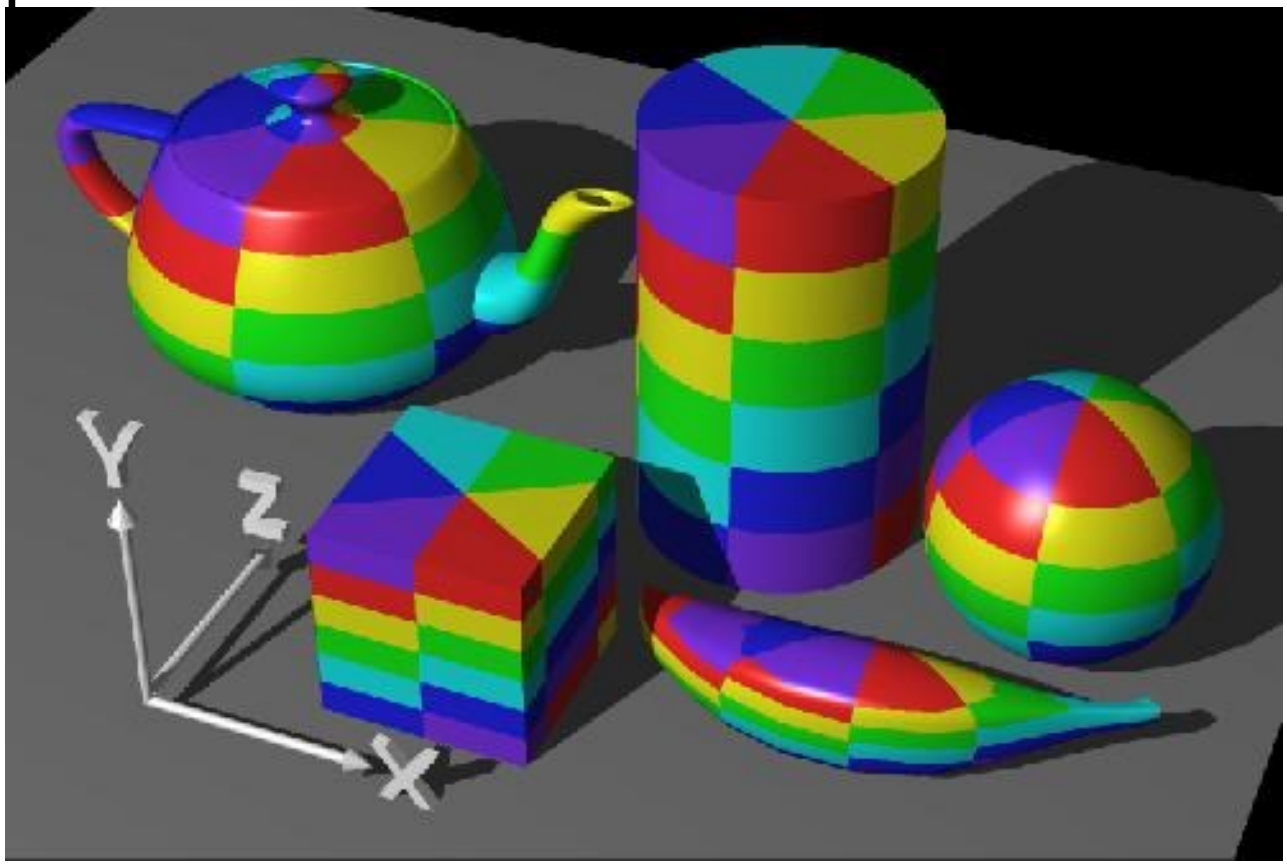
[Rosalee Wolfe / DePaul University]

Cylindrical mapping

- Cylinder parameterization
  - Using proper coordinate system (u, v)
    - $x = r\cos u$
    - $\mathrm{y} = r\sin u$
    - $z = v$
  - Then find $u = u(s,t), v = v(s,t)$ which is linear
    - $s = 0 \rightarrow u = 0,\ s = 1 \rightarrow u = 2\pi$
    - $t = 0 \rightarrow v = 0,\ t = 1 \rightarrow v = h$
    - $u = 2\pi s,\ v = ht$
  - This maps u v space to cylinder of radius r and height h in world coordinates

Cylindrical mapping

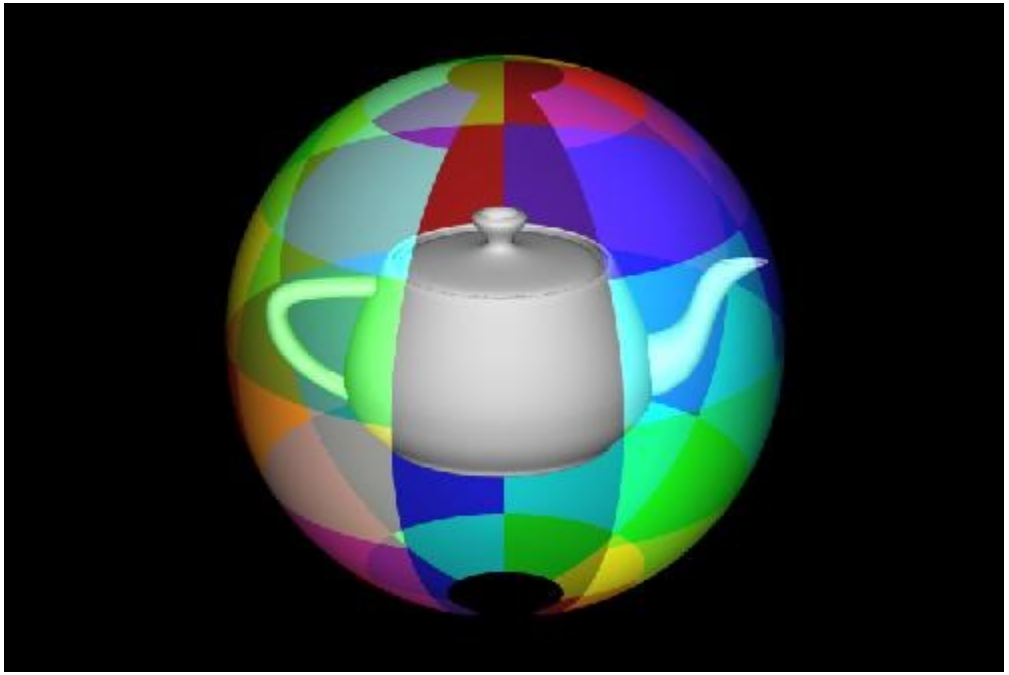
- Then the uv is mapped orthogonally to the shape



[Rosalee Wolfe / DePaul University]

Spherical mapping

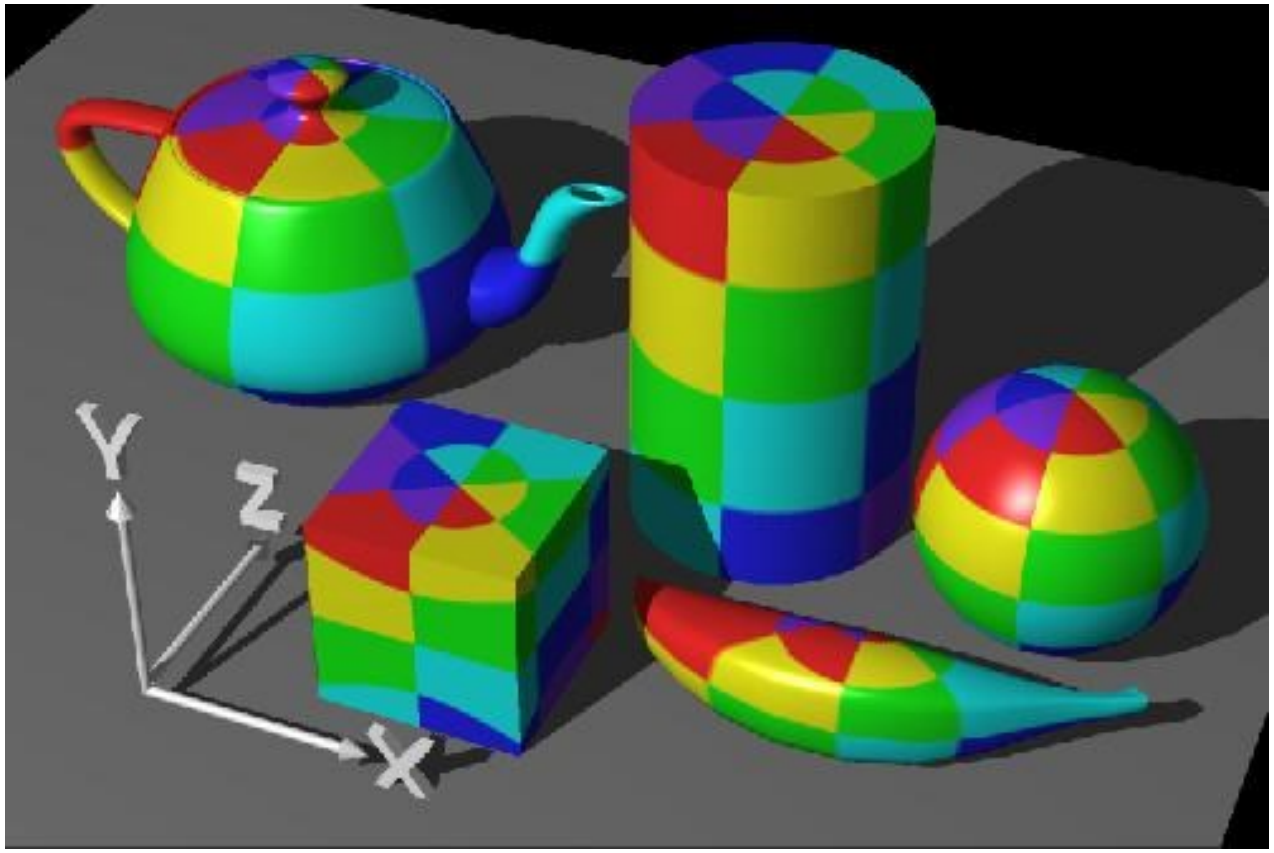
- We can further use a parameterized sphere
  - $x = rcosu$
  - $y = rsinu \cdot cosv$
  - $z = rsinu \cdot sinv$



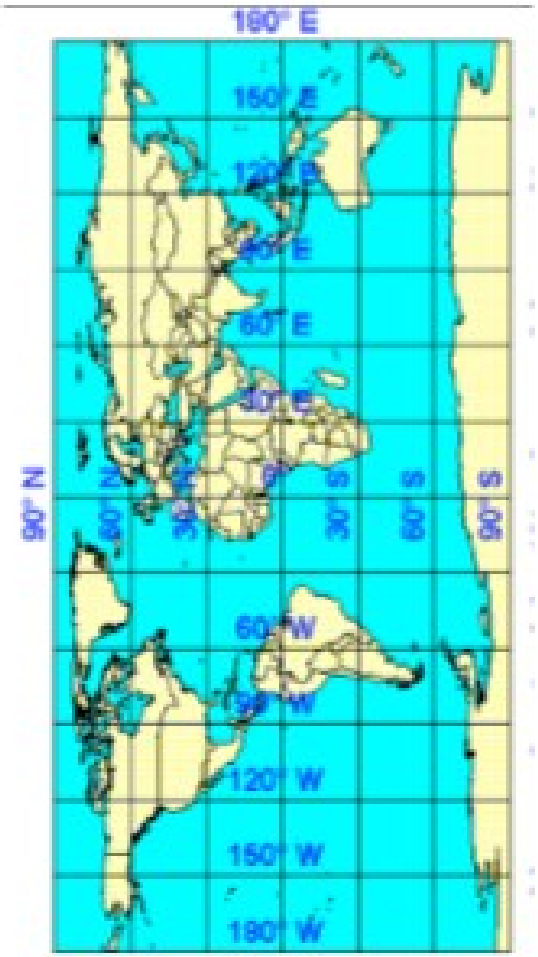
[Rosalee Wolfe / DePaul University]

Spherical mapping

- Then the uv is mapped orthogonally to the shape



[Rosalee Wolfe / DePaul University]



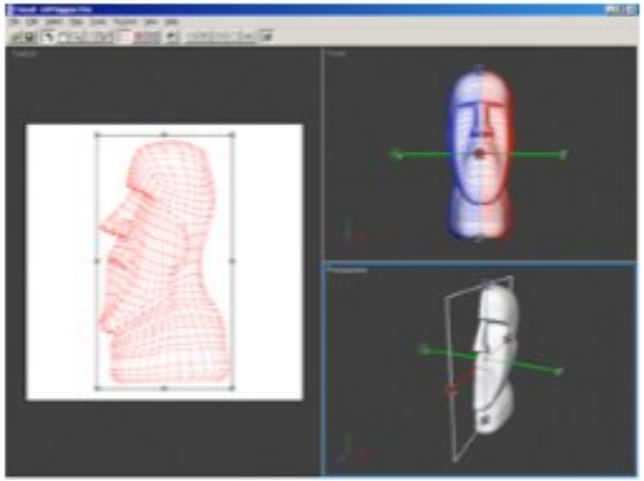
[map: Peter H. Dana]

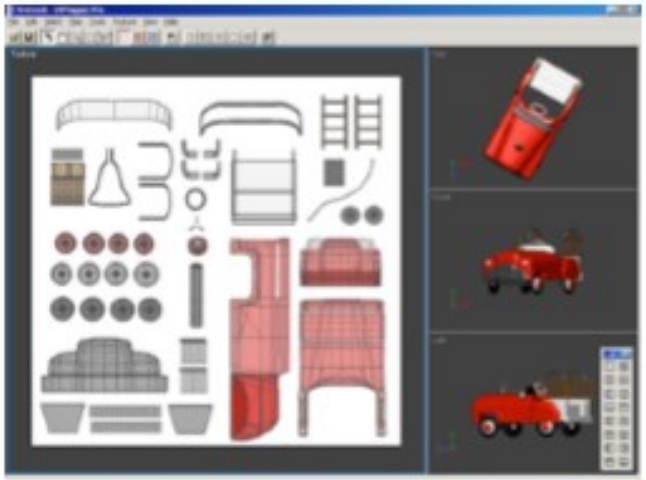
UVMapping

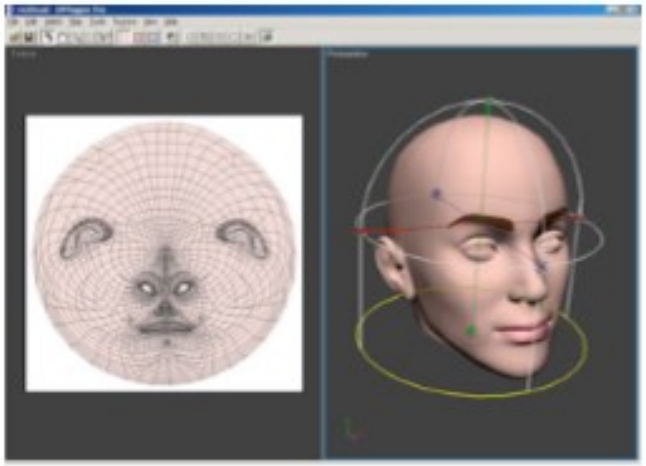
- UV mapper in modeling software

Example: UVMapper

http://www.uvmapper.com

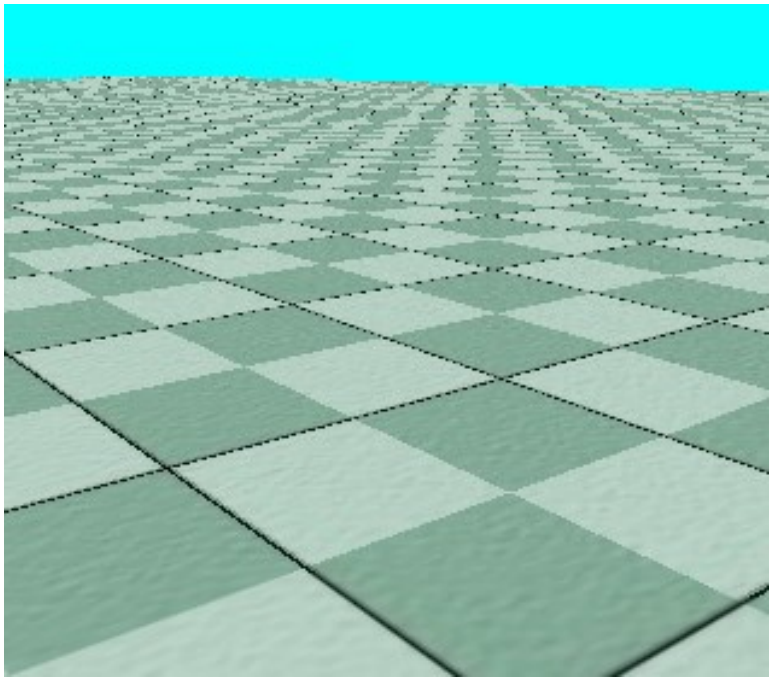


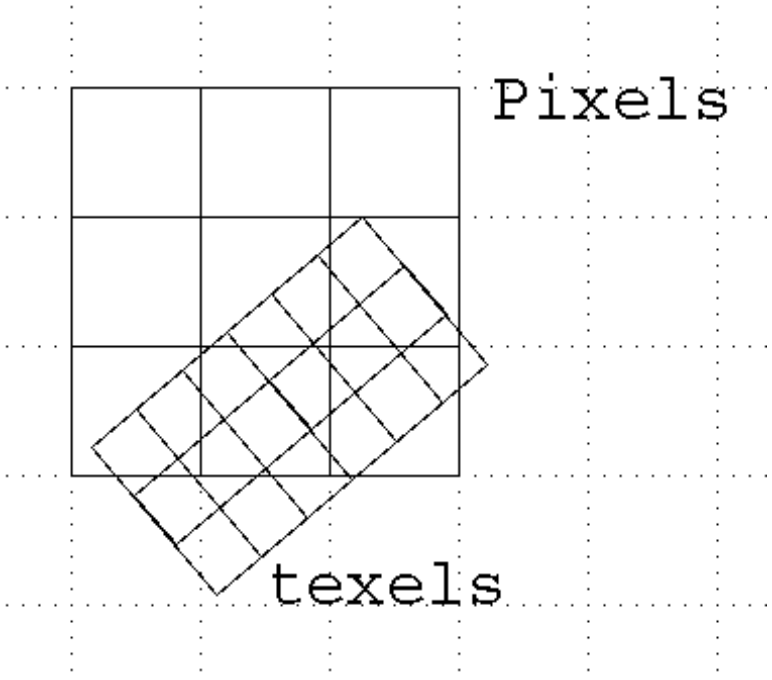




Aliasing of textures

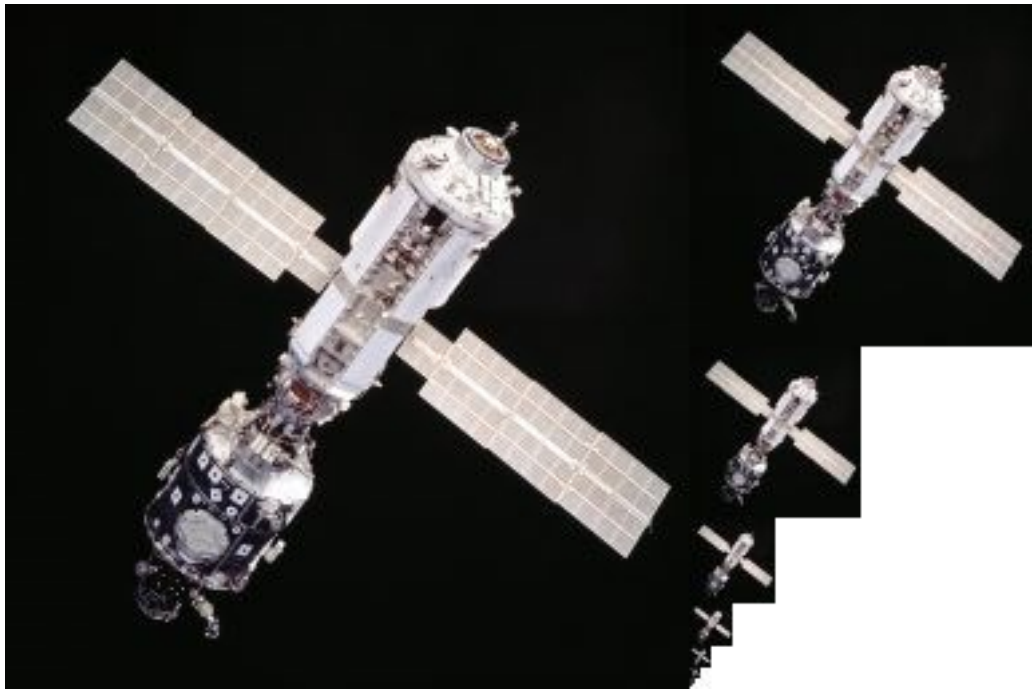
- When several texels covering a pixel
- or the textured surface is magnificated too much (we see pixels!)





Mipmapping

- Use a texture of multiple resolutions
- Switch the resolution according to the number of texels in one pixel




tom's hardware guide
No Mipmapping
With Mipmapping


tom's hardware guide
256x256
128x128
64x64
32x32
...
LOD0
LOD1
LOD2
LOD3

OpenGL

```
imgData = LoadAnImage(filename);
unsigned int texture;
glGenTexture(1, &texture);
glBindTexture(GL_TEXTURE_2D, texture);
// set the texture wrapping/filtering options (on the currently bound texture object)
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
GL_LINEAR);
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, w, h, 0, GL_RGB,
GL_UNSIGNED_BYTE, ImgData);
glGenerateMipmap(GL_TEXTURE_2D);
```

GLSL

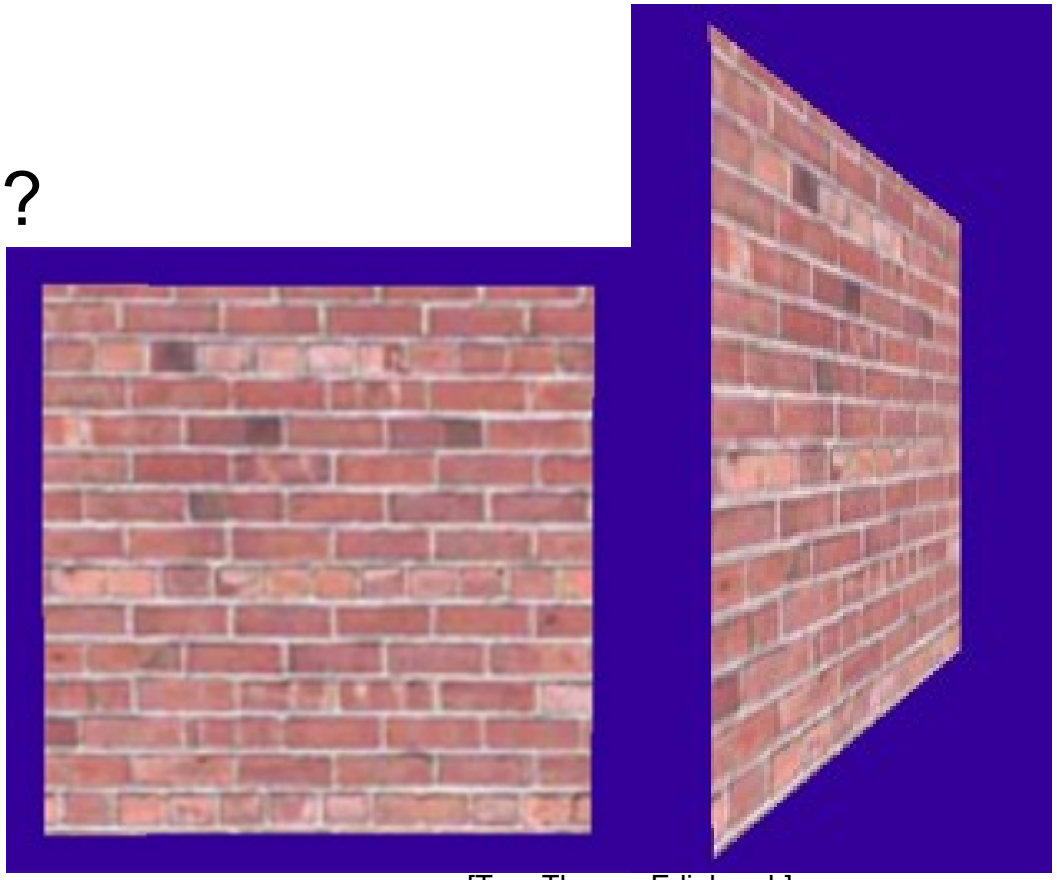
```
 //Vertex shader
#version 330
layout (location = 0) in vec3 pos;
layout (location = 1) in vec3 color;
layout (location = 2) in vec2 tex;
out vec3 outColor;
out vec2 texCoord;
void main()
{
    gl_Position = vec4(pos, 1.0); outColor = color; texCoord = tex;
}

//Fragment shader
#version 330
out vec4 FragColor;
in vec3 outColor;
in vec2 texCoord;
uniform sampler2D myTexture;
void main()
{
    FragColor = texture(myTexture, texCoord);
}
```

Other texturing techniques

- Bump mapping
- Displacement mapping
- Environmental mapping

Bump mapping

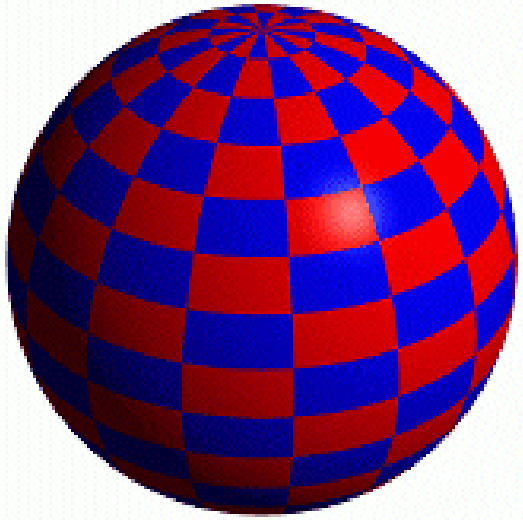
- Now using textures, we can model a wall using two triangles
- Is it realistic?
- How about shading?

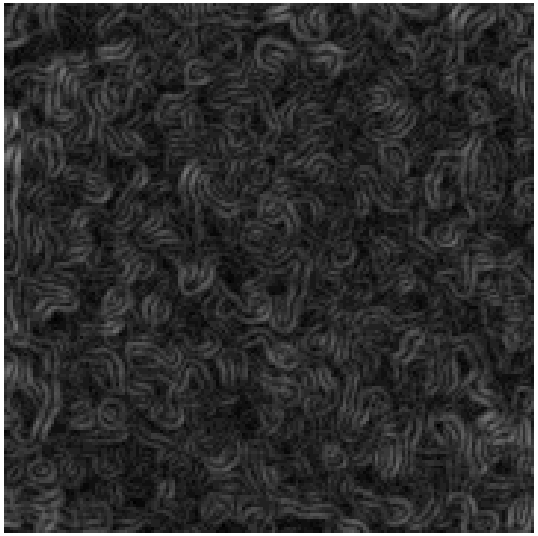


[Tom Thorne, Edinburgh]

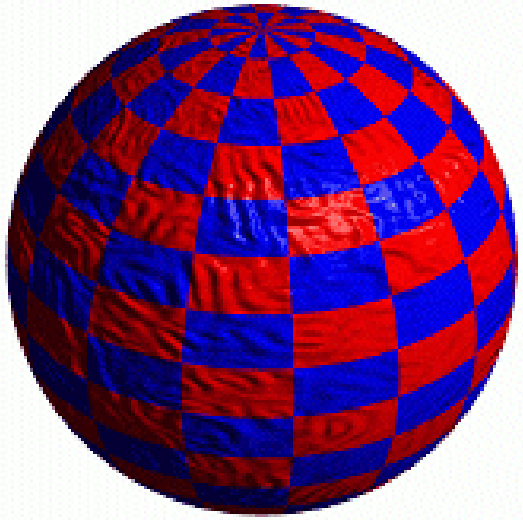
Bump mapping

- Recall that the normal defines the shading and a plane only have one normal for the wall
- We can use texture to disturb the normal!





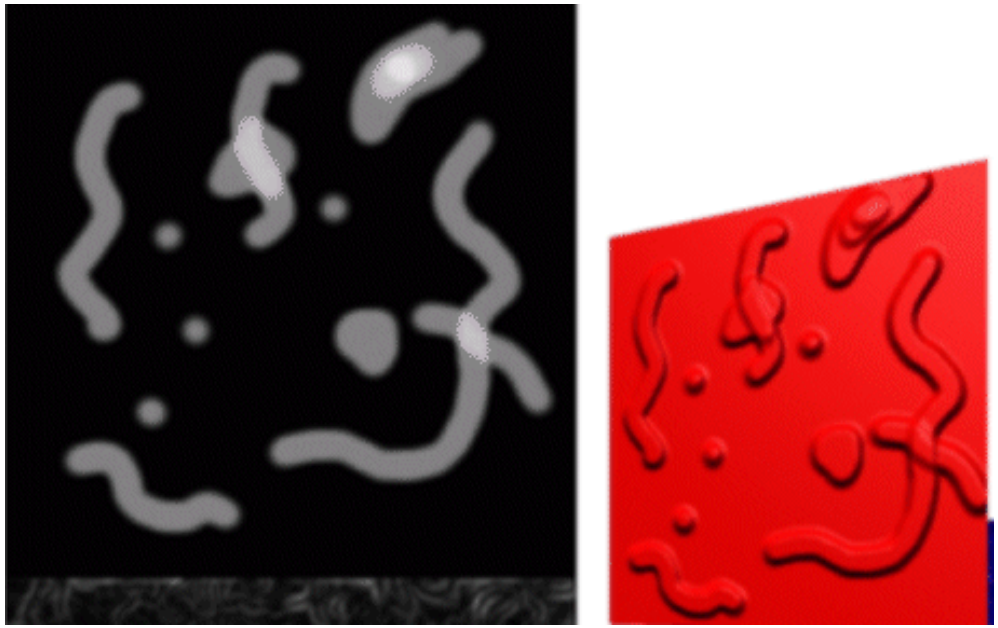
Swirly Bump Map



[Tom Thorne, Edinburgh]

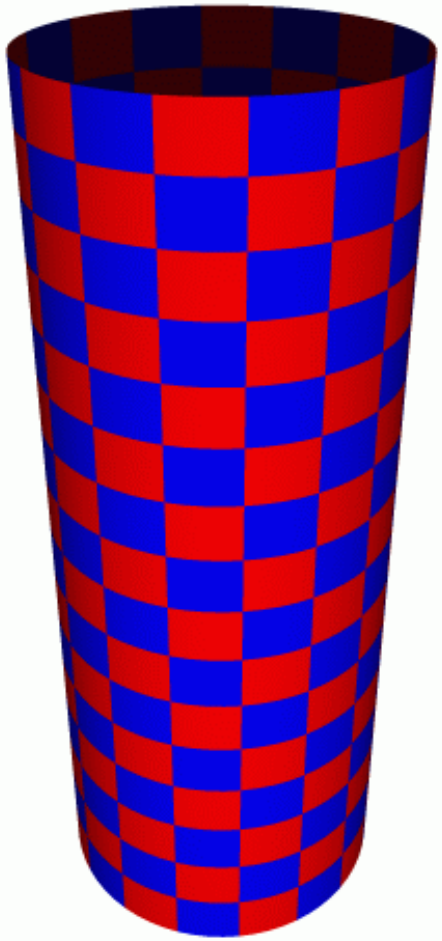
Bump mapping

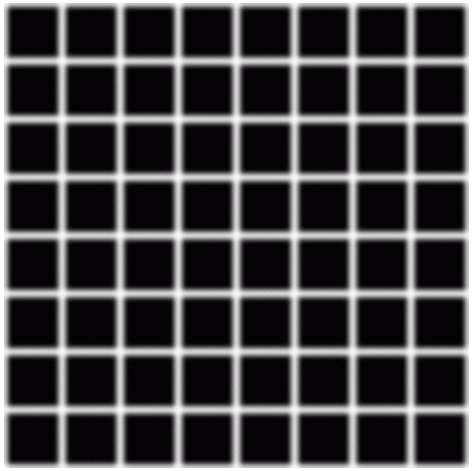
- Treat the texture as a height function
- Compute the normal from the partial derivatives in the texture

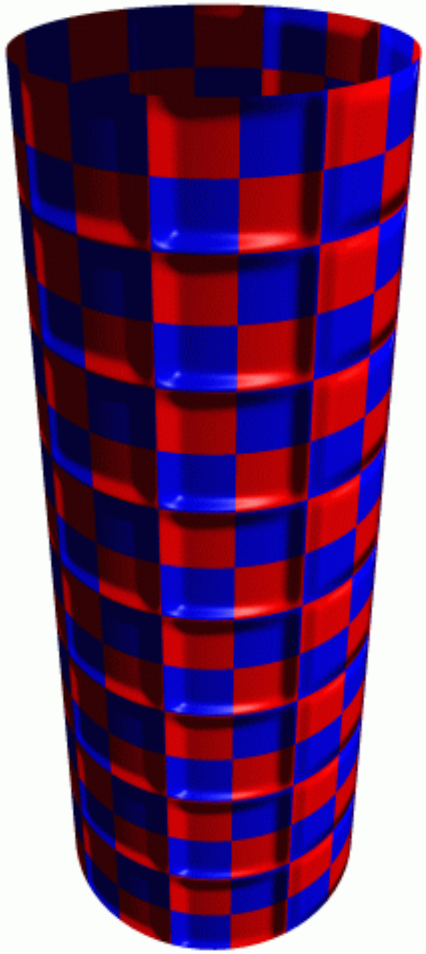


[Tom Thorne, Edinburgh]

Bump Mapping

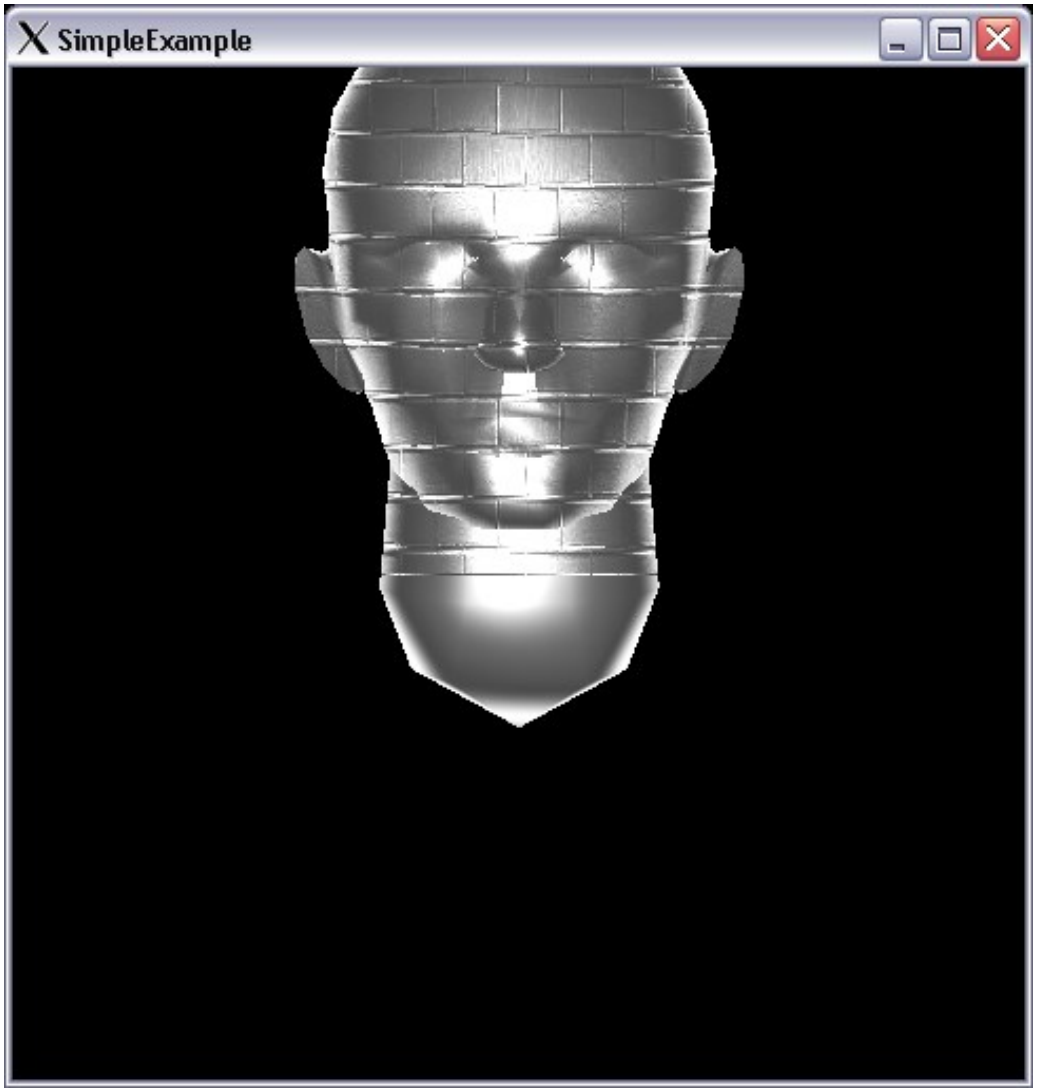


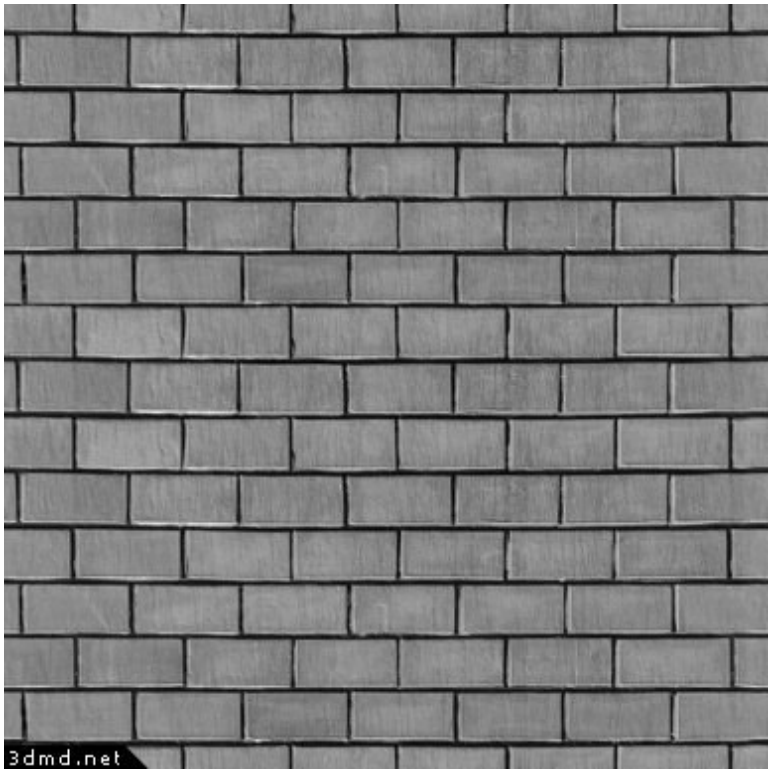




[Tom Thorne, Edinburgh]

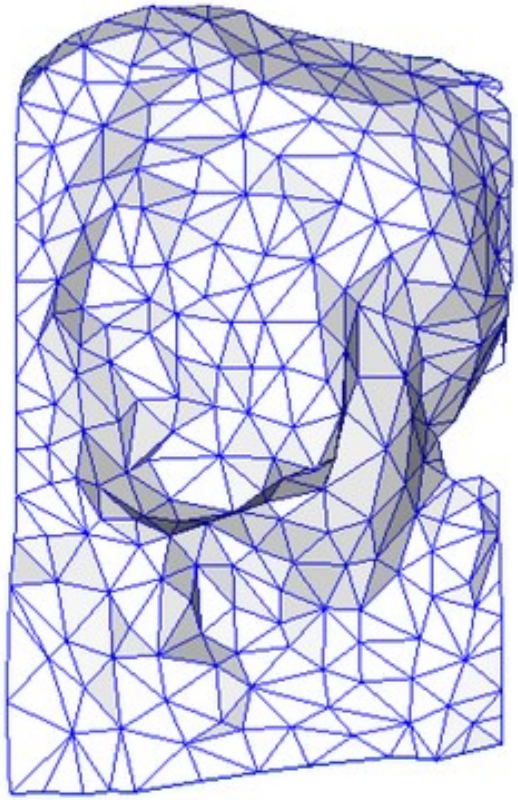
Bump mapping examples





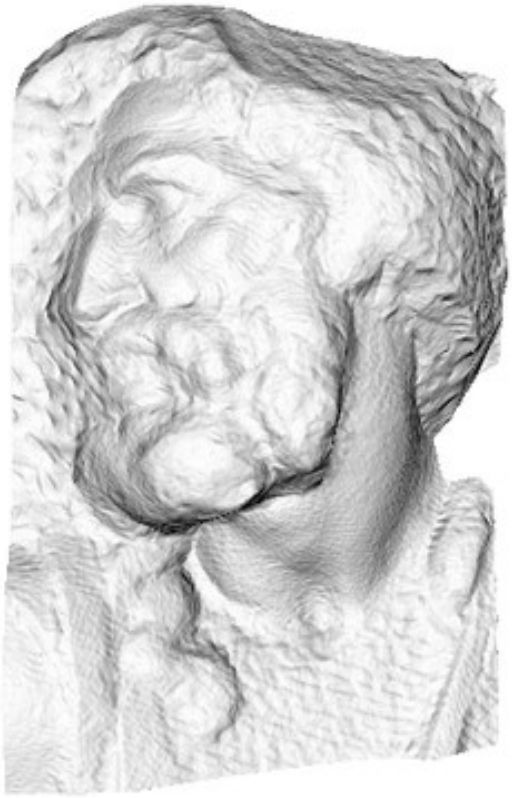
[Tom Thorne, Edinburgh]

original mesh
4M triangles



simplified mesh
500 triangles

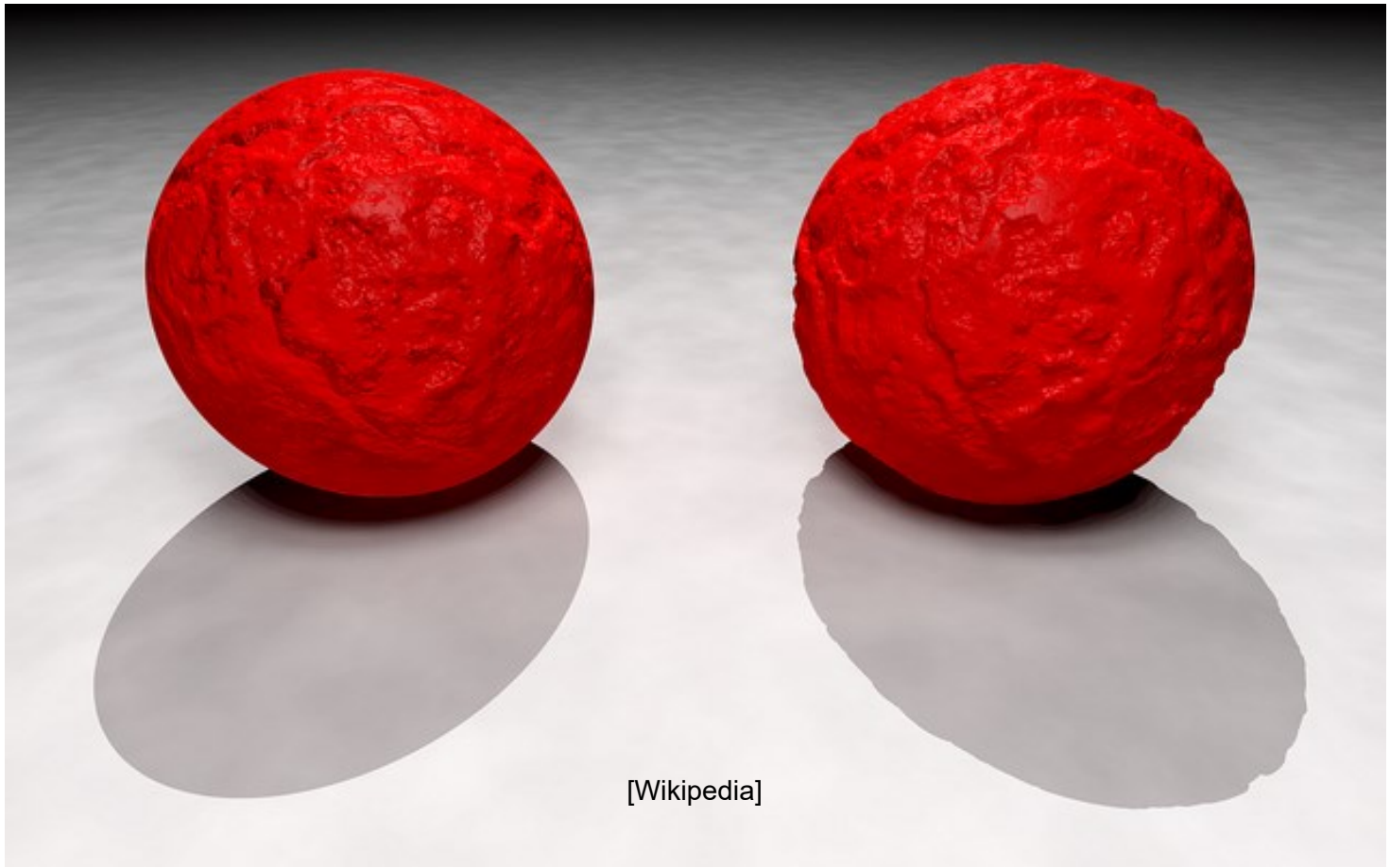
[Wikipedia]



simplified mesh
and normal mapping
500 triangles

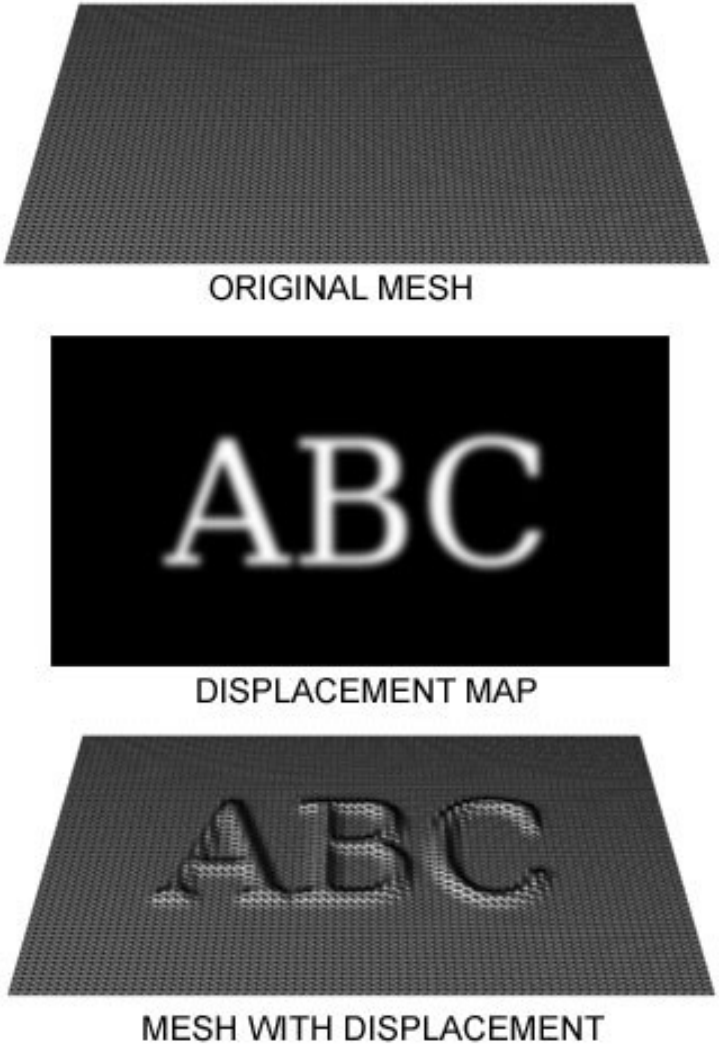
Bump mapping examples

- Notice the difference?

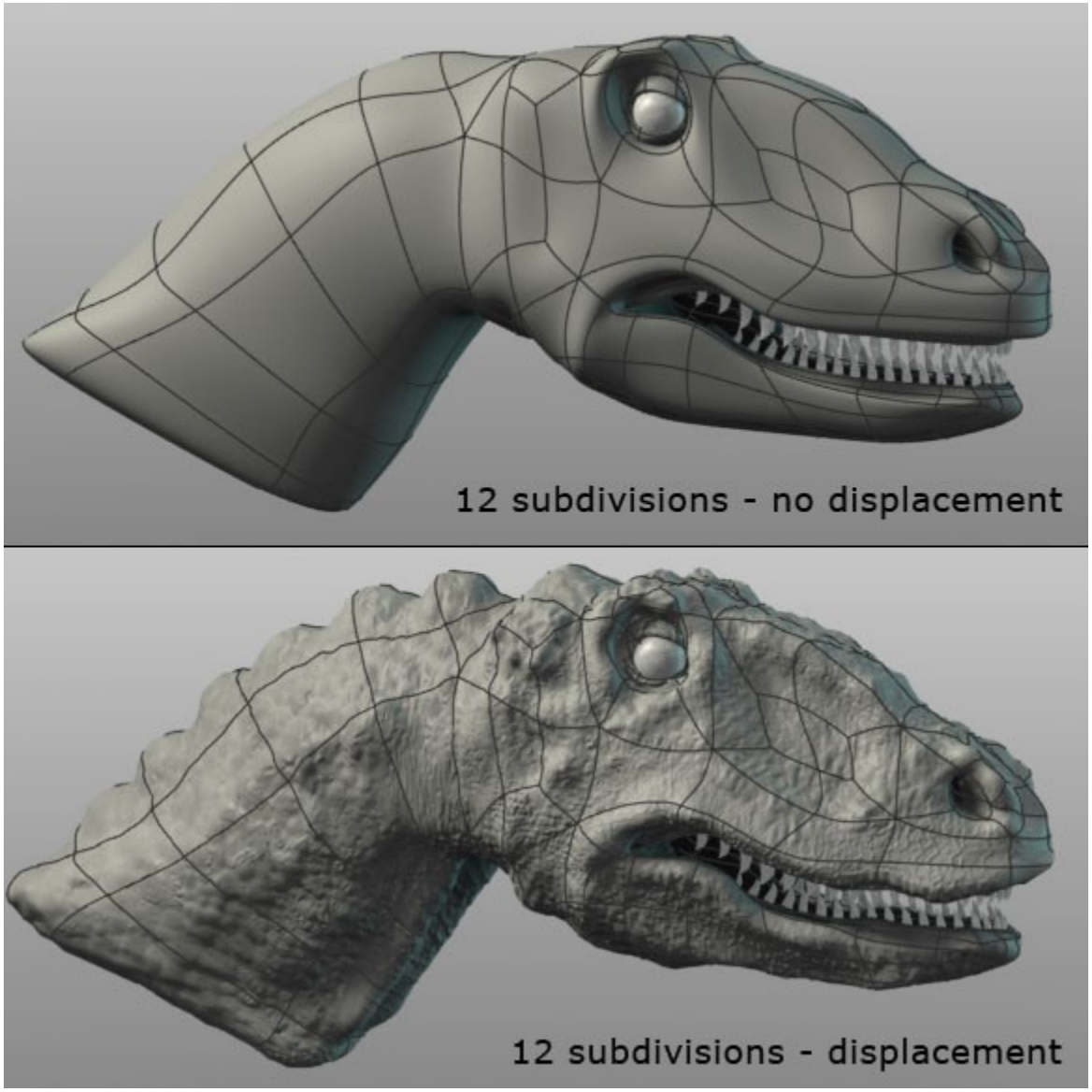


Displacement mapping

- Use the texture map to actually move the surface point
- The geometry must be displaced before visibility is determined



[Wikipedia]

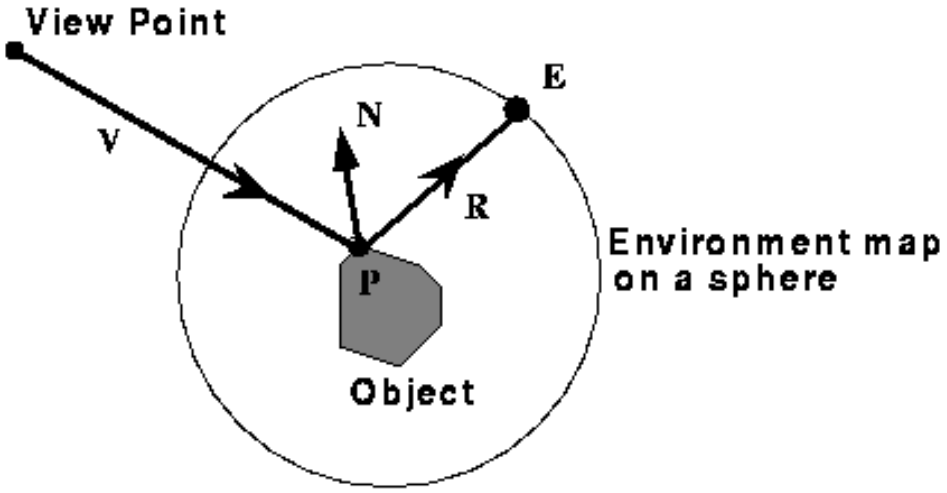


http://www.outside-hollywood.com/img/profilelg.jpg

http://www.cggallery.com/tutorials/displacement/

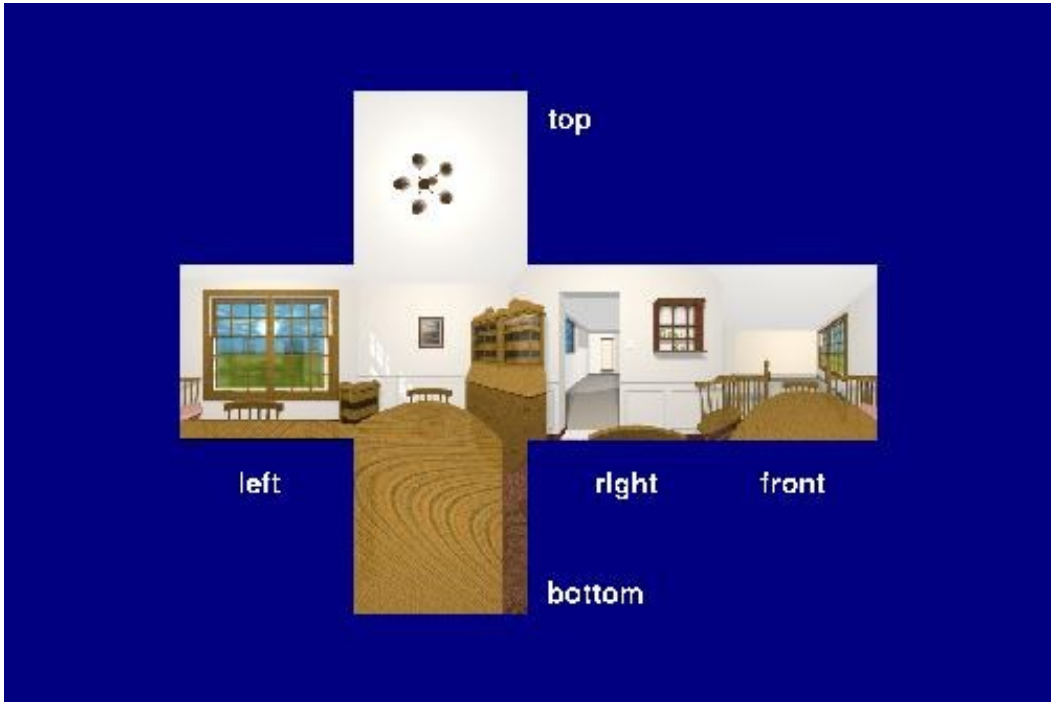
Environment mapping

- It’s difficult to generate reflections without GI method
- We can simulate the reflection using texture mapping!

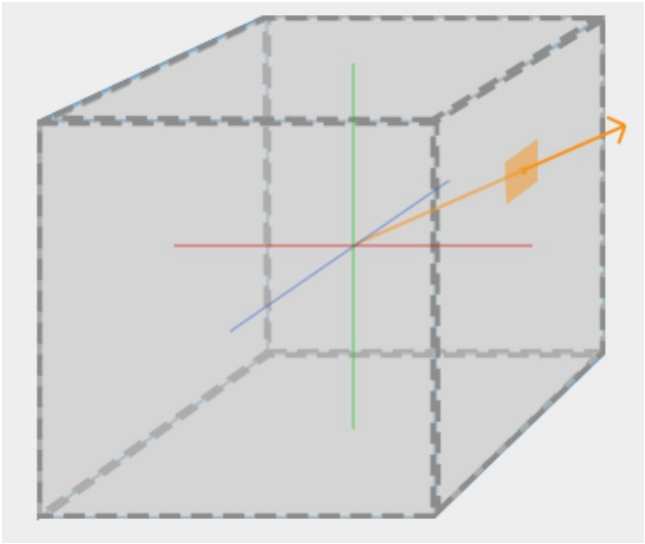


Cubic mapping

- Cubic texture maps
- 2D UV coordinates becomes 3D texture directions



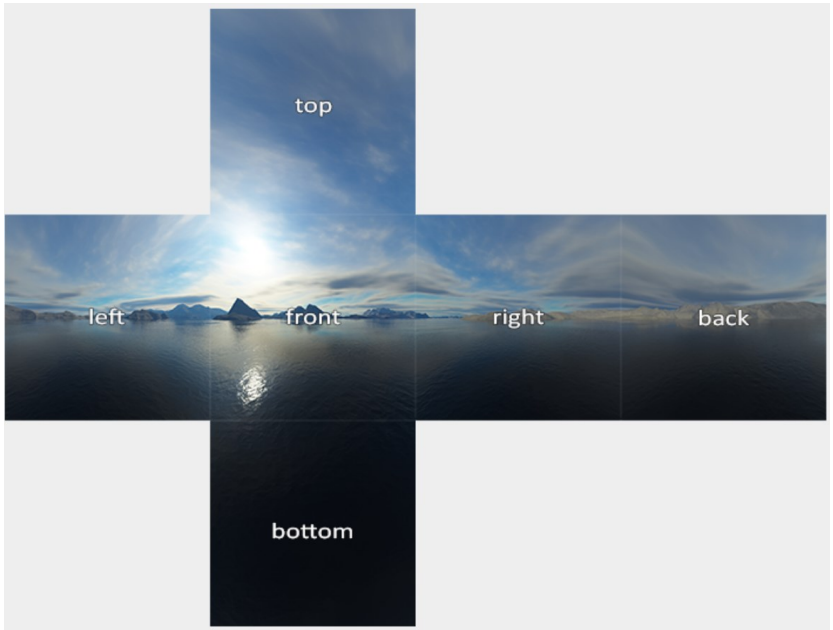
[Rosalee Wolfe / DePaul University]



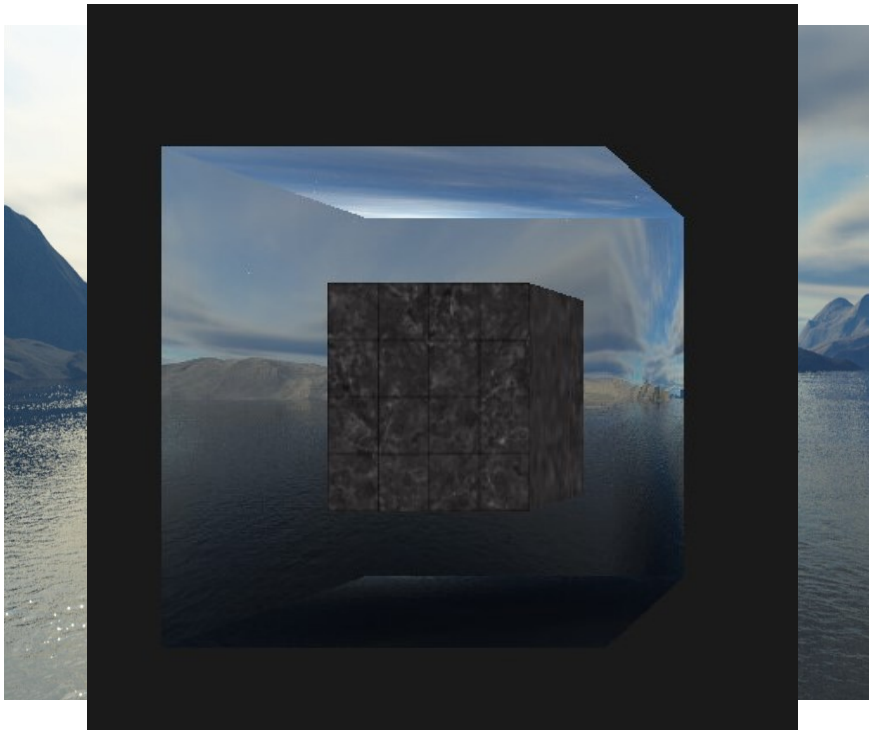
www.learningopengl.com

The Sky Box

- Render a Box
- Using vertex coordinates as texture directions
- Taking away the translation from view matrix



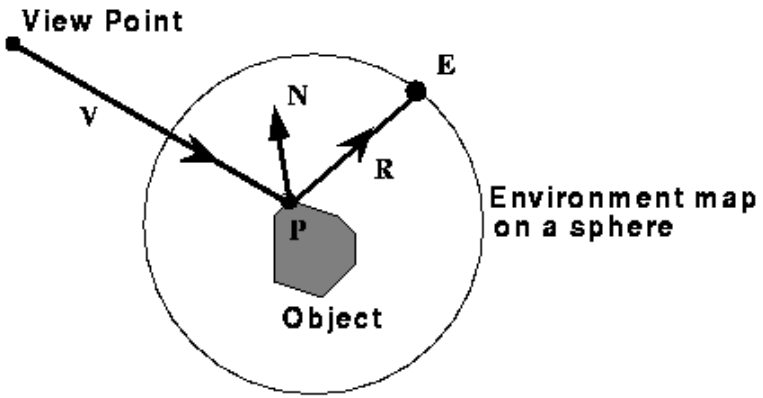
www.learningopengl.com



Environmental mapping

- Calculate proper texture direction in Fragment shader

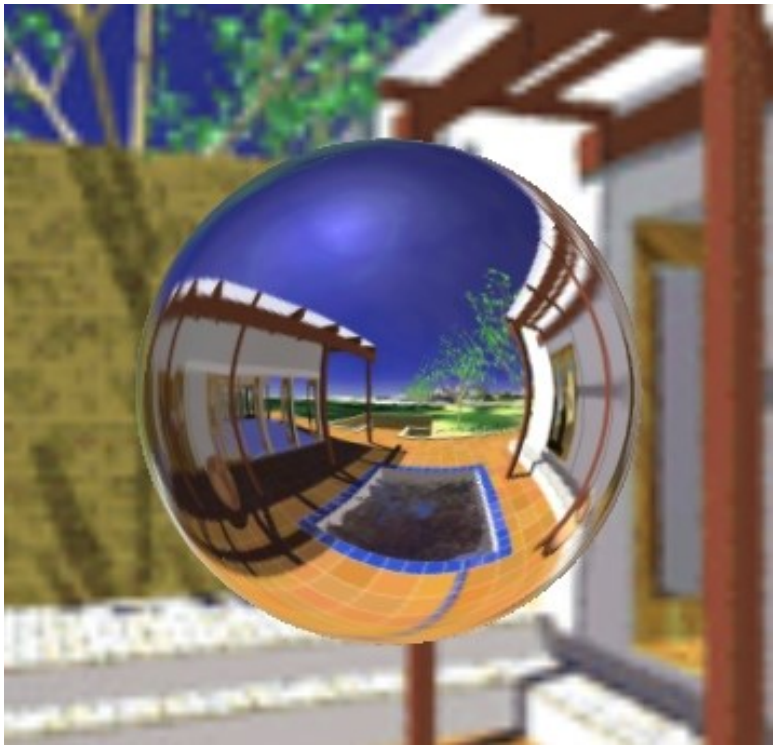
```
vec3 I = normalize(fragPos - cameraPos);
vec3 R = reflect(I, normalize(Normal));
FragColor = vec4(texture(skybox, R).rgb,
1.0);
```



[Rosalee Wolfe / DePaul University]

https://learnopengl.com/Advanced-OpenGL/Cubemaps

Examples



Examples

How to implement refraction?

References

- Ed Angel, CS433 Computer Graphics, NewMexico
- Cutler and Durand, MIT EECS 6.837
- Tom Thorne, COMPUTER GRAPHICS, The University of Edinburgh
- Steve Marschner, CS4620/5620 Computer Graphics, Cornell
- https://learnopengl.com/Getting-started/Textures
- https://learnopengl.com/Advanced-OpenGL/Cubemaps

- Questions?