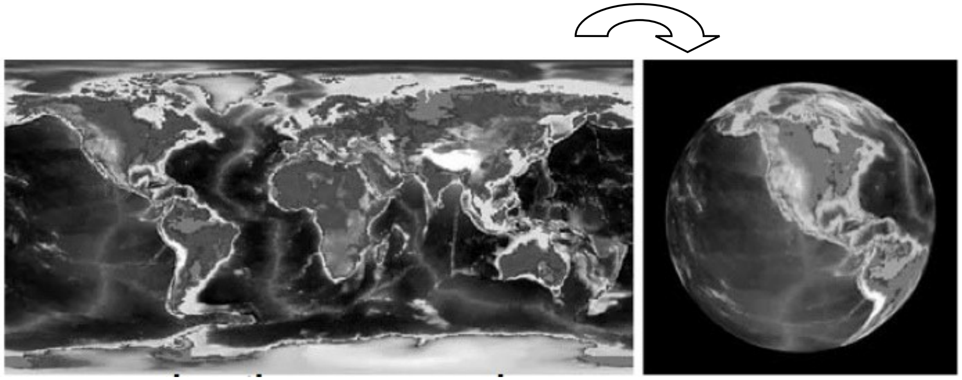
Computer Graphics:
4-Textures and shadows

Prof. Dr. Charles A. Wüthrich,
Fakultät Medien, Medieninformatik
Bauhaus-Universität Weimar
caw AT medien.uni-weimar.de

Textures

- We have been looking at light reflectance for surfaces which have no detail on the surface
- In fact, reality shows richness of surface detail.
- One could model the surface with detailed geometry
- However, this would increase greatly the complexity of the model.
- A better appproach is therefore to „paint“ detail on simple geometry
- The image, called texture, is „glued“ to a simple geometry to obtain detail
- First approaches due to Catmull (74) and Blinn & Newell (76)



Textures

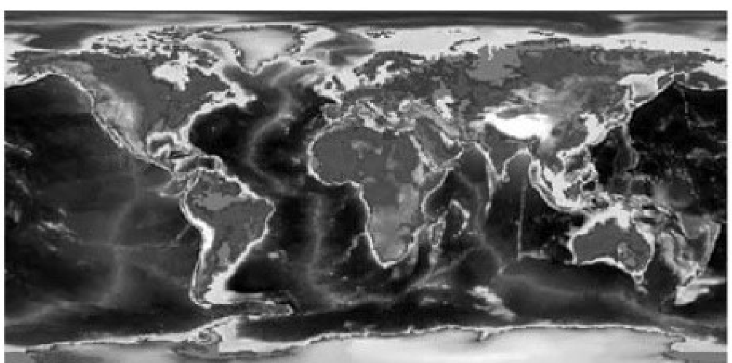
- There are basically two ways of texture mapping:
  - 2D
  - 3D
- Let us look first at 2D textures
- Image data (surface pixel colors) is stored in a 2D image, the pixels of which are called texels
- Let's assume the coordinates of the image are called u,v and that u and v vary in the interval [0,1]

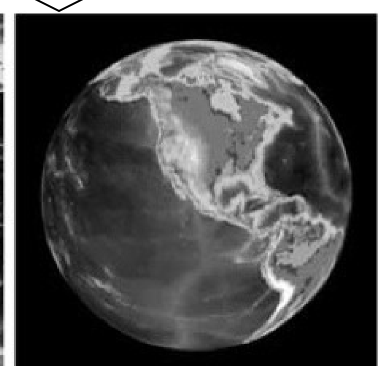
- To compute what colour is reflected by the sphere, one must find a correspondence between sphere and the texture space
- Parametric sphere:

$$\begin{cases} x=x_c+R\cos\psi\sin\vartheta \\ y=y_c+R\sin\psi\sin\vartheta \\ z=z_c+R\cos\pi \end{cases}$$

$$\left.\begin{array}{l} \vartheta= (z-z_c)/R \\ \psi=artan((y-y_c)/(x-x_c)) \end{array}\right\} \text{longitude +latitude}$$

- $$\left.\begin{array}{l} u= \psi /2\pi \\ v=(\pi-\vartheta)/\pi \end{array}\right\} \textit{to texture}$$

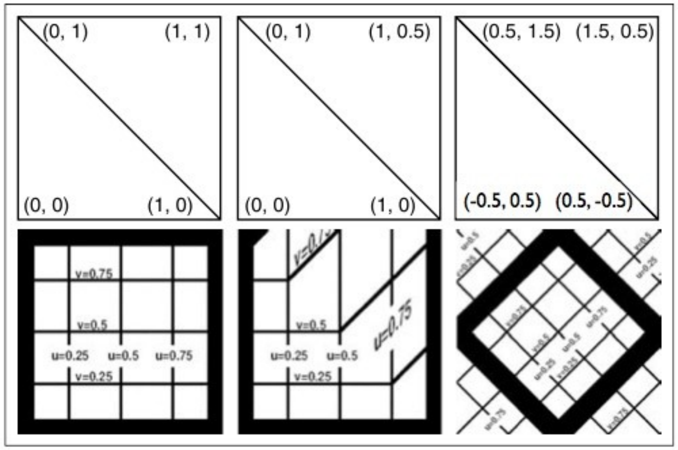


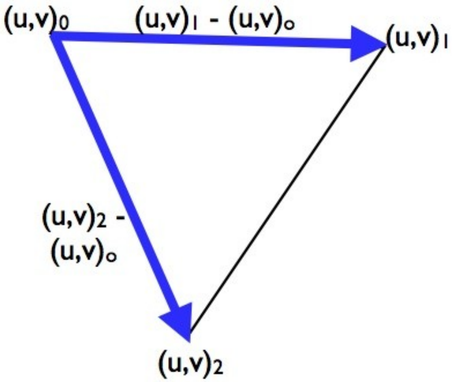


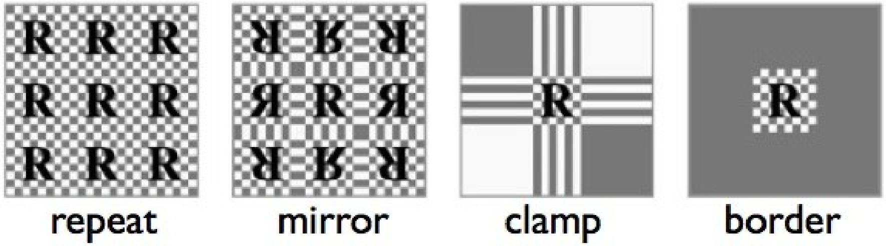
- Similarly, for other simple maps
  - Cube
  - Cylinder
  - Plane

Textures

- And what if my object is a mesh?
- Determine texture coordinate for each vertex of the mesh
- Bilinear interpolation between vertices
  - For triangles, use baricentric coordinates
    (same as done for normals)
- If texture coordinates are beyond the image, then



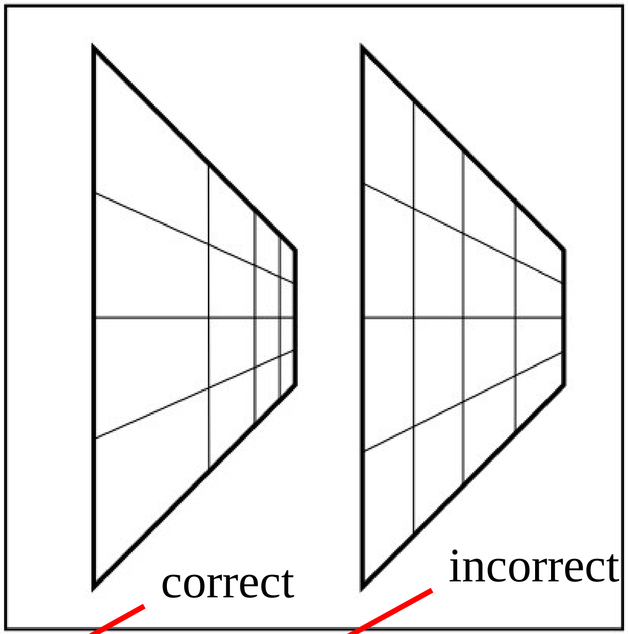




$$u(\beta,\gamma) = u_0 + \beta(u_1 - u_0) + \gamma(u_2 - u_0)$$
$$v(\beta,\gamma) = v_0 + \beta(v_1 - v_0) + \gamma(v_2 - v_0)$$

Correct Textures

- Be careful when you apply the texture to your object!
  - applying textures in screen space leads to WRONG results!
  - One has to apply the texture BEFORE the perspecive is done!
- Otherwise perspective is lost in the texture!
- This is different from what we do in shading, where things are done in screen space

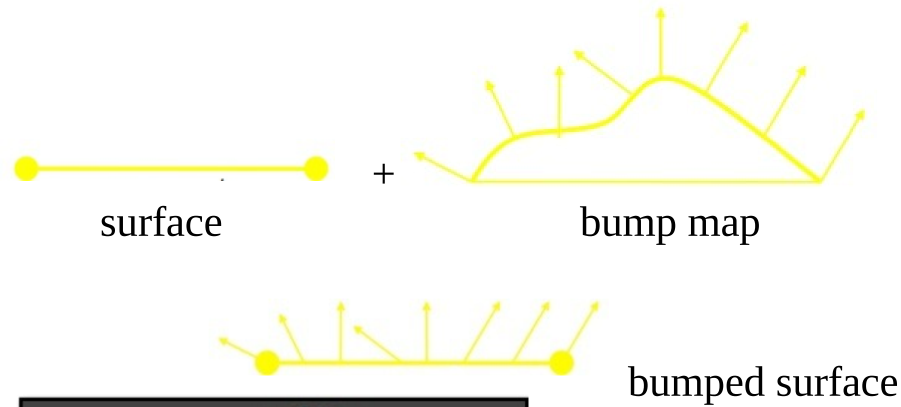


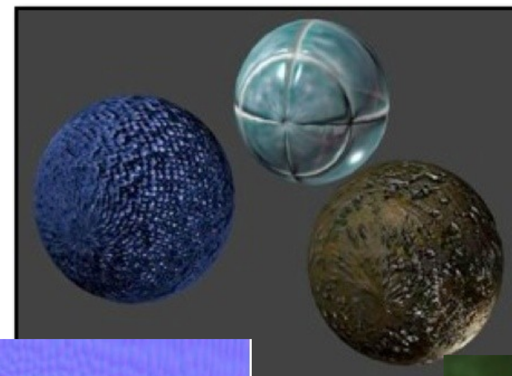
Scene → View → Projection → Perspective → Viewport

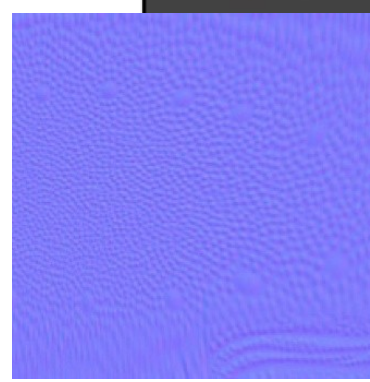
Should we do the math?

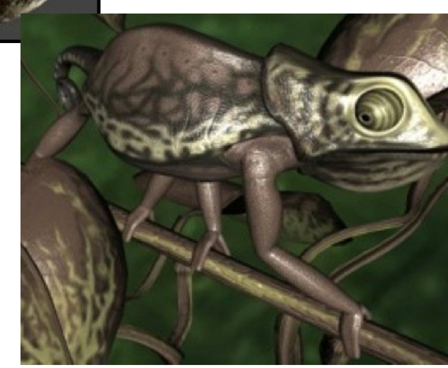
Bump maps

- Textures help with the color of the pixels to be drawn
- However, the resulting objects still look flat
- To improve this, one can store in a texture (bump map) normal variations, and use it for lighting computations while rendering
- This achieves a bumpy surface
- However, when bump mapped polygons are seen from a flat angle they show their flatness



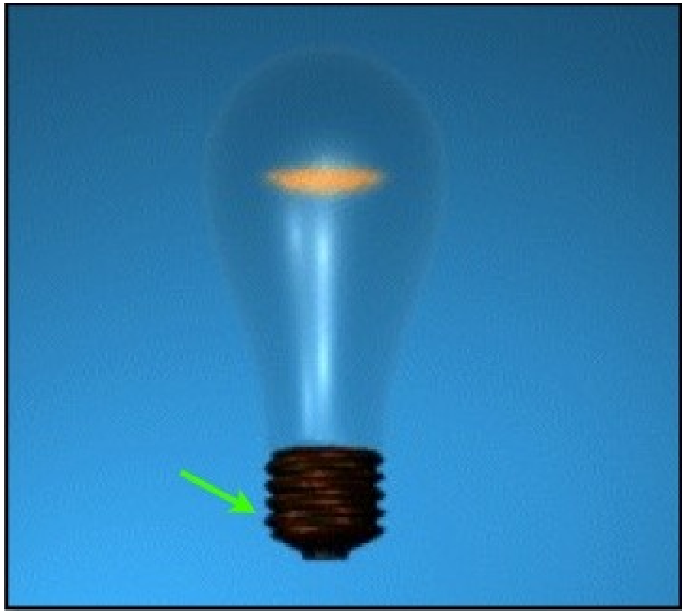


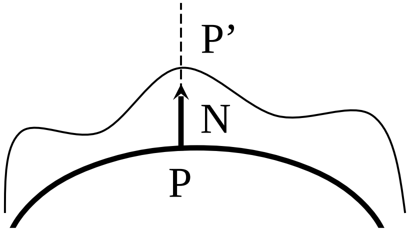




Displacement maps

- Bump maps do not modify geometry height, which does not look good from the profile
- A way to correct this is to interpret an additional black and white texture as displacement offsets along the normal
- This is called a displacement map
- Since the displacement map “modifies” the surface to add detail to it, usual lighting computations can be done in the result

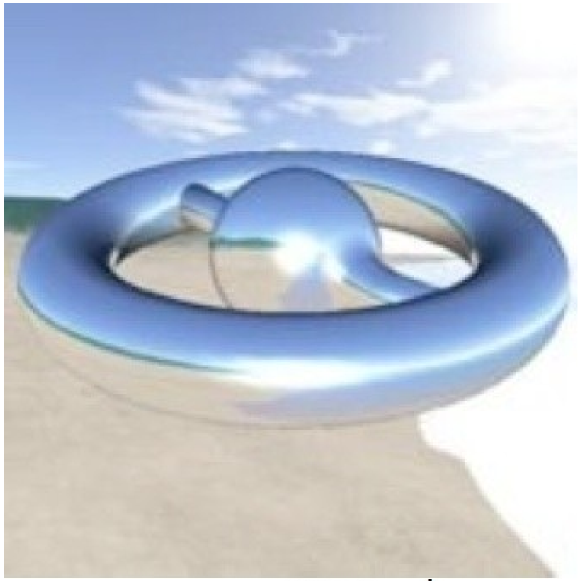


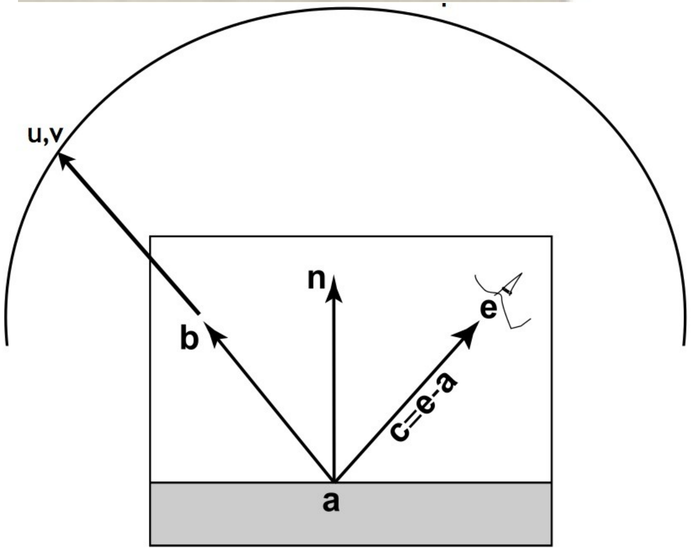


Surface + displacement

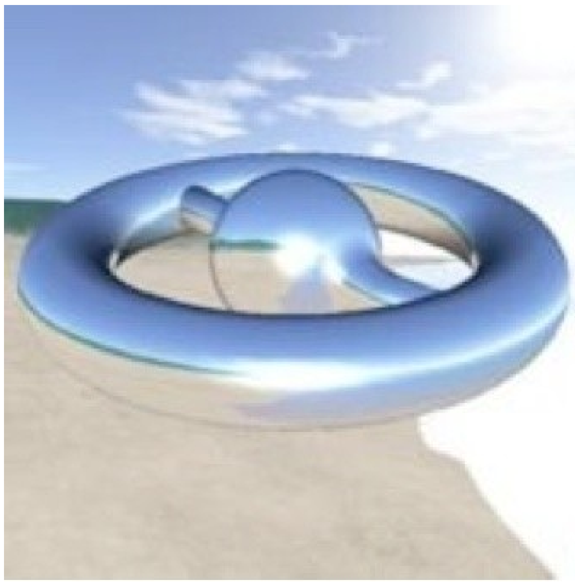
Environment maps

- There are many ways to use textures to obtain special effects in a picture
- Environment maps are used to simulate reflections on objects
- In this case, the world is surrounded by a closed surface having a texture
- The colour at the pixel to be rendered is looked up on the texture according to the reflection ray
-

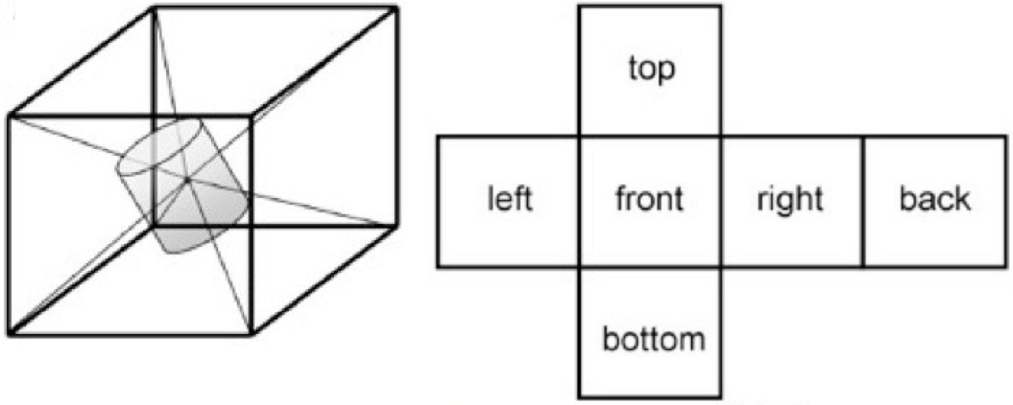




Environment maps

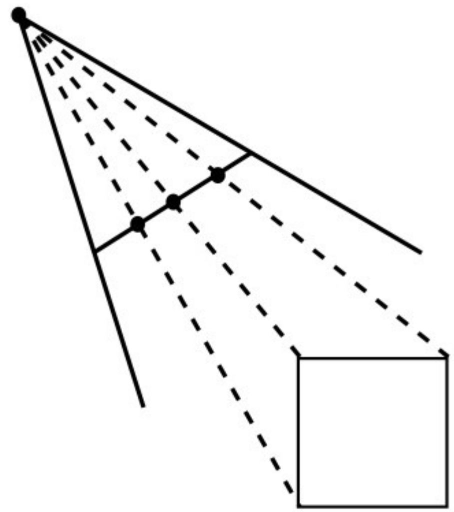
- There are two different ways of surrounding the world with a surface
- With a sphere: spherica maps
- 

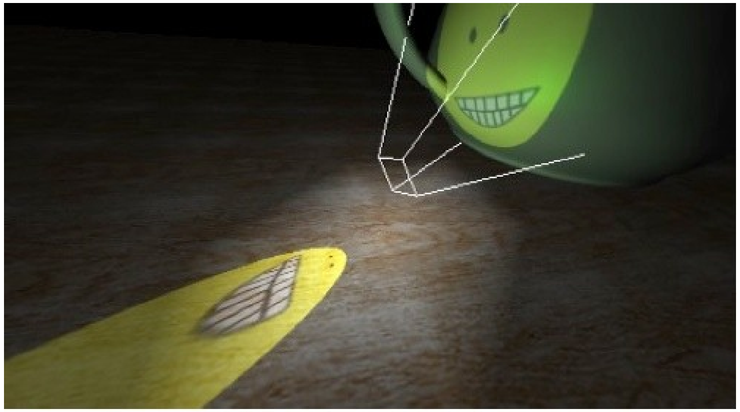
- With a cube: cube maps



Projective texture maps

- Projective texture maps work like beamer projection:
  – Texture coordinates of a vertex are mapped to texels projected from a given perspective
  – What this does is to add a perspective projection from a texture positioned in space
  – One needs to find out a mapping M_{d2t} from normalized device coordinates [-1,1] to texture coordinates [0,1].
  – M_s=scene transformation
    M_v=view projector transformation
    M_{pp}=persp. projection of projector
    h=homogeneous coords for perspective division

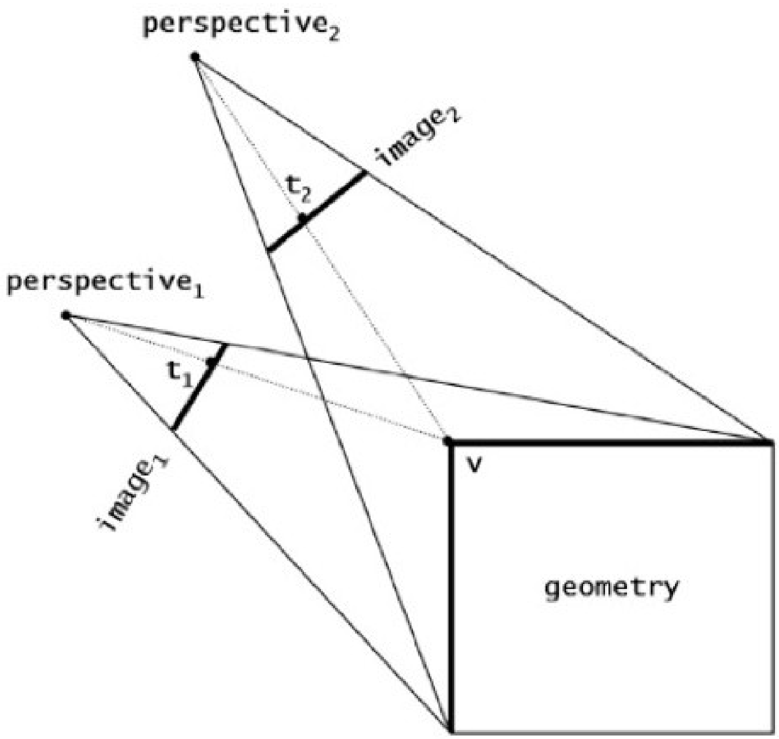




- Then $t=M_{d2t}(M_{pp}M_vM_sv)/h$, where $M_{d2t} = \begin{bmatrix} 0.5 & 0 & 0.5 \\ 0 & 0.5 & 0.5 \\ 0 & 0 & 1 \end{bmatrix}$

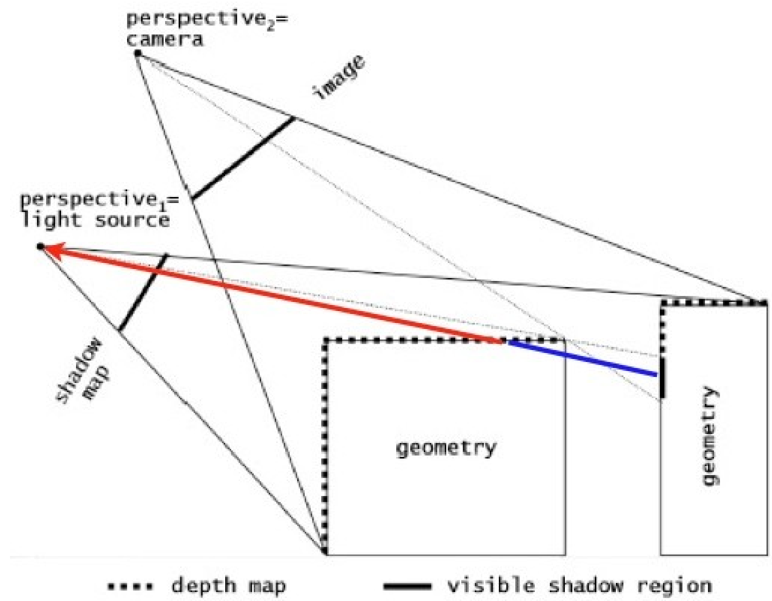
Projective texture maps

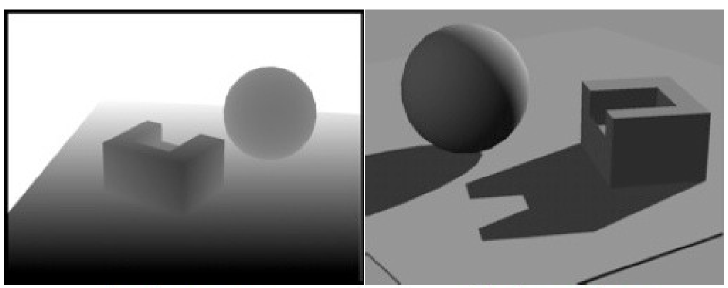
- One can use perspective texture maps to warp one perspective into another one:
  - Render image1 from perspective1
  - Compute projective texture coordinates for perspective1 and assign them to the vertices
  - Render scene from perspective2 from texture mapped from image1 and the computed projective textures coordinates
  - For texturing, this is not so interesting, because the result is almost the same as rendered from perspective2
  - However, image1 can be an arbitrary texture



Shadow maps

- One of them can for example contain shadow information to allow hard shadows in my environment
- In this case, you set perspective1 at the light source, and image1 contains z-buffer values of the scene from the light source
- When rendering the scene, at each pixel one is rendering one looks if its distance from the light source is smaller or bigger than the z-buffer from the light source
- In case it is bigger, then this point is in the shadow of something else



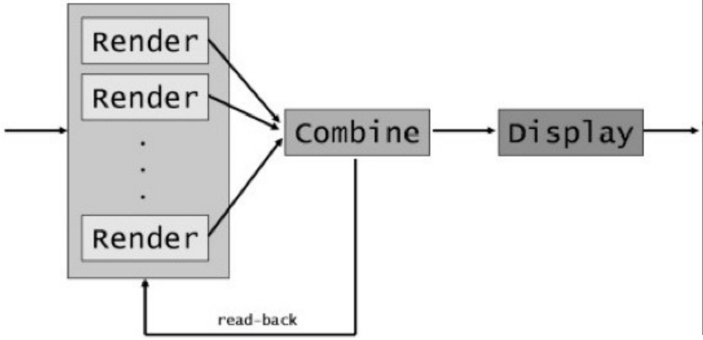


shadow map
1st pass

final rendering
2nd pass

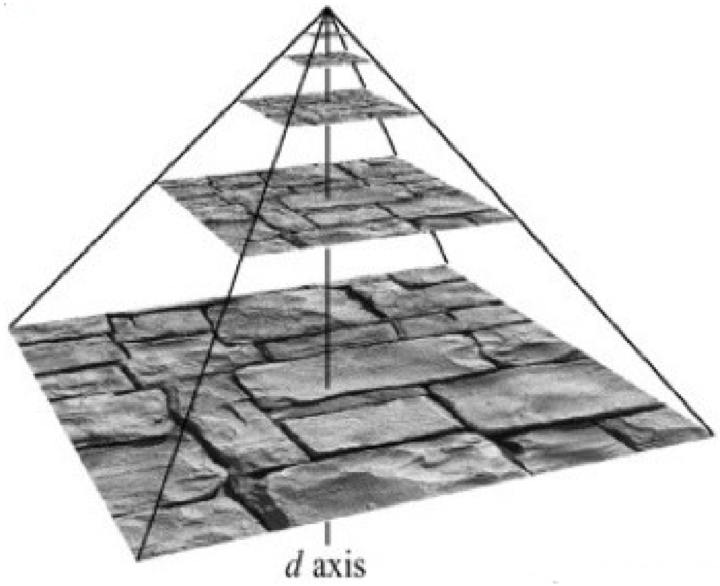
Multi-pass rendering

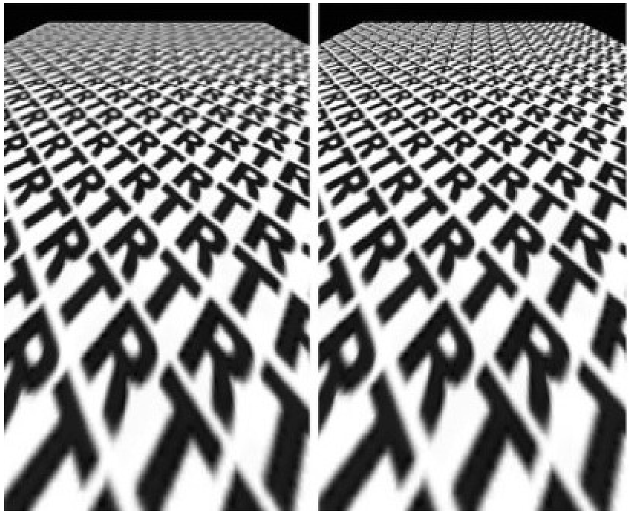
- To achieve more complex rendering effects, different texture rendering passes are rendered to a texture and not displayed
- This allows the layering of different effects, by blending the results of different rendering passes
- This is called multi-pass rendering



Mip-mapping

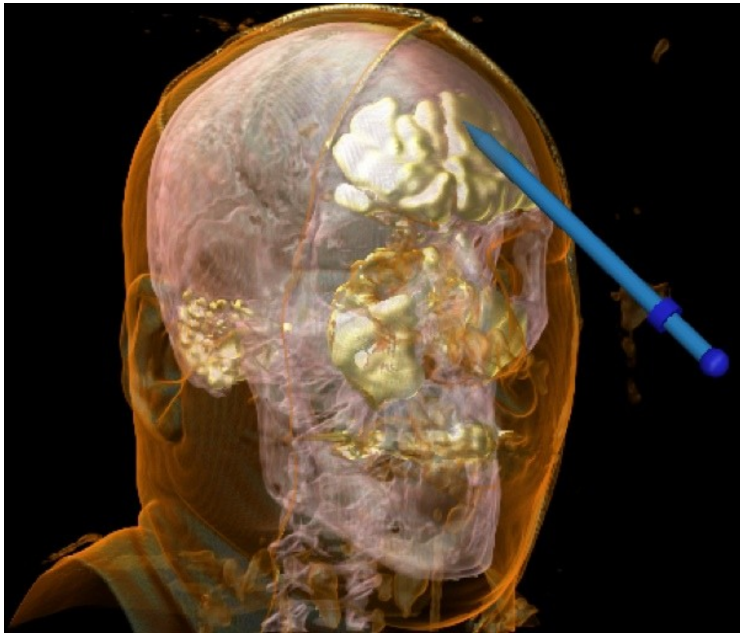
- The visual quality of the image depends also from texture resolution with respect to the potion of the screen occupied
- One can define lower resolution textures for scene parts that do not need so much detail
  - This is done by decreasing the resolution of the texture
- Transitions between different levels of detail have to be carefully computed, for example with bilinear interpolation

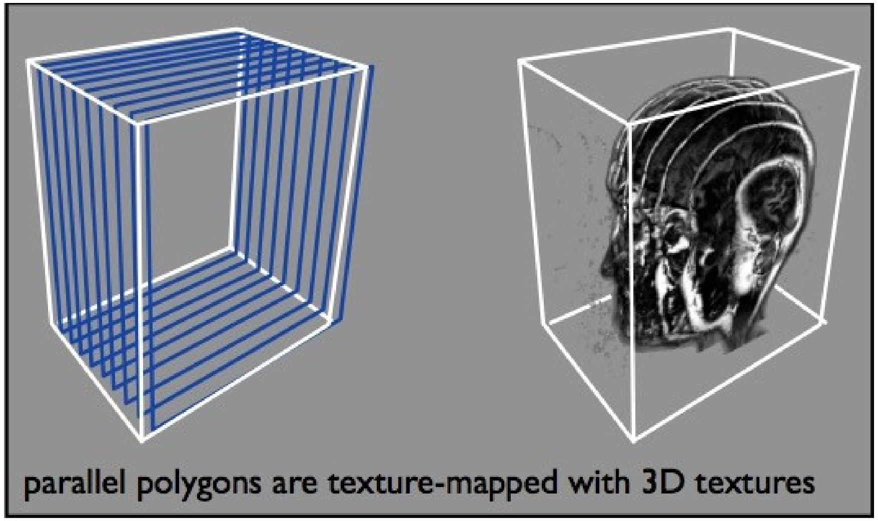


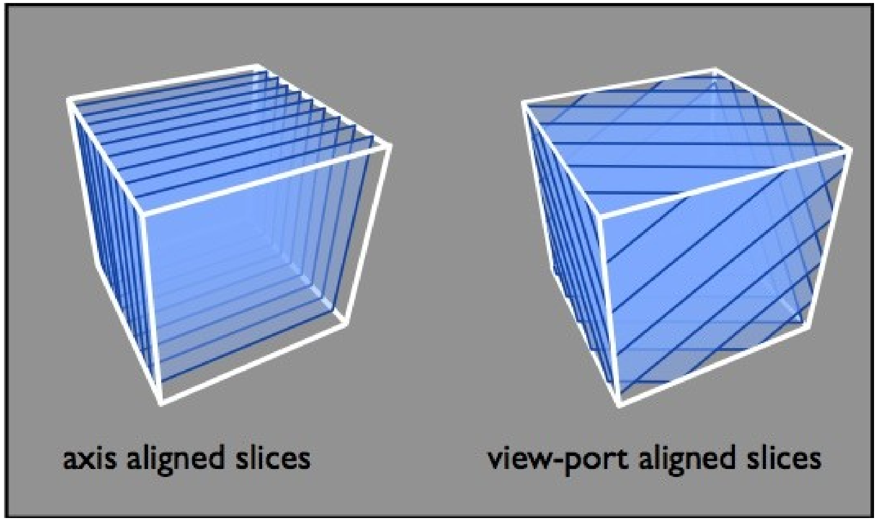


3D textures

- For volume data, colour can be given by looking up in a 3D texture
- But we will not go into detail with this







End

+++ Ende - The end - Finis - Fin - Fine +++ Ende - The end - Finis - Fin - Fine +++