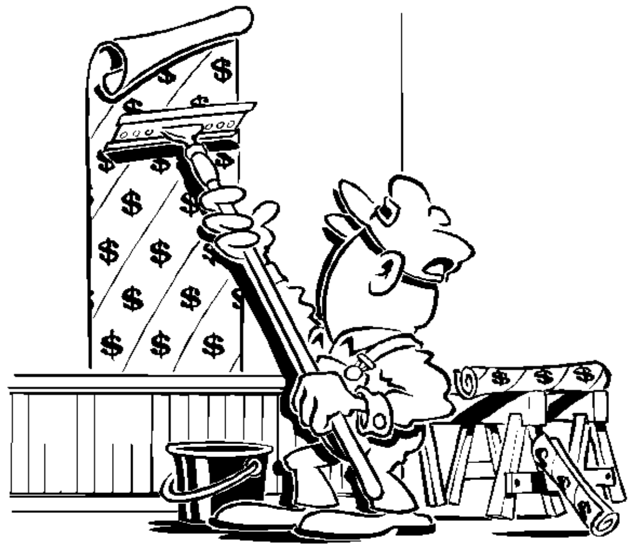
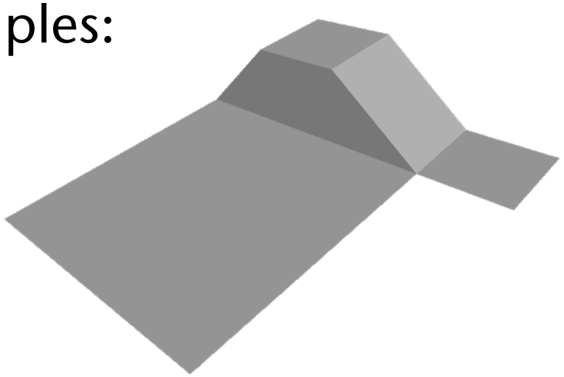
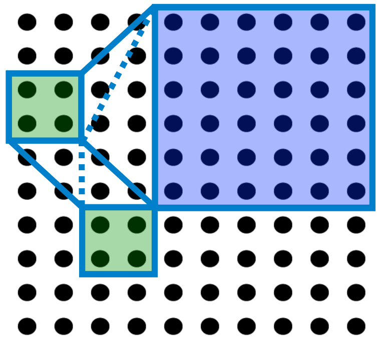
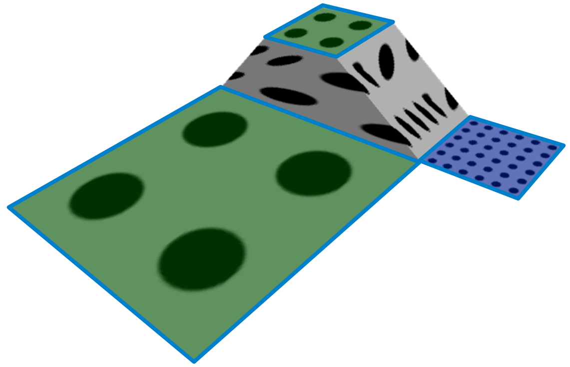
Advanced Computer Graphics
Advanced Texturing Methods

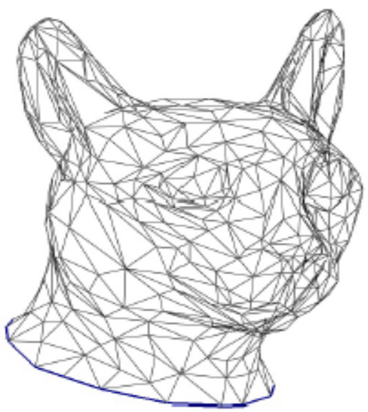


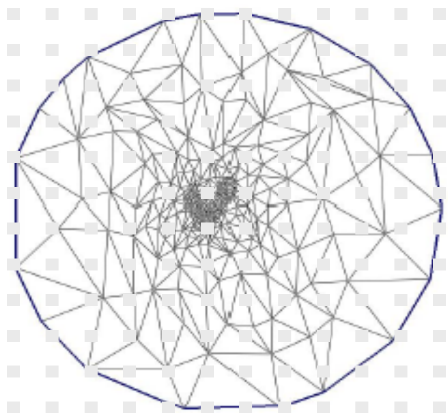
G. Zachmann
University of Bremen, Germany
cgvr.informatik.uni-bremen.de

Problems with (Simple) Parameterizations

- Distortions: size & form
- Consequence: relative over- or under-sampling
- Examples:

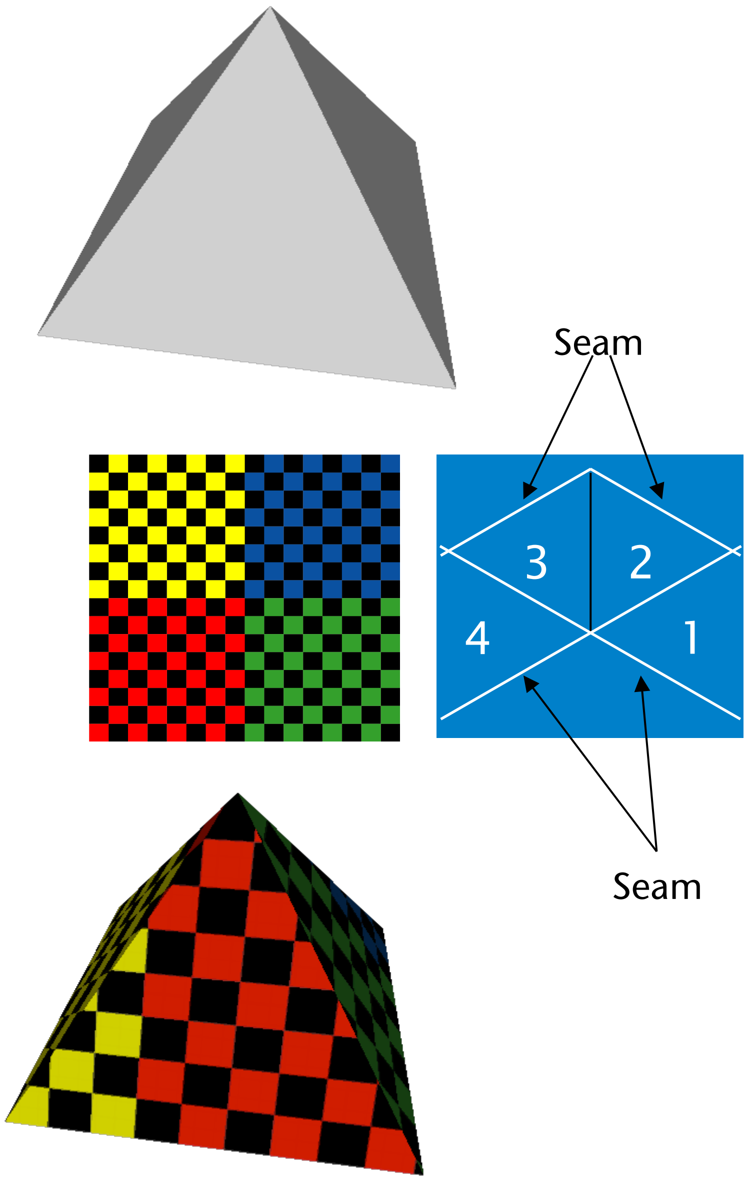

Mesh


Embedding

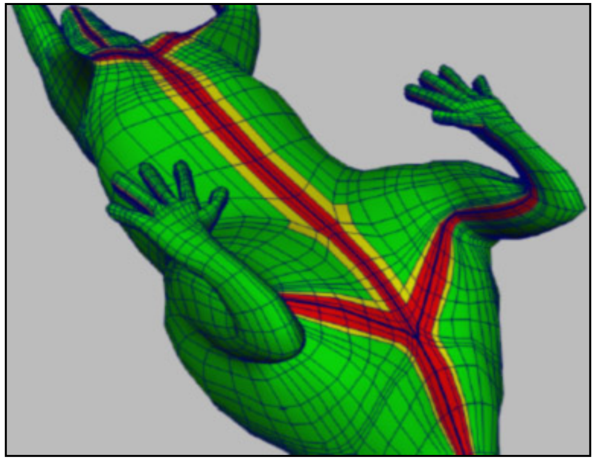

Distortion

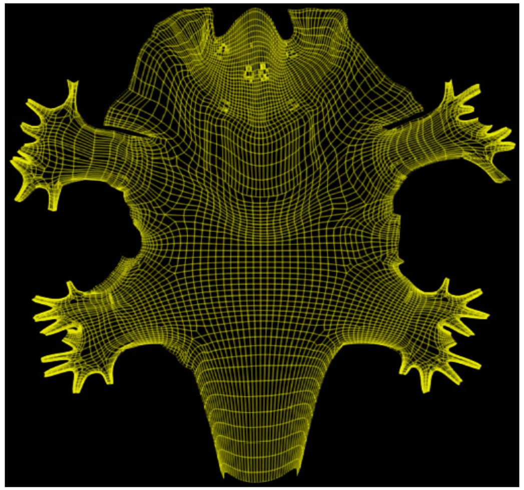
One Technique: Seams ("Nähte", Textursprünge)

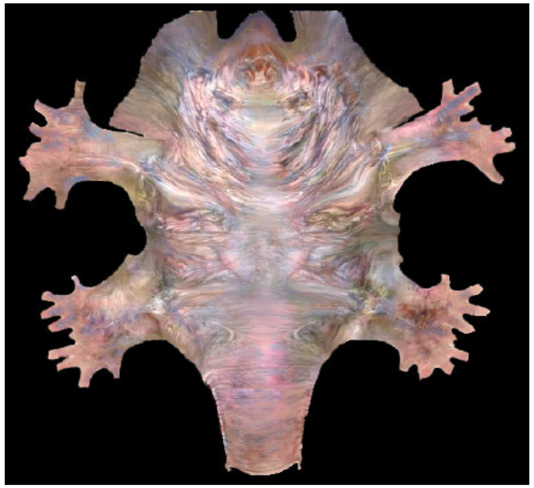
- Goal: minimize the distortion
- Idea: cutting up the mesh along certain edges
- Results in "double edges", also called *seams*
- Unavoidable with non-planar typology



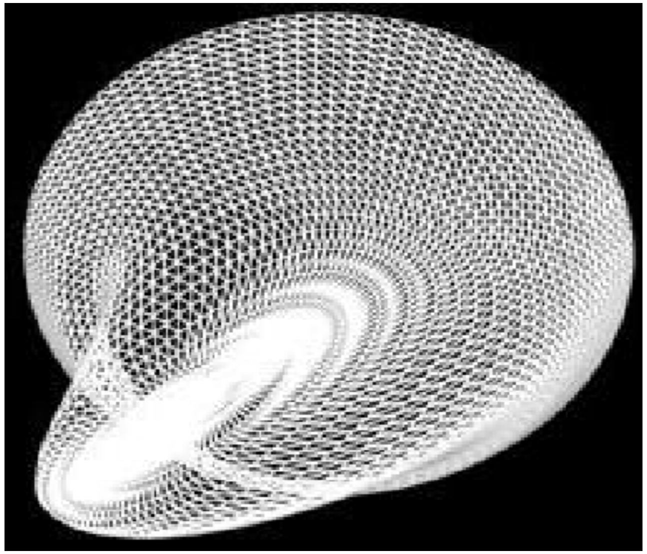
- Idea 1 [Piponi 2000]:
  - Cut the object along only one continuous edge
  - Effect: the resulting mesh is now topologically equivalent to a disc
  - Then embed this cut-open mesh into the 2D plane

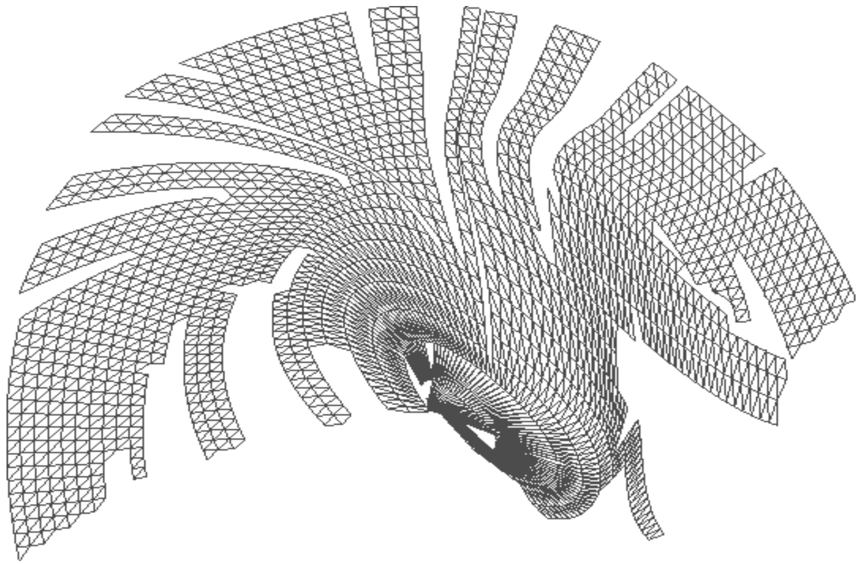




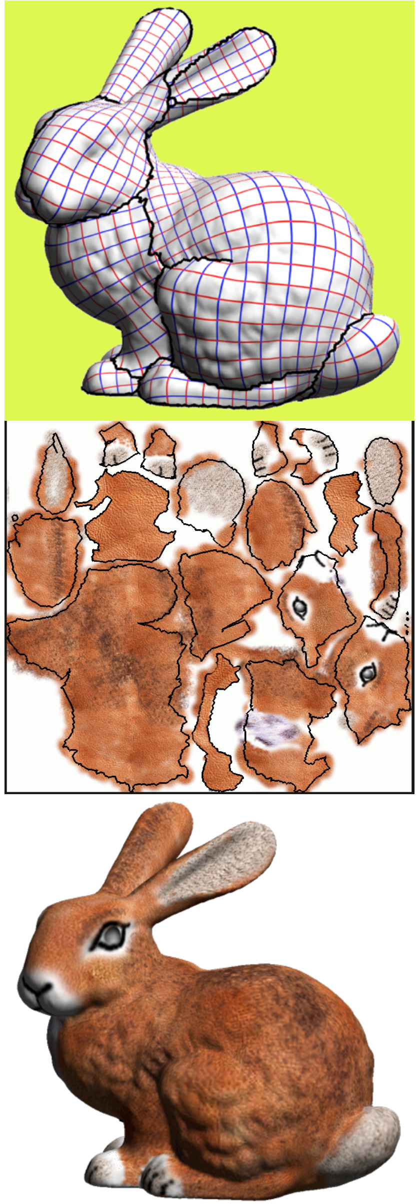


- Problems:
  - There are still distortions
  - Multiple incisions produce a severely frayed embedded grid

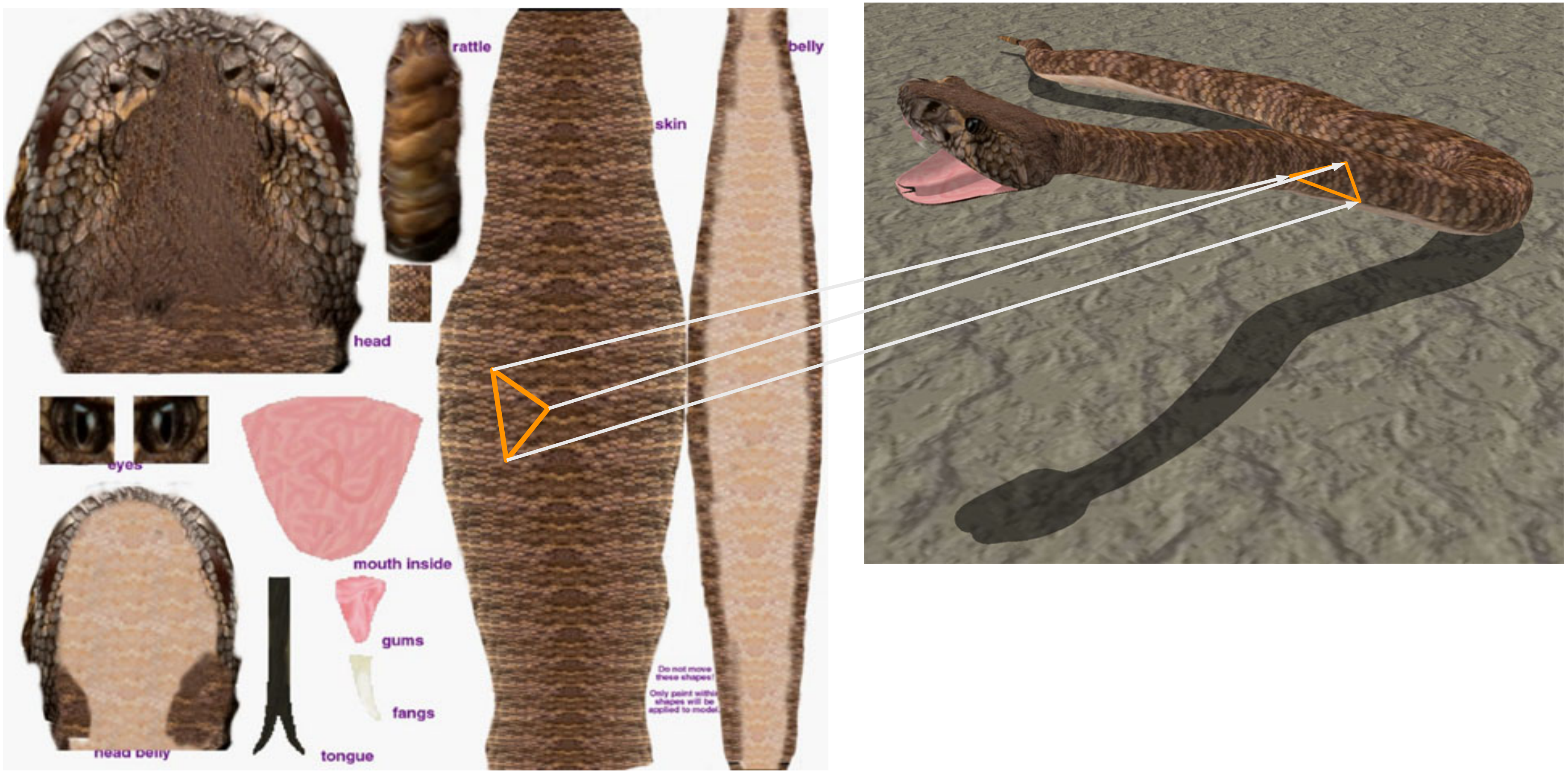


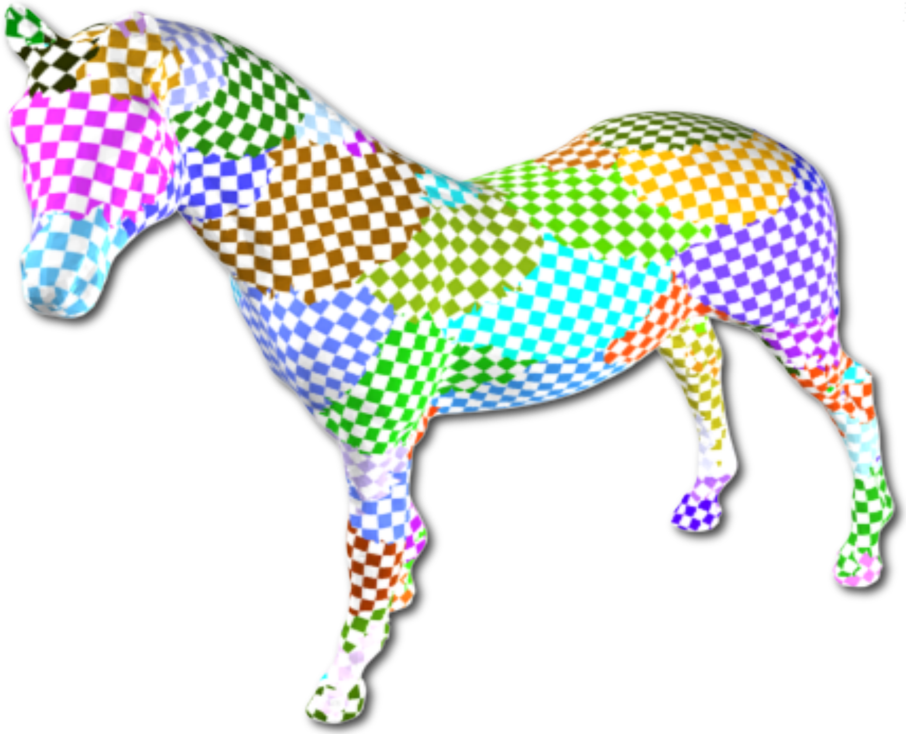


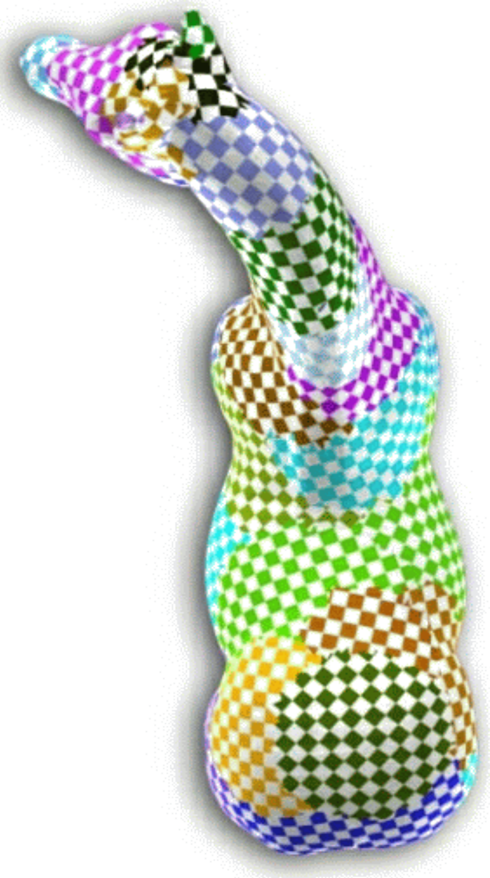
- Idea 2:
  - Cut the 3D surface in individual patches
  - Map = individual parameter domain in texture space for a single patch
  - Texture Atlas = set of these patches with their respective maps (= parameter domains)
- Statement of the problem:
  - Choose a compromise between seams and distortion
  - Hide the cuts in less visible areas
    - How do you do that automatically?
  - Determine a compact arrangement of texture patches (a so-called *packing problem*)



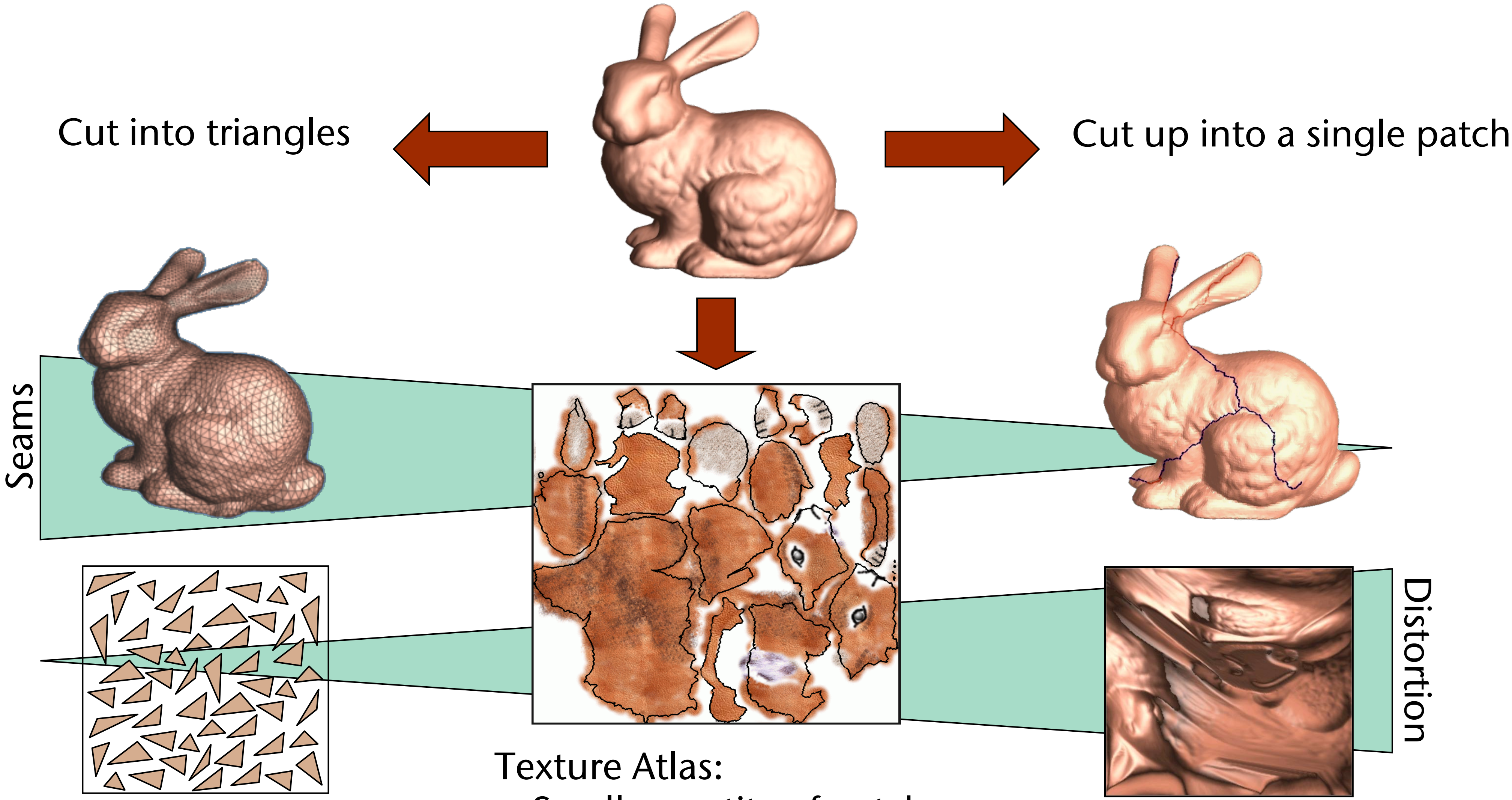
- Example:


rattle
belly
skin
head
eyes
mouth inside
gums
fangs
tongue
head belly





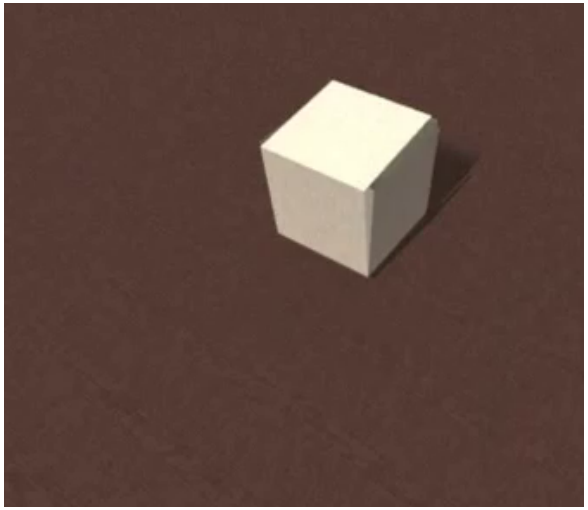
Distortion or Seams?



Texture Atlas:
- Small quantity of patches
- Short and hidden seams

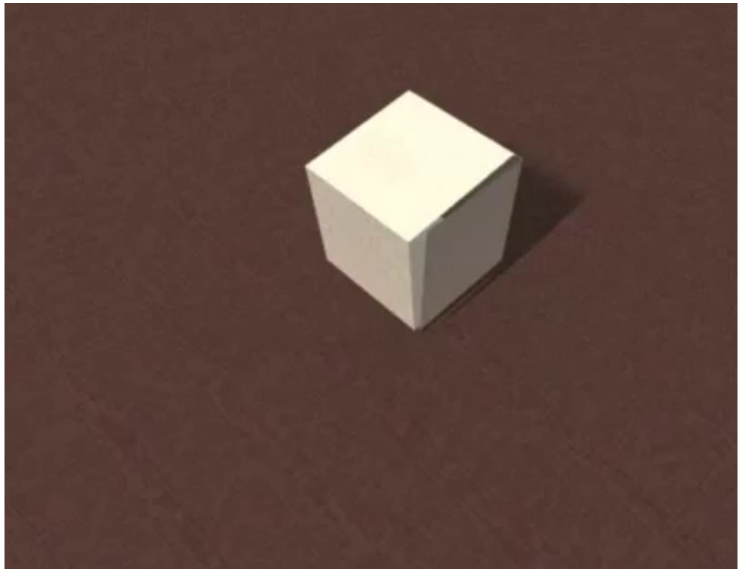
Digression: A Geometric Brain-Teaser

- A cube can be unfolded into a cross:



Katie Park / unfoldit.org

- Into what other forms can a cube be unfolded, too?



Katie Park / unfoldit.org

- Side note: the (unfolded) cube can be folded into a parallelogram

- BTW: all platonic solids except for the dodecahedron can be folded into a parallelogram in this way …

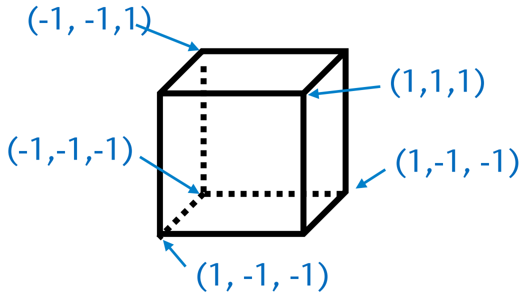
Cube Maps

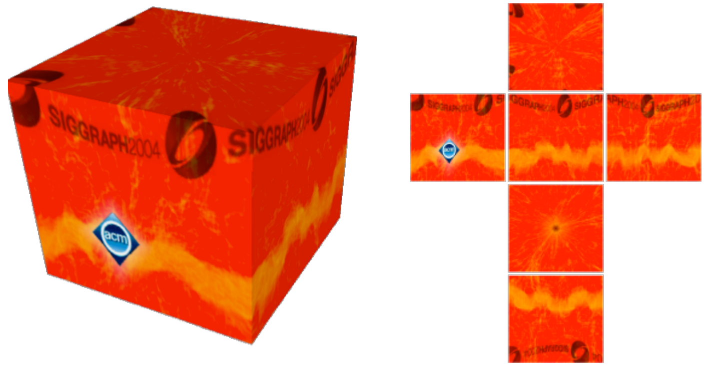
[Greene '86, Voorhies '94]

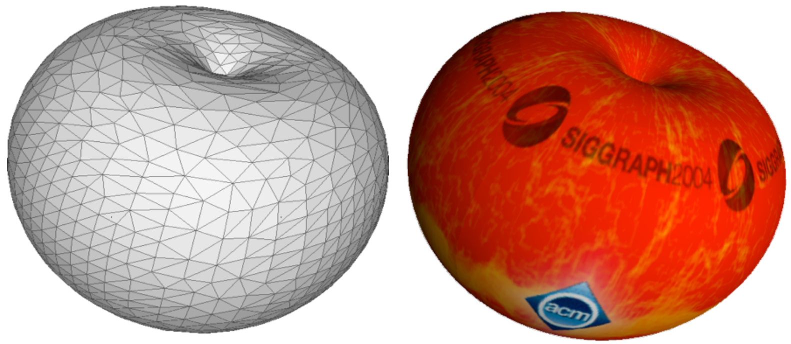
- Parameter domain Ω = unit cube:
  - Six quadratic texture bitmaps
  - 3D texture coordinates in OpenGL:

```
glTexCoord3f( s, t, r );
glVertex3f( x, y, z );
```

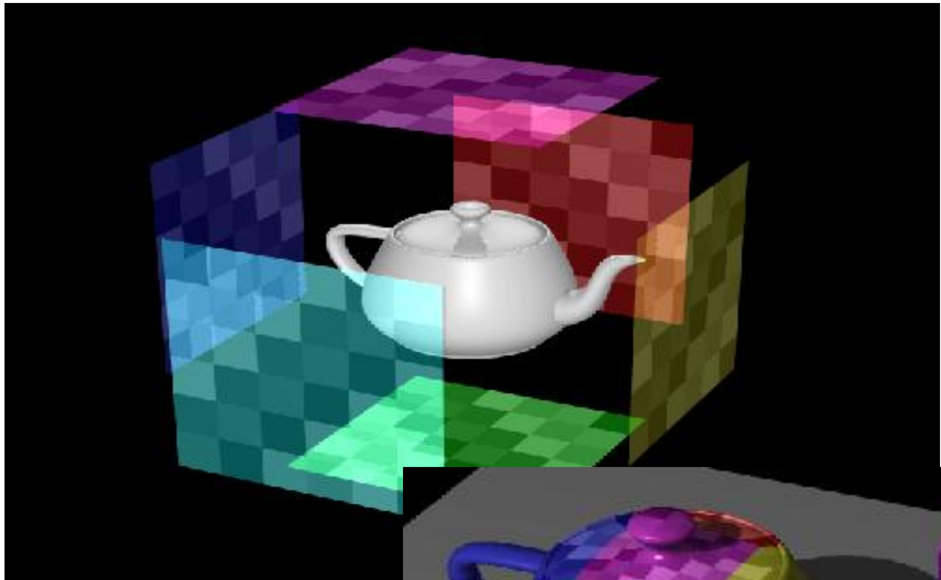
  - Largest component of (*s*,*t*,*r*) determines the map, intersection point determines (*u*,*v*) within the map
- Rasterization of cube maps:
  1. Interpolation of (*s*,*t*,*r*) in 3D
  2. Projection onto the cube ⟶ (u,v)
  3. Texture look-up in 2D
- Pro: relatively uniform, OpenGL support
- Con: one needs 6 images

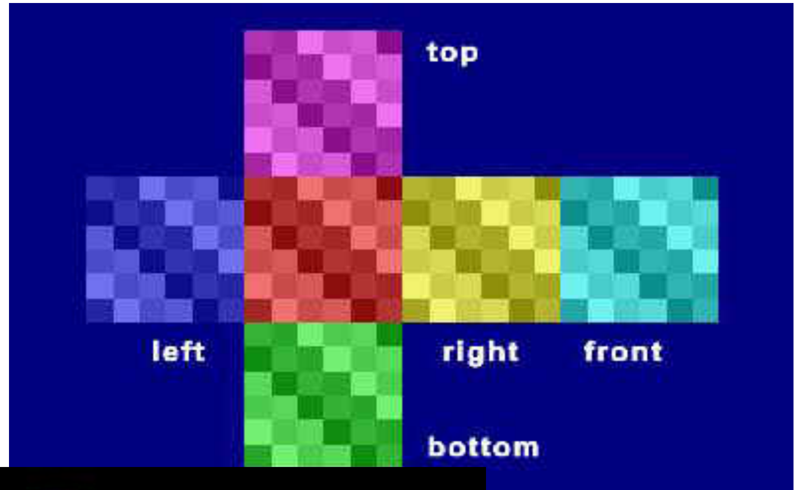


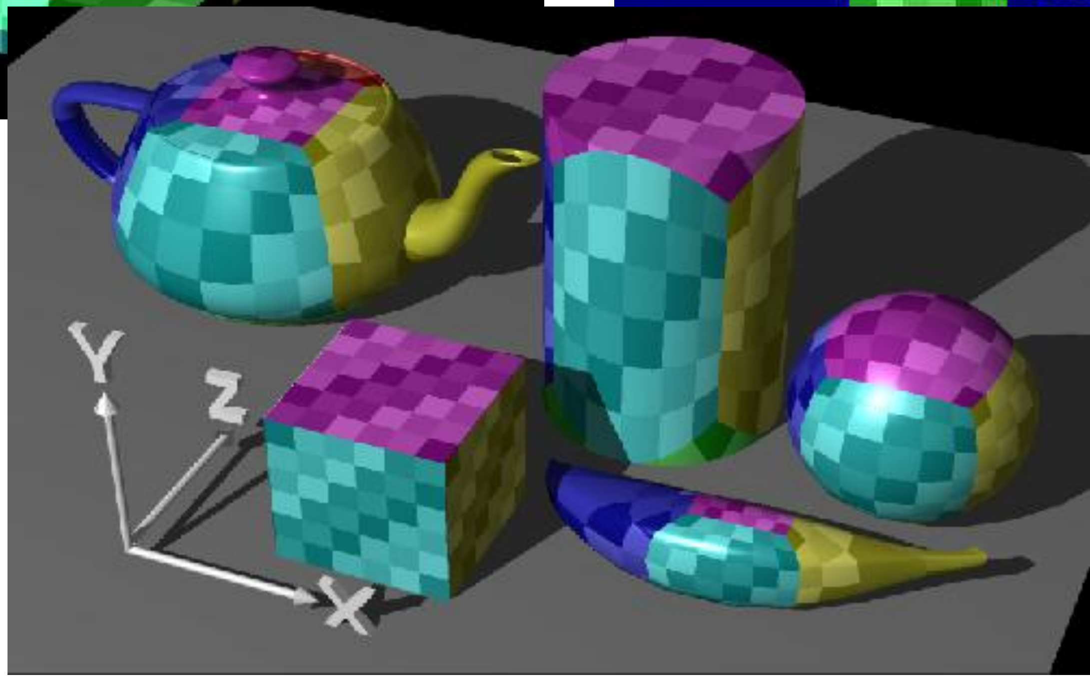




Examples







Cube Maps in OpenGL

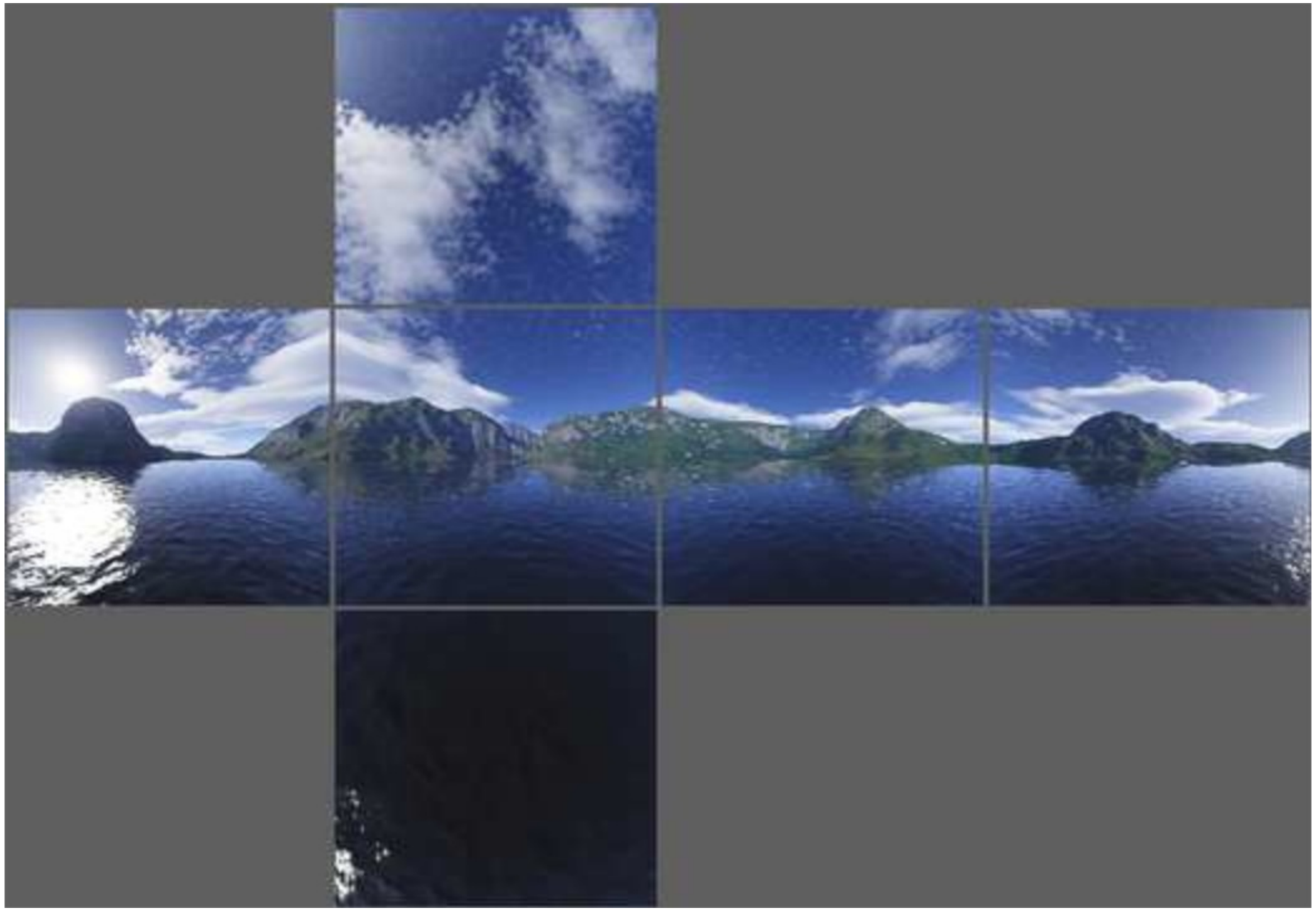
```
glGenTextures( 1, &textureID );
glBindTexture( GL_TEXTURE_CUBE_MAP, textureID );
glTexImage2D( GL_TEXTURE_CUBE_MAP_POSITIVE_X, 0, GL_RGBA8, width, height,
              0, GL_RGB, GL_UNSIGNED_BYTE, pixels_face0 );
... Load the texture of the other cube faces
glTexParameteri( GL_TEXTURE_CUBE_MAP,
                 GL_TEXTURE_WRAP_S, GL_CLAMP_TO_EDGE );
... Set more texture parameters, like filtering

glEnable( GL_TEXTURE_CUBE_MAP );
glBindTexture( GL_TEXTURE_CUBE_MAP, textureID );
glBegin( GL_... );
glTexCoord3f( s, t, r );
glVertex3f( ... );
...
```

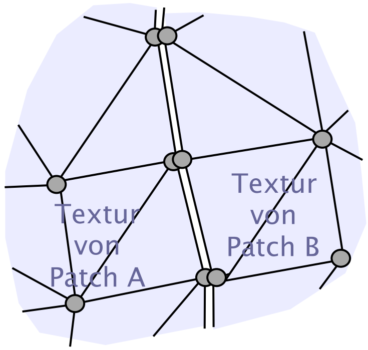
Analog:
`GL_TEXTURE_MAG_FILTER`,
`GL_TEXTURE_WRAP_T`, `etc.` …

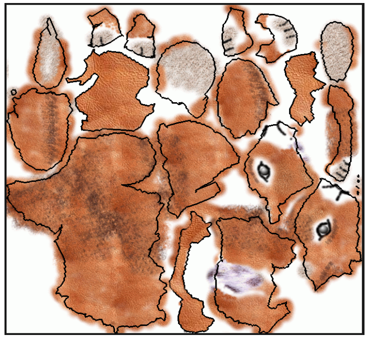
Just like with all other vertex attributes in OpenGL: first send all attributes, then the coordinates

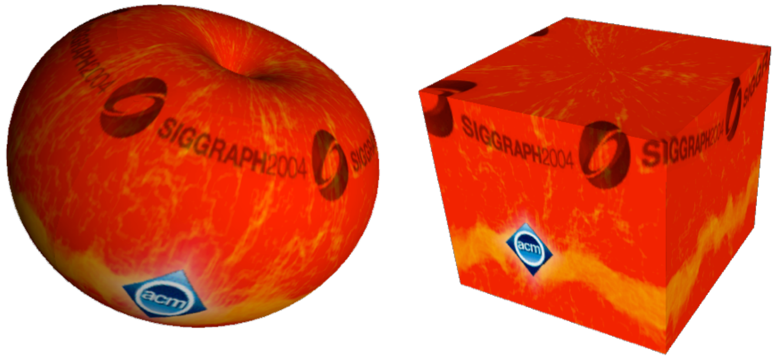
- Example cube map for a sky box:



Texture Atlas vs. Cube Map

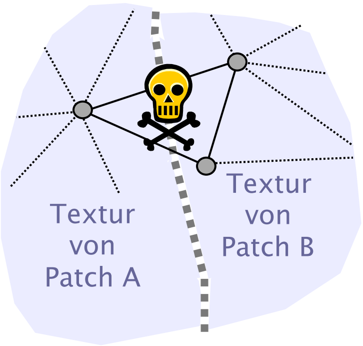


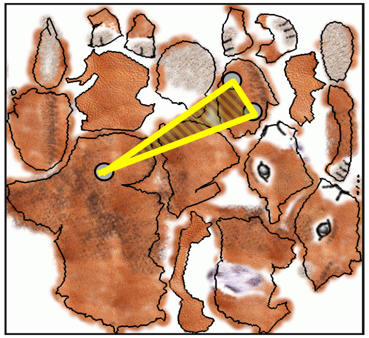


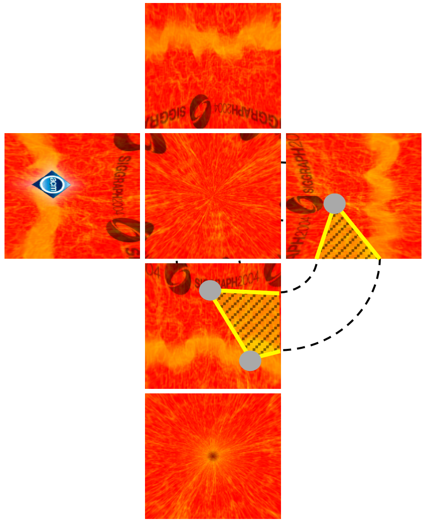


- Seams
- No seams

Texture Atlas vs. Cube Map

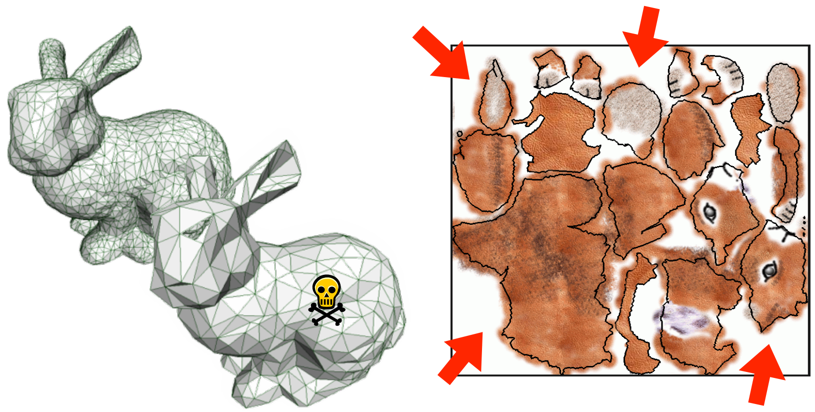


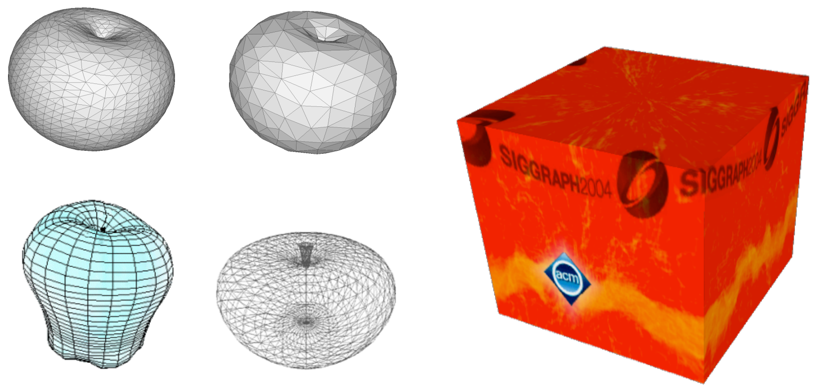




- Seams
- Triangles can lie only within the patches
- MIP-mapping is difficult

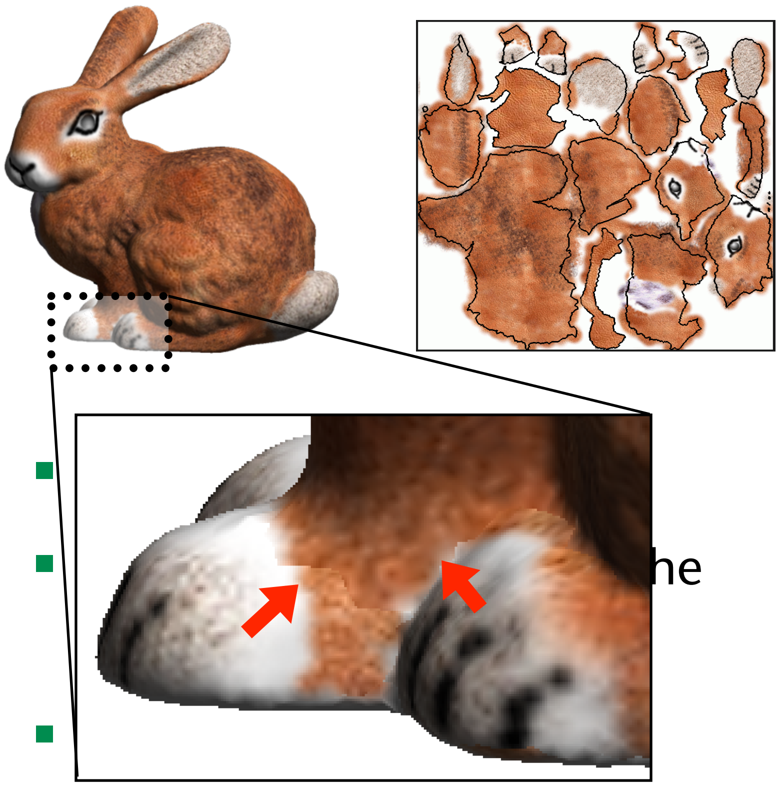
- No seams
- Triangles can lie in multiple patches
- MIP-mapping is okay

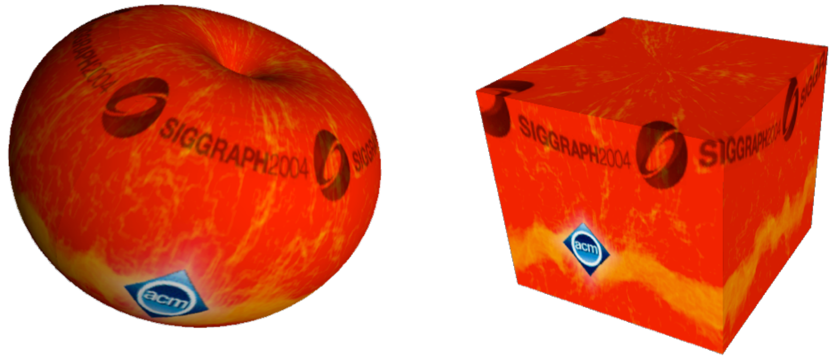




- Seams
- Triangles may lie within the patches
- MIP-mapping is difficult
- Only valid for a specific mesh
- Texels are wasted

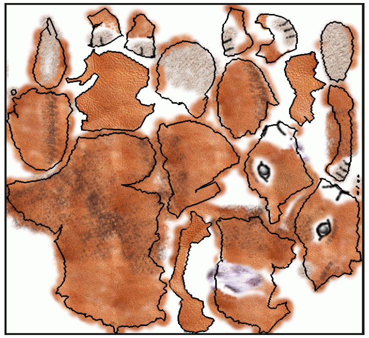
- No seams
- Triangles can lie in multiple patches
- MIP-mapping is okay
- Valid for many meshes
- All texels are used

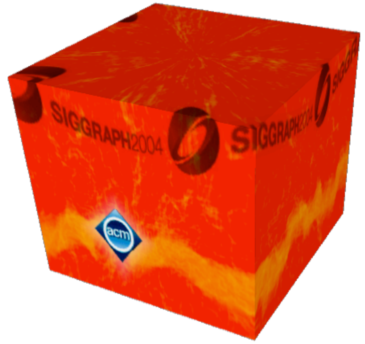




-
- ...he
-
- Only applies to a specific mesh
- Disappearing texels
- Sampling artifacts at the edges of the patches

- No seams
- Triangles can lie in multiple "patches"
- MIP mapping okay
- Applicable to many meshes
- All texels are used
- No edges, no sampling artifacts





- Seams
- Triangles may lie within the patches
- MIP mapping is difficult
- Only applies to a specific mesh
- Disappearing texels
- Sampling artifacts at the edges of the patches

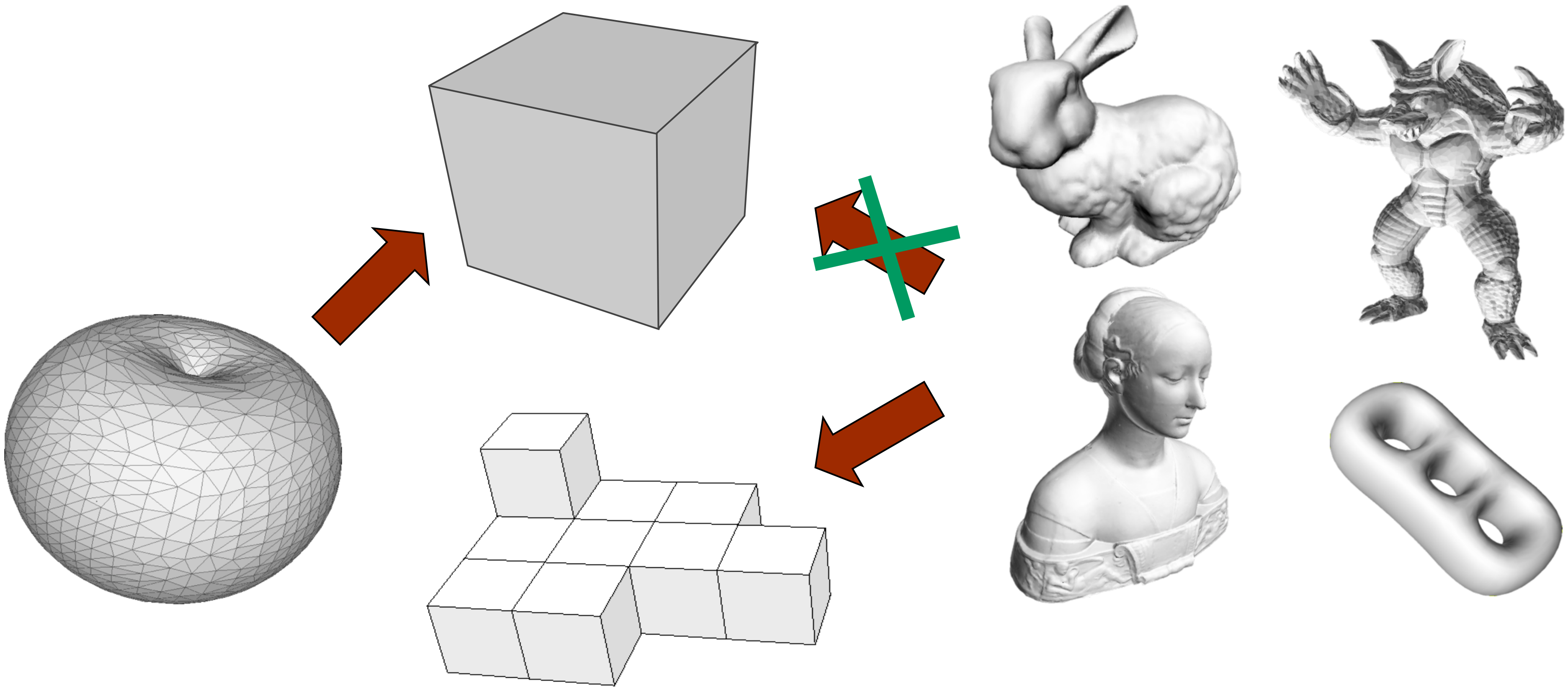
Works for any mesh

- No seams
- Triangles can lie in multiple "patches"
- MIP mapping okay
- Applicable to many meshes
- All texels are used
- No edges, no sampling artifacts

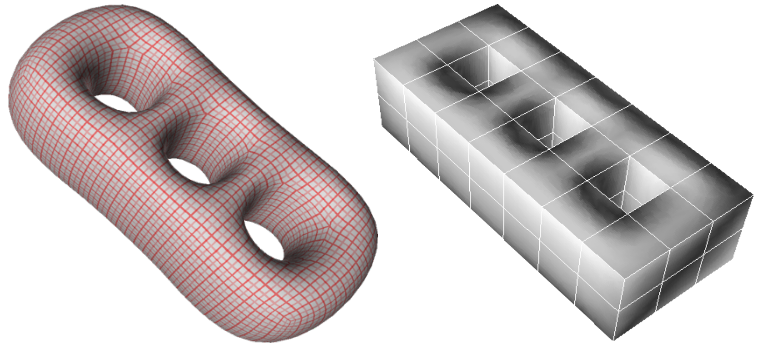
Only for "spheres"

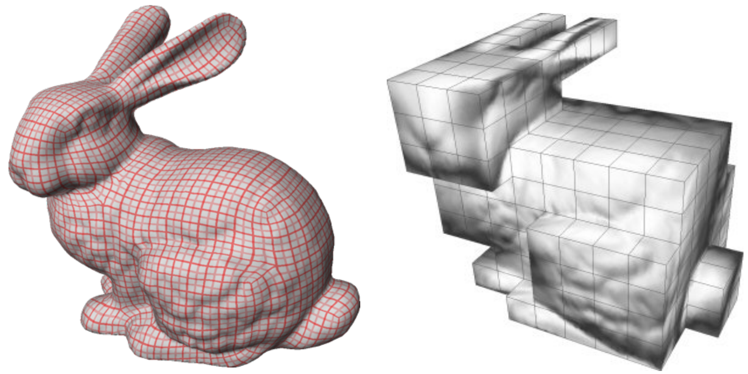
Polycube Maps

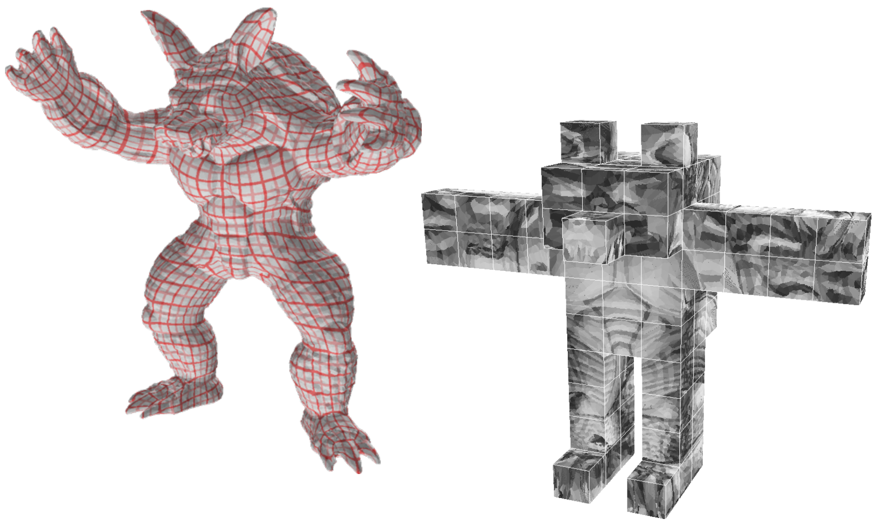
- Use many cube maps instead of an individual cube ⟶ polycube map
- Adapted to geometry and topology

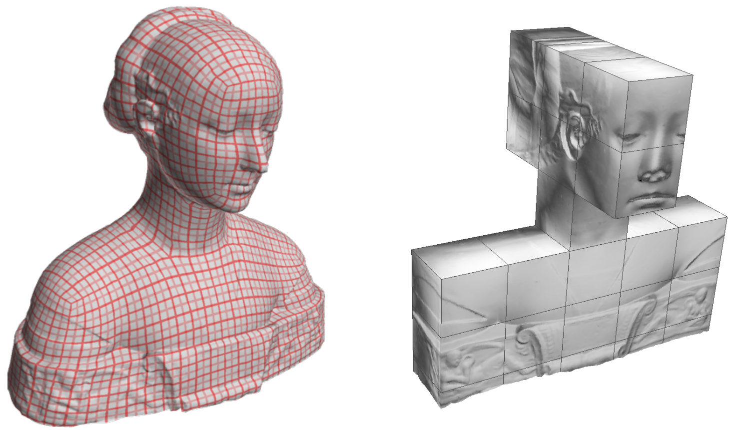


Examples



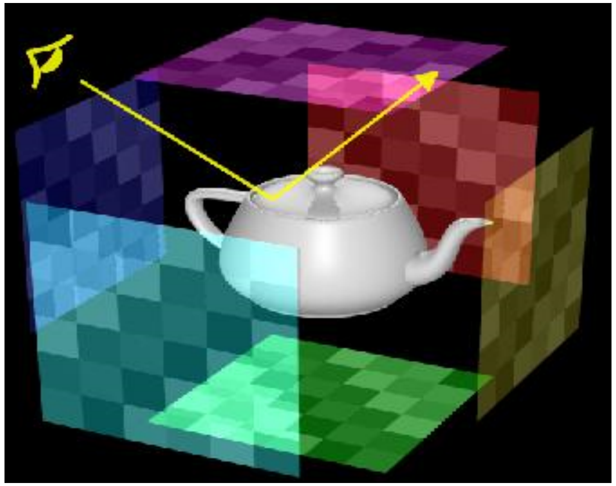




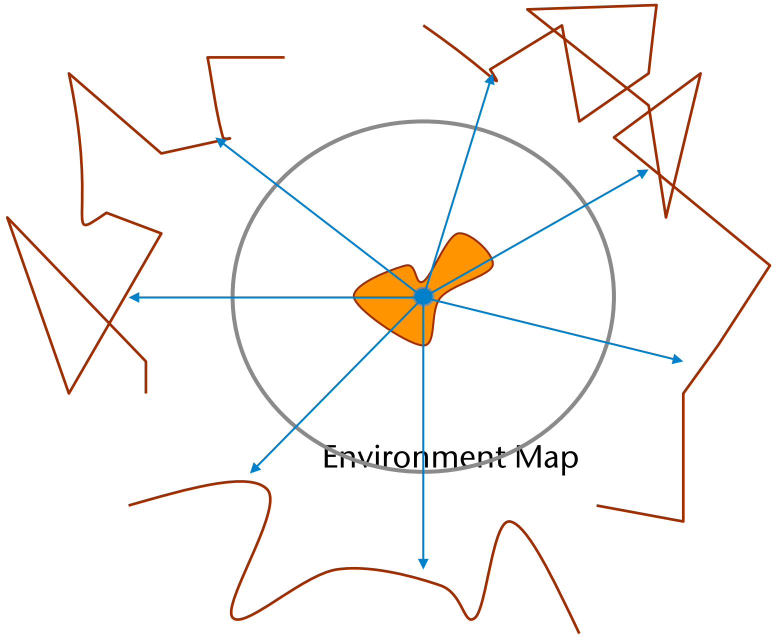


Environment Mapping

- With very reflective objects, one would like to see the surrounding environment reflected in the object
- Ray tracing can do this, but not the simple Phong shading model
- The idea of environment mapping:
  - "Photograph" the environment in a texture
  - Save this in a so-called environment map
  - Use the reflection vector (from the ray) as an index in the texture
  - A.k.a. reflection mapping



- For every spatial direction, the environment map saves the color of the light that reaches a specific point
- Only correct for one position
- No longer correct if the environment changes



Historical Examples of Applications

Lance Williams, Siggraph 1985

Flight of the Navigator in 1986;
first feature film to use the technique

Terminator 2: Judgment Day - 1991
most visible appearance — Industrial Light + Magic