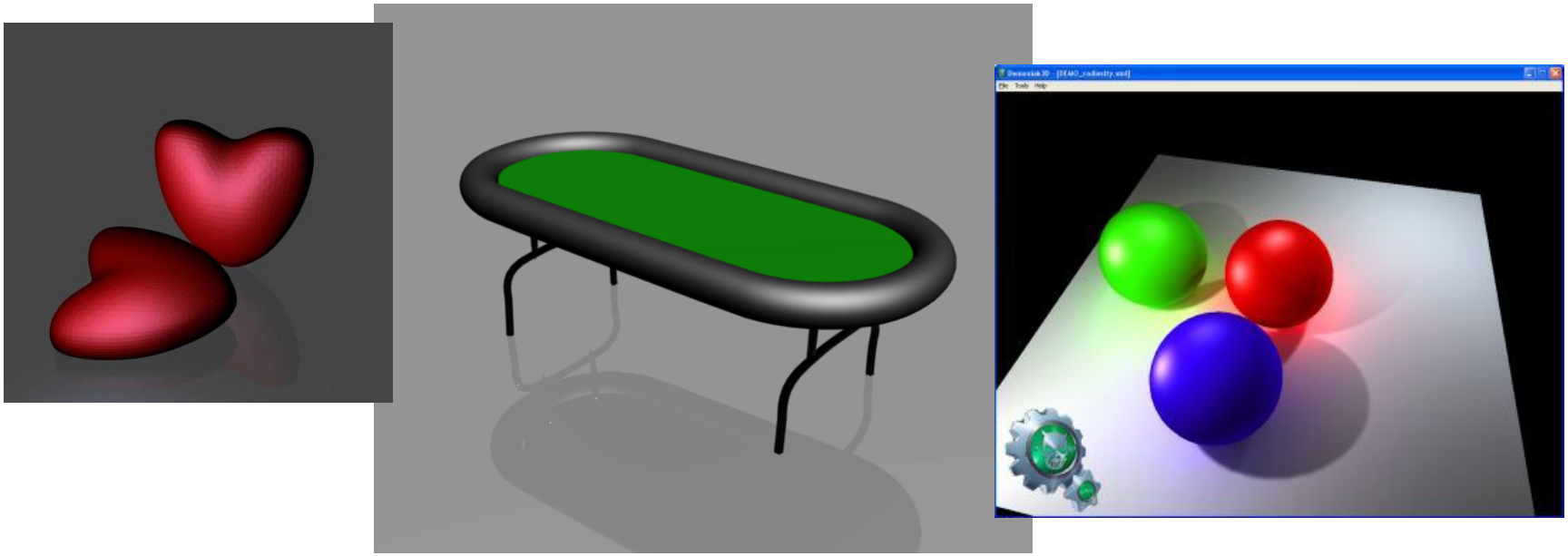
Design of Human Interface Game Software

– Textures

Texture and Surface Detail

- Providing color to objects
  - Solid colors: through rasterization
  - Lighting and shading: through lighting and shading models

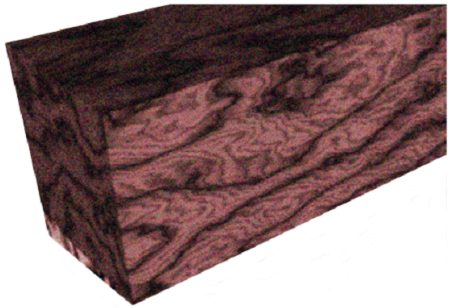


Texture and Surface Detail

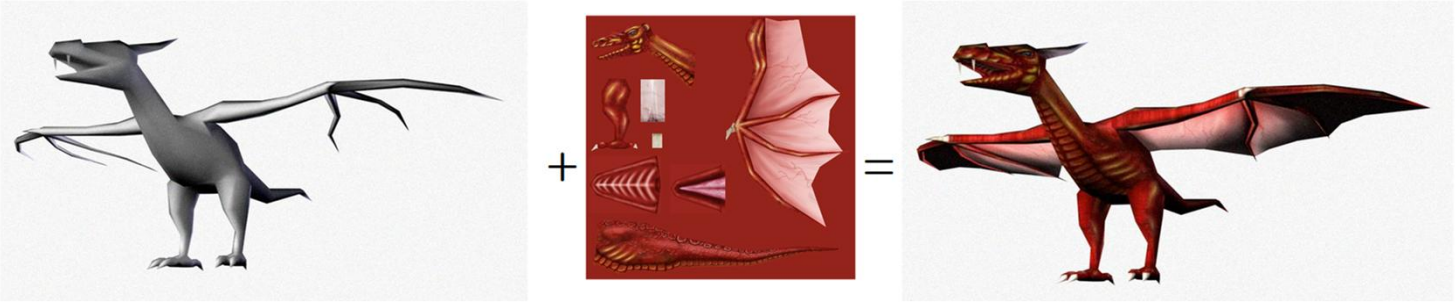
- Adding realism through surface detail
  - Natural surfaces: stone, wood, grass, gravel
  - Printing and painting: labels, newspapers
  - Clothing and fabric: woven and printed patterns



Texture Mapping



- Texture Mapping
  - Adding surface detail, color, or surface texture to a 3D model using images
  - For color variation in interior of polygons
- Texture or Texture Map
  - A 2D image to be mapped to the surface of a 3D object

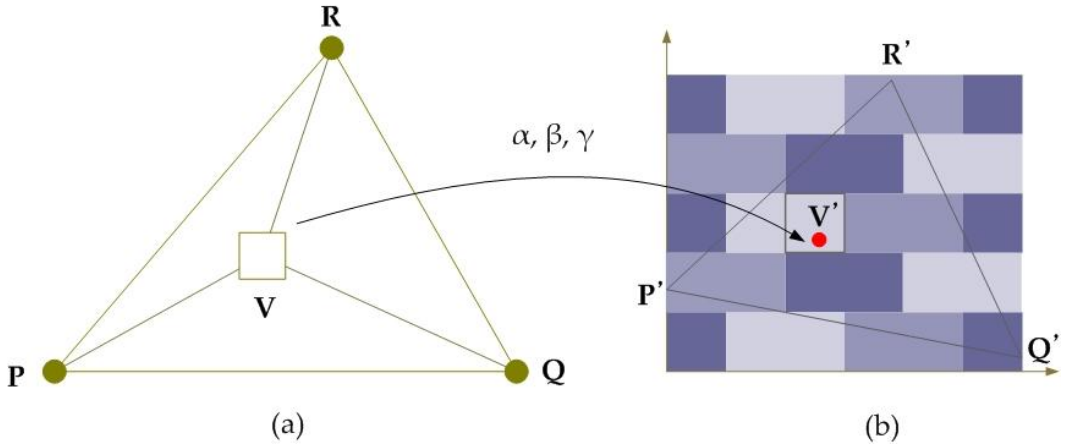


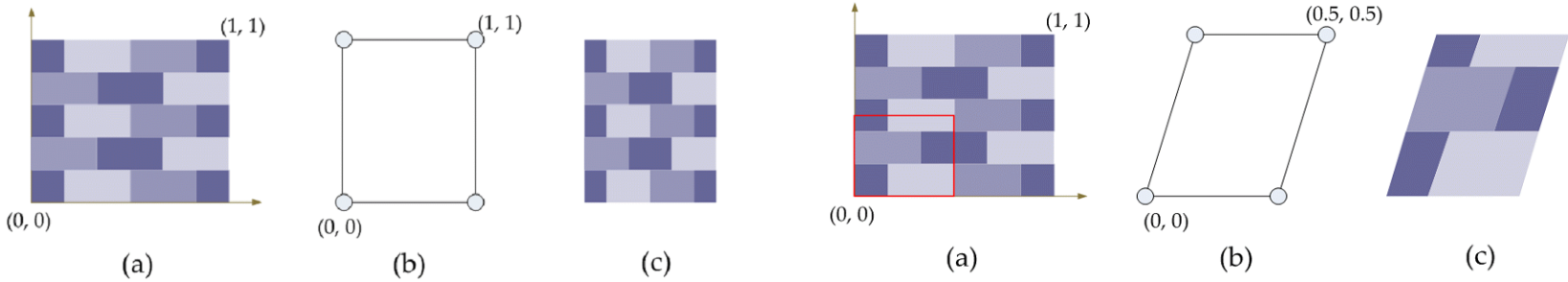
Texture Mapping

- Compute which **texel** corresponds to a given surface point

$$V = \alpha P + \beta Q + \gamma R$$

$$V' = \alpha P' + \beta Q' + \gamma R'$$





Texturing in OpenGL

- Initialization
  - Create a new texture object
  - Provide the associated texture image
- For each drawing
  - Enable texture mapping
  - Draw the textured polygons
    - Identify which texture is to be used
    - Specify texture coordinates with vertices
  - Disable texture mapping
    - When returning to normal drawing mode

Creating Texture Objects

- glGenTextures (GLsizei *n*, GLuint **textureIDs*);
  - Returns *n* currently unused texture IDs in *textureIDs*
  - Each texture ID is an integer greater than 0
- glBindTexture (GLenum *target*, Gluint *textureID*);
  - *target* is GL_TEXTURE_1D, GL_TEXTURE_2D, or GL_TEXTURE_3D
  - if *textureID* is being used for the first time a new texture object is created and assigned the ID = *textureID*
  - if *textureID* has been used before, the texture object with ID = *textureID* becomes active

Specifying a 2D Texture Object

- glTexImage2D (GLenum *target*, GLint *level*, GLint *internalformat*, GLsizei *width*, GLsizei *height*, GLint *border*, GLenum *format*, GLenum *type*, const GLVoid **texels*);
  - Ex) glTexImage2D (GL_TEXTURE_2D, 0, GL_RGBA, 128, 128, 0, GL_RGBA, GL_UNSIGNED_BYTE, image);
  - *format* and *type* used to specify the way the *texels* are stored
  - *internalFormat* specifies how OpenGL should store the data internally
  - *width* and *height* have to be powers of 2; you can use gluScaleImage( ) to scale

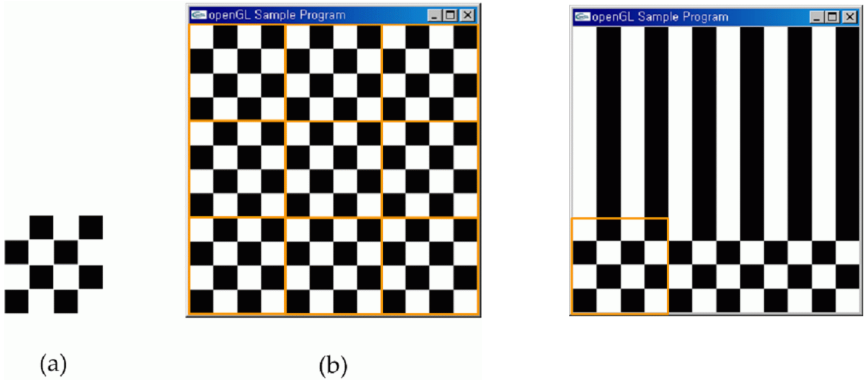
Specifying How Texture is Applied

- glTexEnv{if} (GLenum *target*, GLenum *pname*, {GLint|GLfloat} *value*);
  - *target* is GL_TEXTURE_ENV
  - *pname* is GL_TEXTURE_ENV_MODE
  - *value*
    - GL_MODULATE: mix with lighting
    - GL_REPLACE: just paint the texture color
  - Whether the texture color is combined with existing object color after lighting (modulation) or is just painted on (replacement)

Specifying How Texture is Applied

- glTexParameter{if} (GLenum *target*, GLenum *pname*, TYPE *param*)
- *target* can be: GL_TEXTURE_1D, GL_TEXTURE_2D, ...

pname	param
GL_TEXTURE_WRAP_S	GL_CLAMP, GL_REPEAT
GL_TEXTURE_WRAP_T	GL_CLAMP, GL_REPEAT
GL_TEXTURE_MAG_FILTER	GL_NEAREST, GL_LINEAR
GL_TEXTURE_MIN_FILTER	GL_NEAREST, GL_LINEAR



Enable the Texture

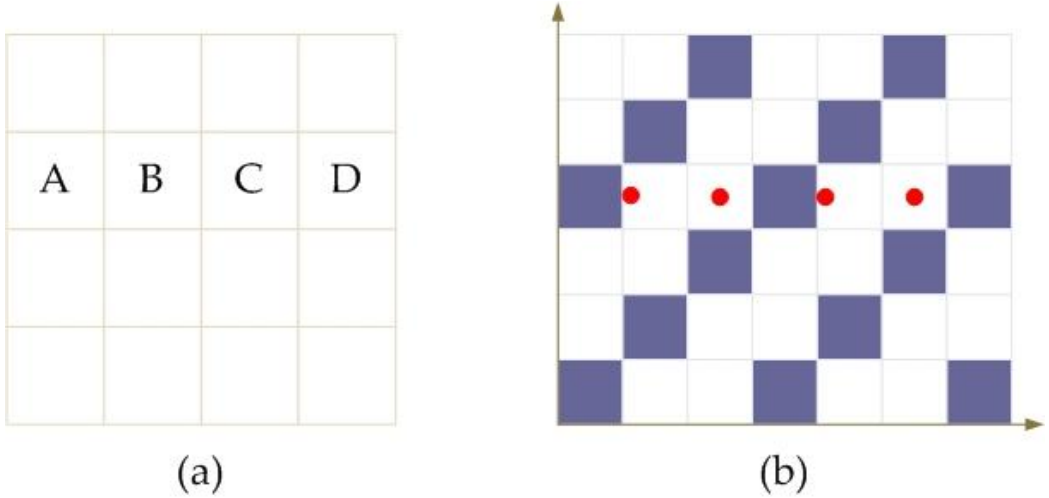
- glEnable (GL_TEXTURE_2D)
  - Enable 2D texturing
- glTexCoord2f(GL_FLOAT *s*, GL_FLOAT *t*)
  - Specify texture coordinates per vertex (just as normals, color, etc)
  - Texture coordinates always vary from 0 to 1
- glDisable (GL_TEXTURE_2D)
  - Disable 2D texturing

Putting it all together

- In initialization:
  glGenTextures(...);
  glBindTexture(...);
  glTexParameteri(...); glTexParameteri(...); ...
  glTexImage2D(...);
  glEnable(GL_TEXTURE_2D);
- In display:
  glBindTexture(...); // Activate the texture defined in initialization
  glBegin(GL_TRIANGLES);
  glTexCoord2f(...); glVertex3f(...);
  glTexCoord2f(...); glVertex3f(...);
  glTexCoord2f(...); glVertex3f(...);
  glEnd( );

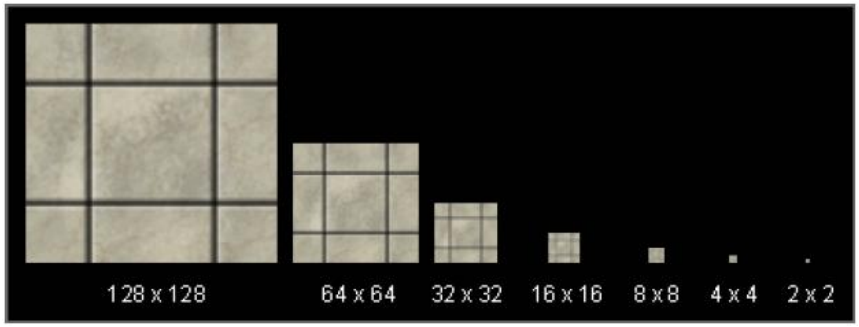
Aliasing

- What if one screen pixel overlaps many texture pixels?
  - A jagged appearance

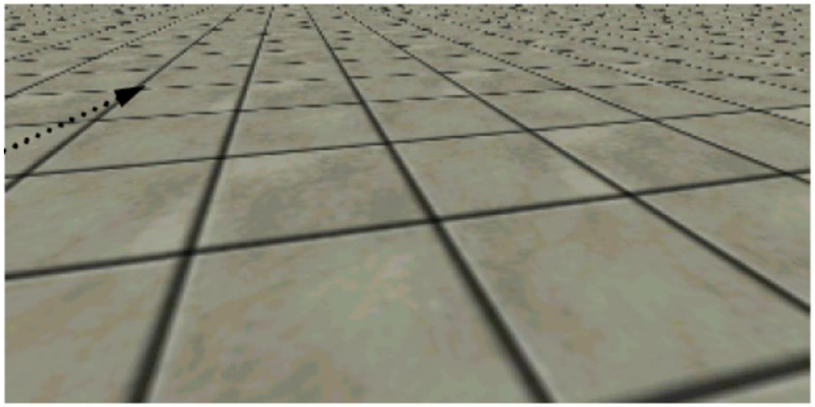


Mip-Mapping

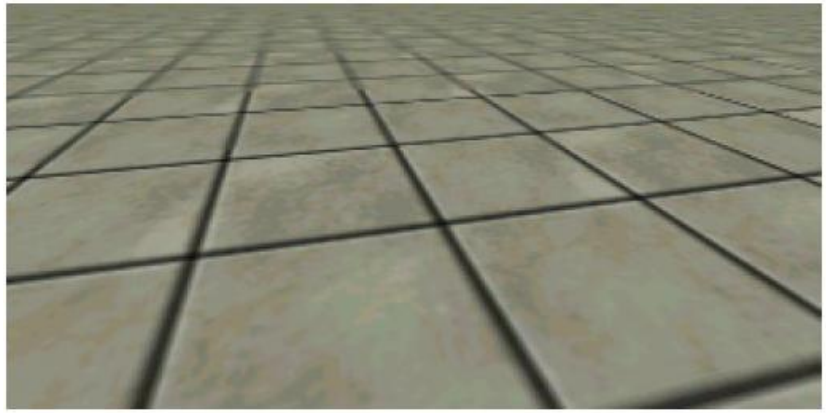
- Precompute averages and build hierarchy based on powers of 2



- To render: OpenGL gets appropriate level in the MIP-Map

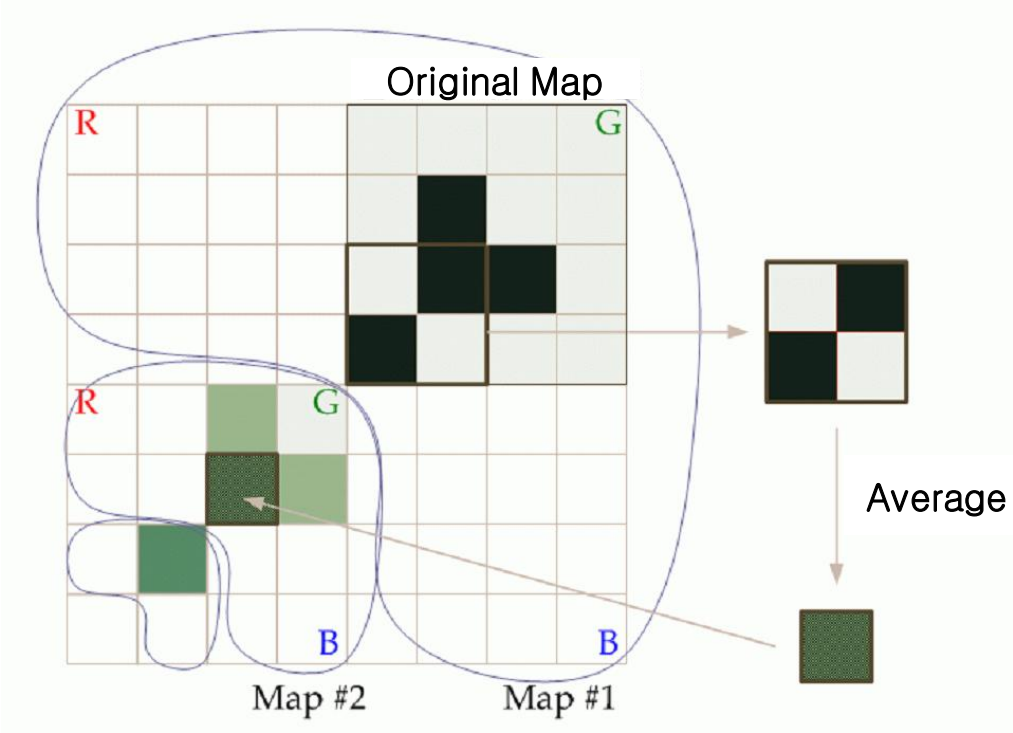


Without MIP-mapping



With MIP-mapping

Mip-Mapping

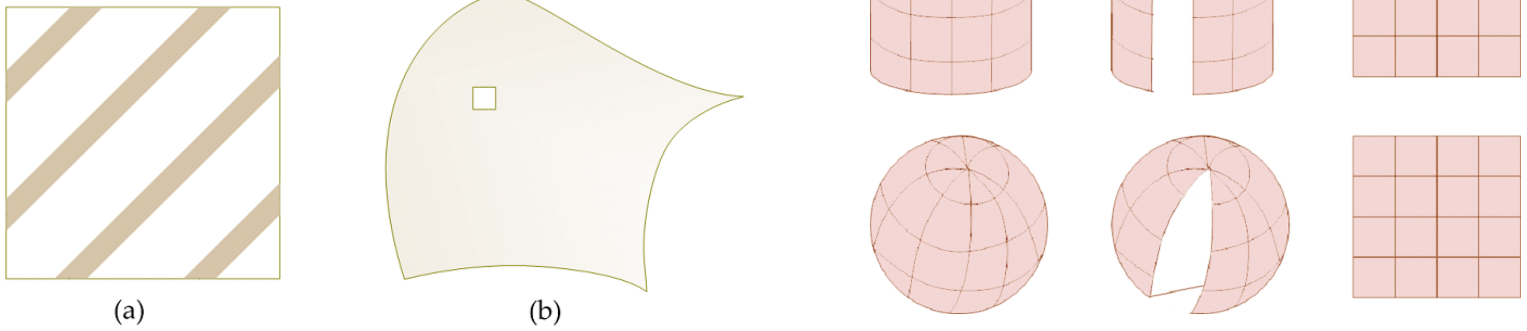


Mip-Mapping in OpenGL

- Initialization
  - glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST_MIPMAP_LINEAR);
- Texture Setup
  - glTexImage2D(..., *myImage*);
- Compute MIP-Maps
  - gluBuild2DMipmaps(GL_TEXTURE_2D, GL_RGB, *width*, *height*, GL_RGB, GL_UNSIGNED_BYTE, *myImage*);

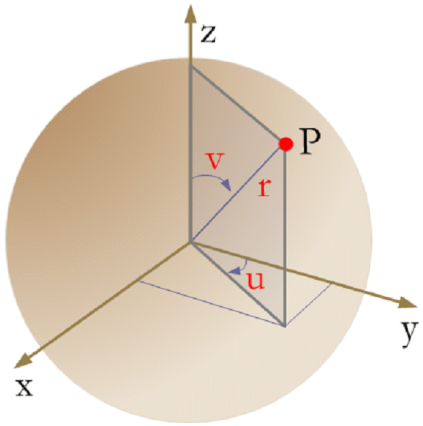
Texture on a Curved Surface

- Issues
  - Mapping a flat image onto a curved surface
  - Constructing a flat texture image of a curved surface

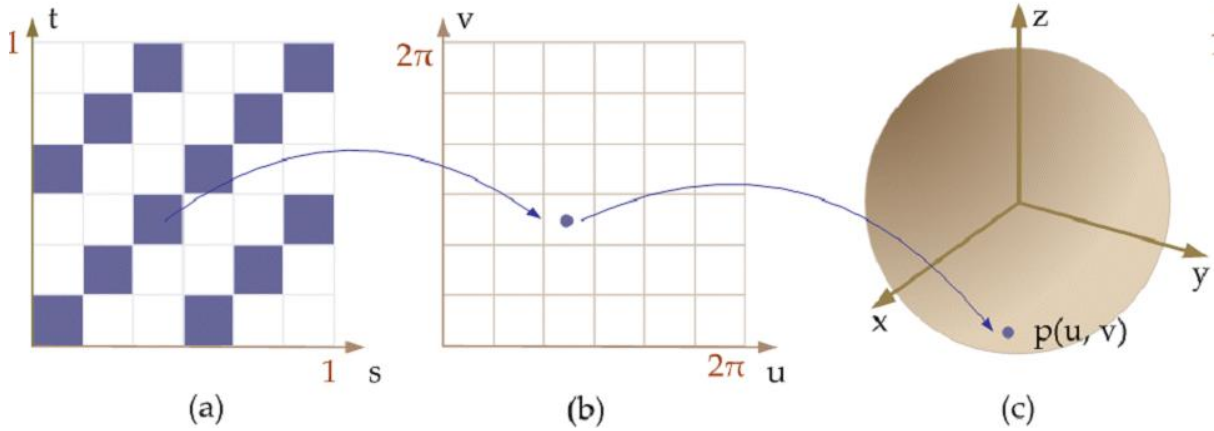


Parameterization

- Ex) Sphere (a point on the surface can be expressed with longitude and latitude)



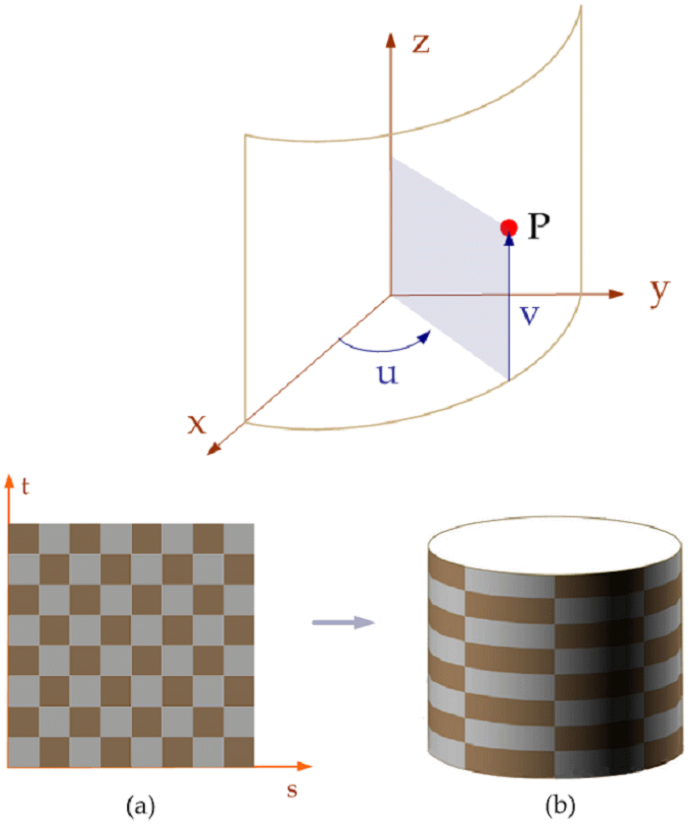
$$z = r\cos v$$
$$y = r\sin v\ \cos u$$
$$x = r\sin v\ \sin u$$



$$u = 2\pi s, \quad v = 2\pi t$$

$$z = r\cos 2\pi t$$
$$y = r\sin 2\pi t\ \cos 2\pi s$$
$$x = r\sin 2\pi t\ \sin 2\pi s$$

Cylinder Parameterization



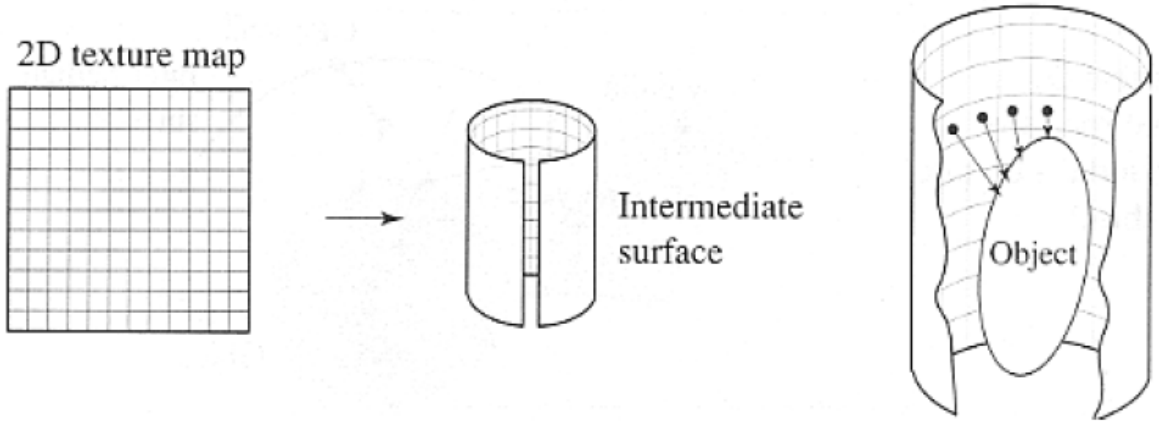
$$x = r\cos u,\ y = r\sin u,\ z = v$$

$$u = 2\pi s,\quad v = t$$

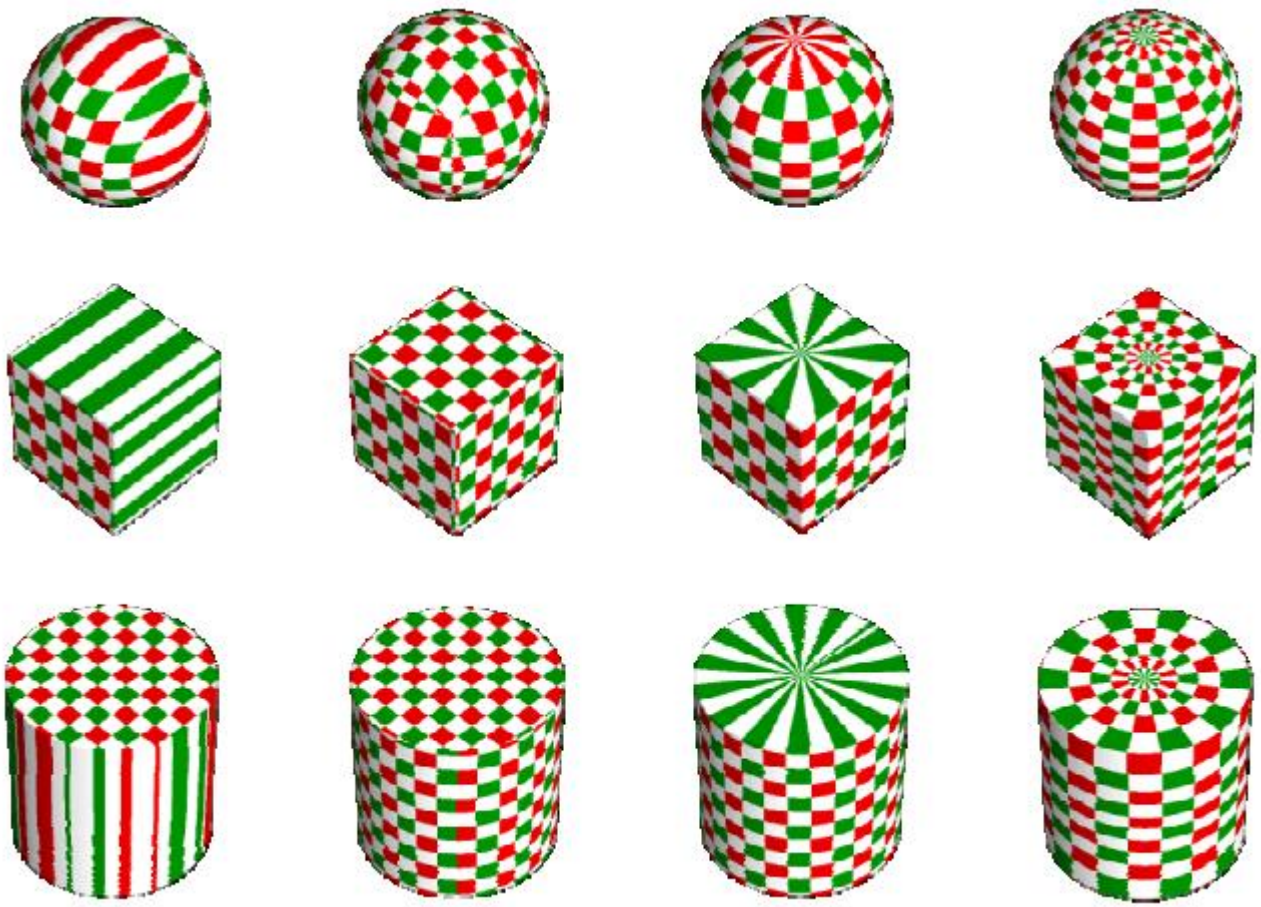
$$x = r\cos 2\pi s,\ y = r\sin 2\pi s,\ z = t$$

Two-Stage Mapping

- For arbitrarily shaped 3D surfaces
  - Mapping from 2D texture space to a simple 3D intermediate surface such as a sphere or a cylinder (*S mapping*)
  - Mapping from the intermediate surface to the destination object surface (*O mapping*)

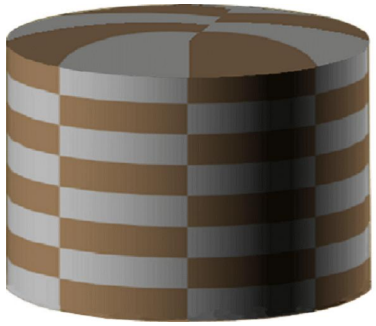


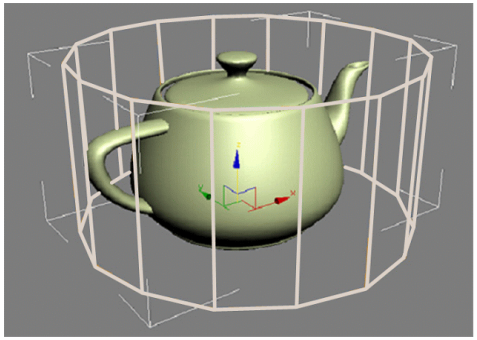
Varieties of projections

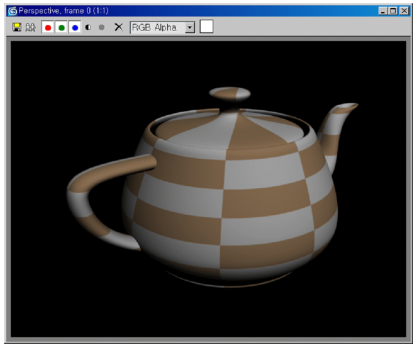


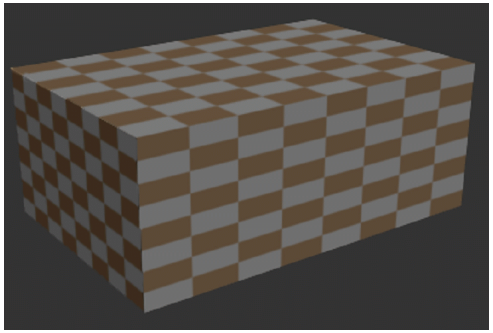
Plane Mapping | Cube Mapping | Cylinder Mapping | Sphere Mapping

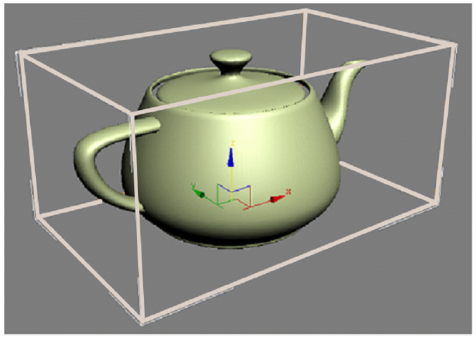
O Mapping

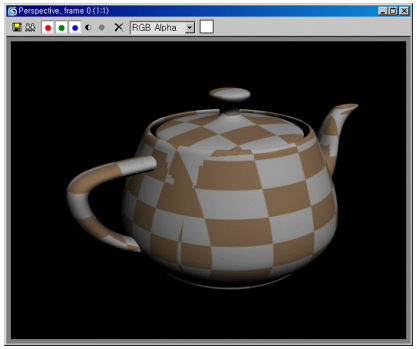


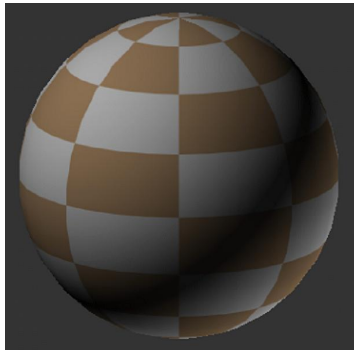


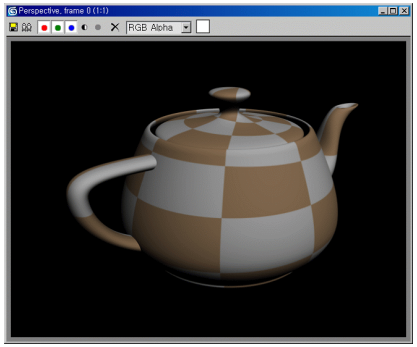












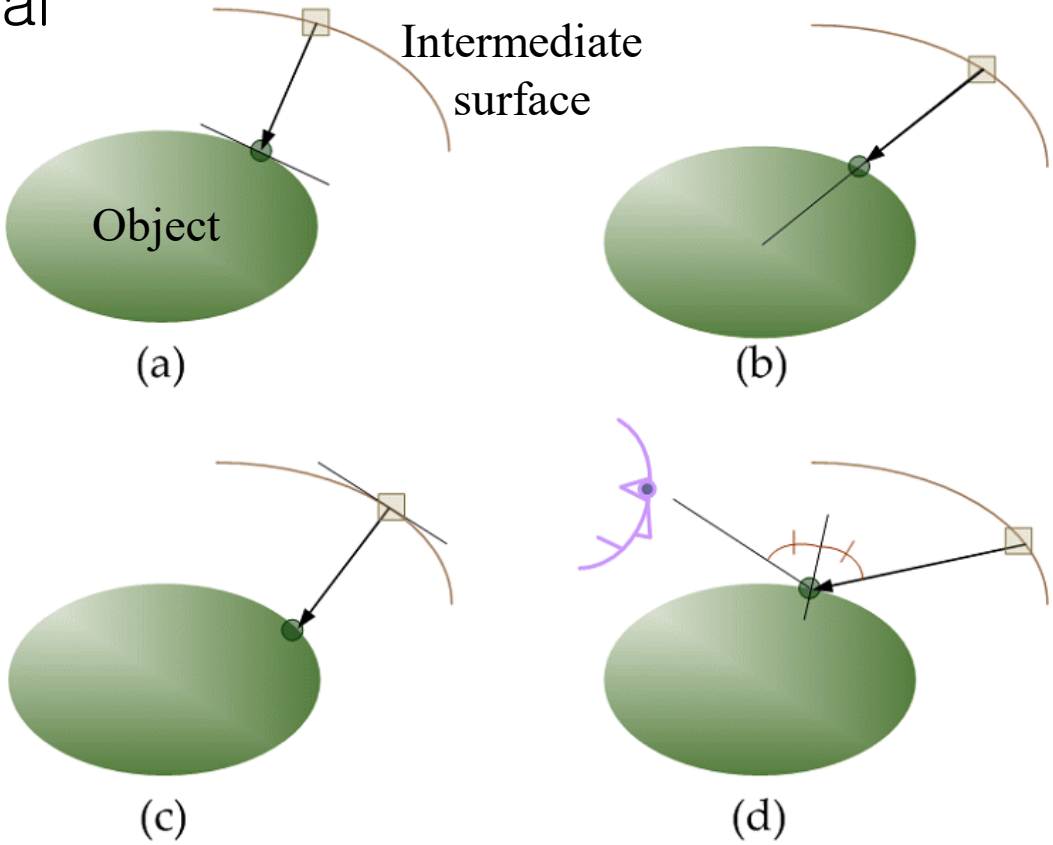
O Mapping

(a) Object surface normal

(b) Center of the object

(c) Intermediate surface normal

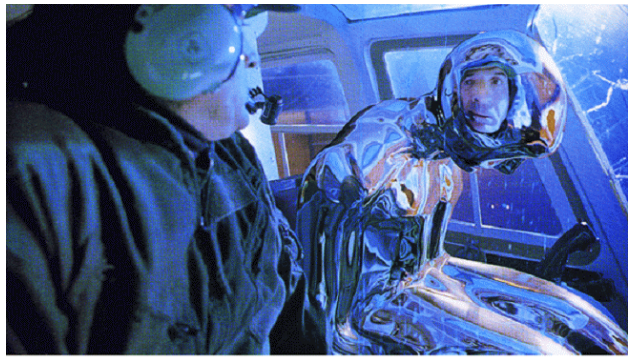
(d) Reflected view ray



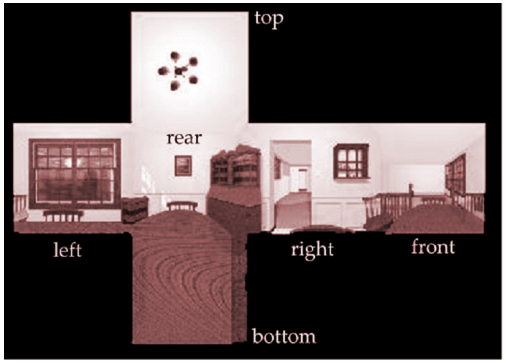
Environmental Mapping

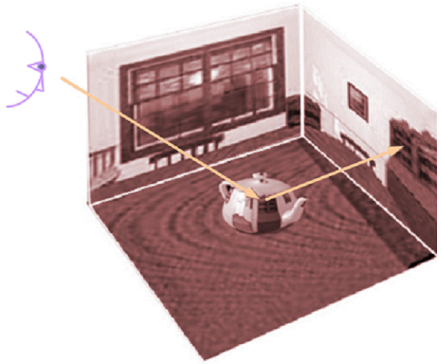
- Specular reflection on a glossy surface (Reflective Mapping).

  Ex) Terminator II



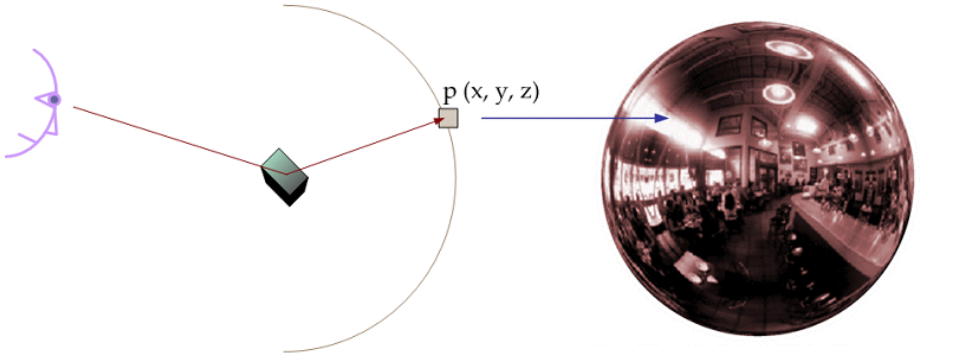
- Use two-stage mapping
  - Reflected view ray for O mapping
  - Different for different view point





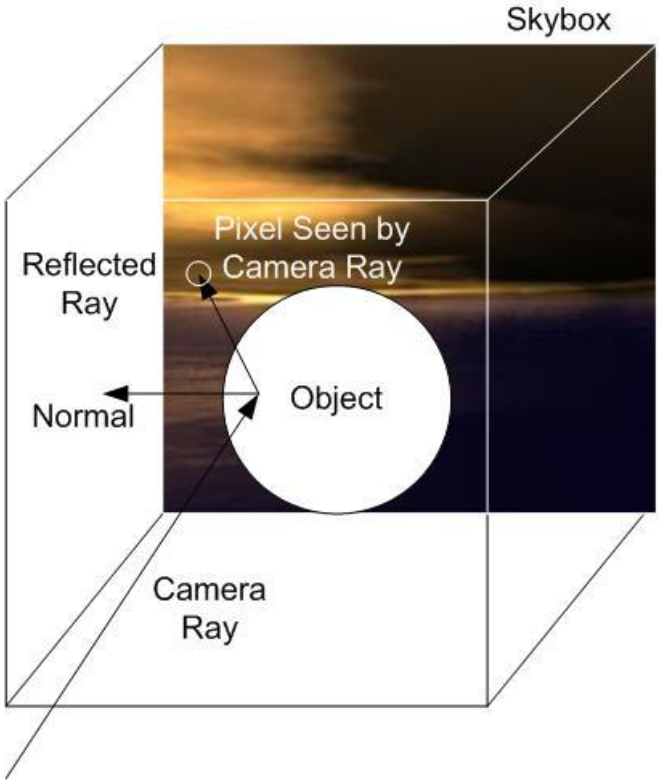
Environmental Mapping

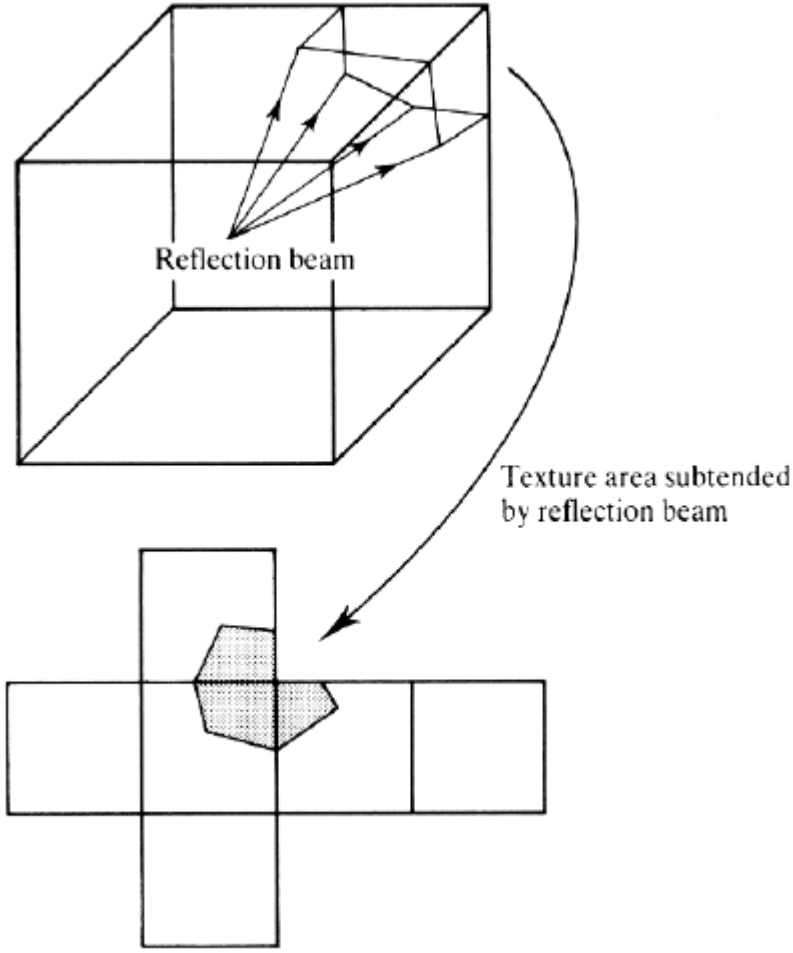
- Spherical environment map
  - Picture of surrounding environment
  - Spherical mapping



Environmental Mapping

- Cubic environment map
  - Subdivide into 6 surfaces





Environmental Mapping

- Environment map acquisition
  - Glossy ball, Fisheye, omni-directional lens