

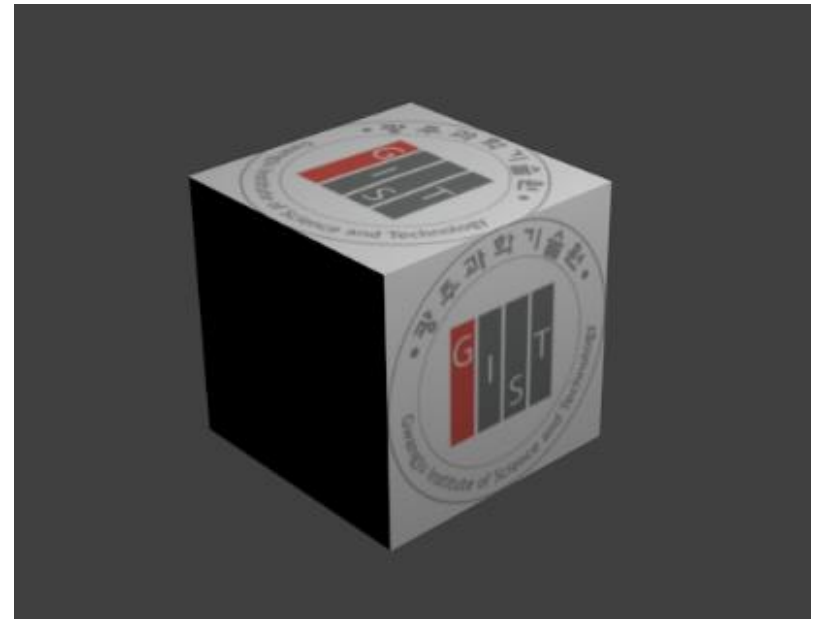
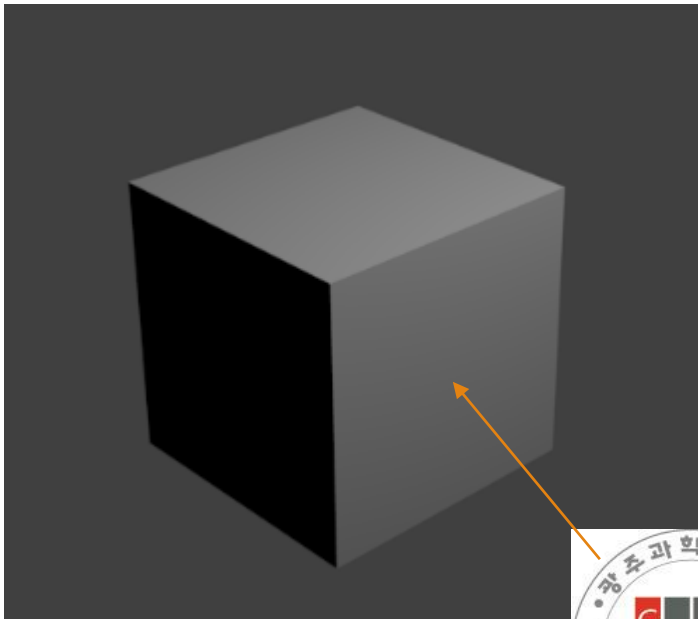
CT5510: Computer Graphics

Texture Mapping

BOCHANG MOON

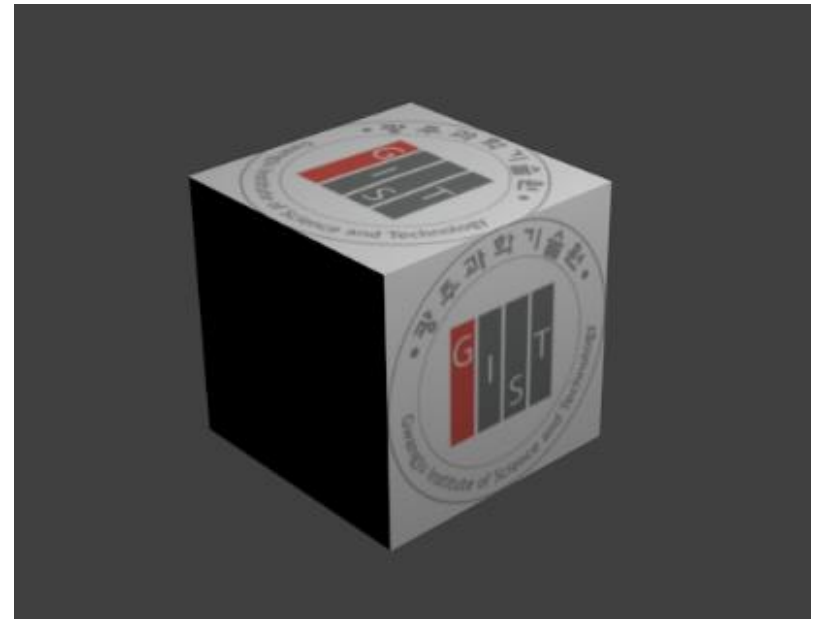
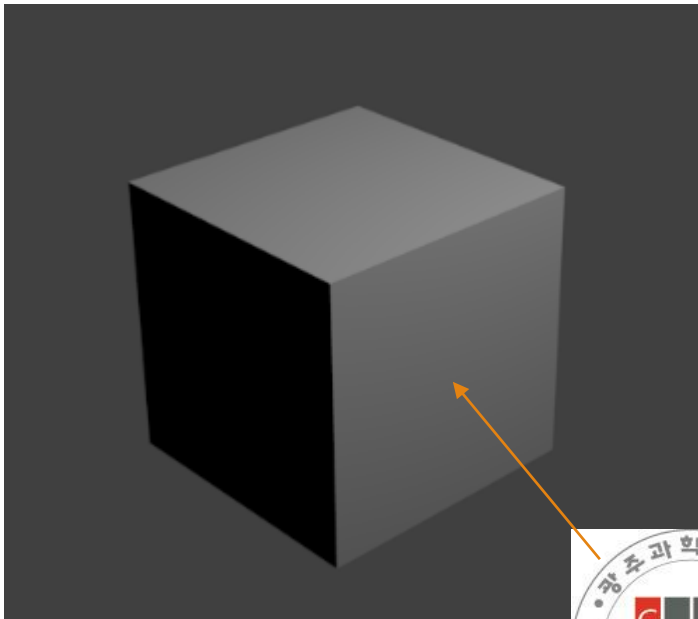
Texture Mapping

- Simulate spatially varying surface properties
 - Phong illumination model is coupled with a material (e.g., color)
 - Add small polygons with different materials
 - Very expensive



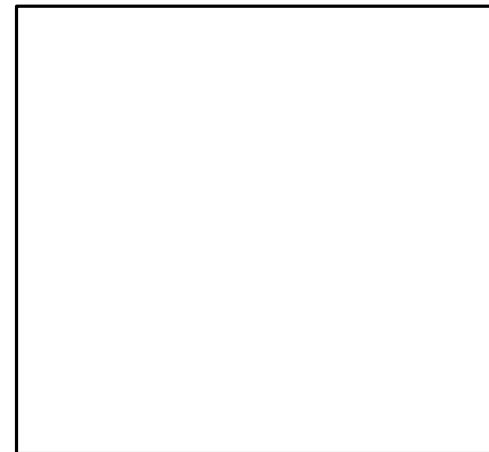
Texture Mapping

- Simulate spatially varying surface properties
 - Geometry of a surface dose not change, but materials need to be changed
 - Add an image onto the surface
 - Need to define a mapping function from the image to the surface



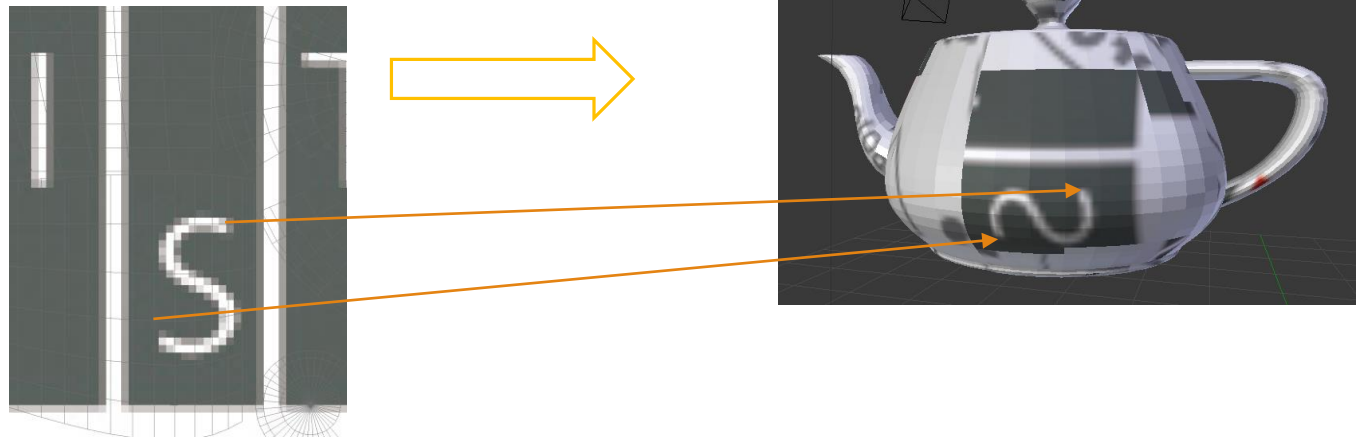
Texture Mapping

- Texture lookup
 - Color lookup(Image T, float u, float v) {
 - $(x, y) = \text{map_function}(u, v)$
 - return T(x,y)
 - }
- Note:
 - Each vertex has a texture coordinate (u, v)
 - A point inside a polygon has an interpolated coordinate



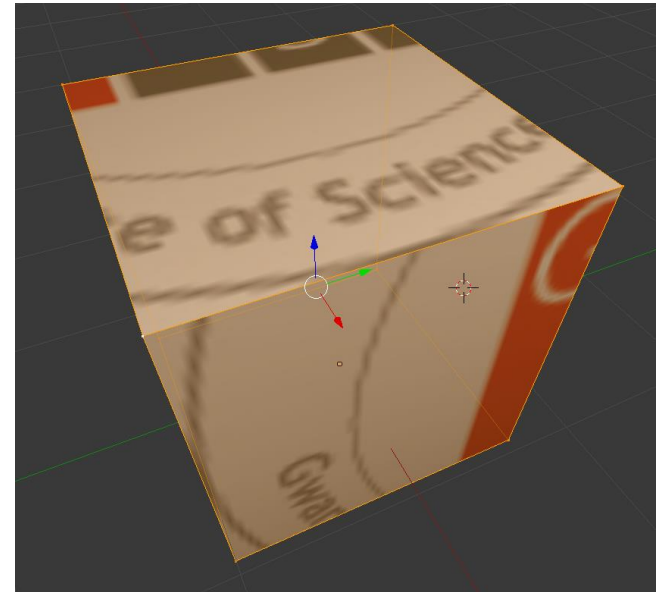
Texture Mapping

- Texture lookup
 - Color lookup(Image T, float u, float v) {
 - $(x, y) = \text{map_function}(u, v)$
 - return T(x,y)
 - }
- Note:
 - Each vertex has a texture coordinate (u, v)
 - A point inside a polygon has an interpolated coordinate



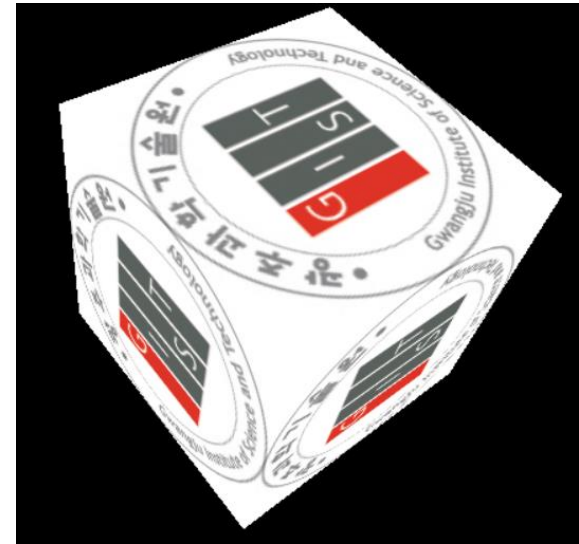
Texture Mapping in OpenGL

- Assign a texture coordinate, (u, v) , to each vertex



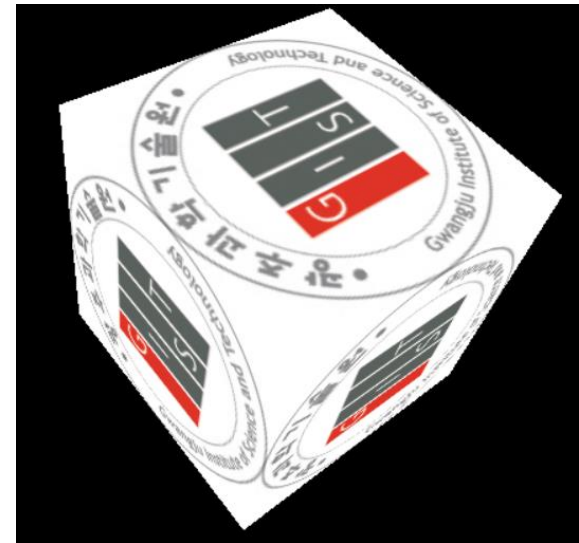
Texture Mapping in OpenGL

- General procedure for creating a texture map:
 - texImg = Read an image file (jpg, bmp, png, exr, ...)
 - Gluint textureID[1];
 - glGenTextures(1, &textureID[0]); // create n texture name(ID) (e.g., n = 1)
 - glBindTexture(GL_TEXTURE_2D, textureID[0]); // bind a texture to the target (e.g., GL_TEXTURE_2D)
 - glTexImage2D(GL_TEXTURE_2D, 0, 3, width of texImg, height of texImg, 0, GL_RGB, GL_UNSIGNED_BYTE, data of texImg);
 - // 0: level-of-detail
 - // 3: number of color components



Texture Mapping in OpenGL

- General procedure of using the generated map:
 - `glBindTexture(GL_TEXTURE_2D, textureID[0]);`
 - `glBegin(GL_QUADS);`
 - `// Front Face`
 - `glTexCoord2f(0.0f, 0.0f); glVertex3f(-1.0f, -1.0f, 1.0f);`
 - `glTexCoord2f(1.0f, 0.0f); glVertex3f(1.0f, -1.0f, 1.0f);`
 - `glTexCoord2f(1.0f, 1.0f); glVertex3f(1.0f, 1.0f, 1.0f);`
 - `glTexCoord2f(0.0f, 1.0f); glVertex3f(-1.0f, 1.0f, 1.0f);`
 - ...
 - `glEnd();`



Texture Mapping

- Assign texture coordinates (normalized coordinates) at each vertex
 - (u, v) in the range $([0...1], [0...1])$
- Texture pixel (texel) is fetched given a (u, v) coordinate
 - e.g., $(u, v) \rightarrow (u_{tex}, v_{tex})$ in the range $([0 \dots width_{tex}], [0 \dots height_{tex}])$
 - As a result, $(x, y) \rightarrow (u, v) \rightarrow (u_{tex}, v_{tex})$

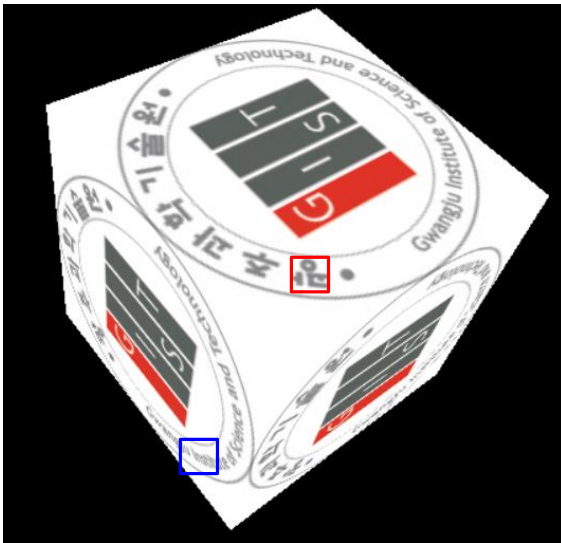


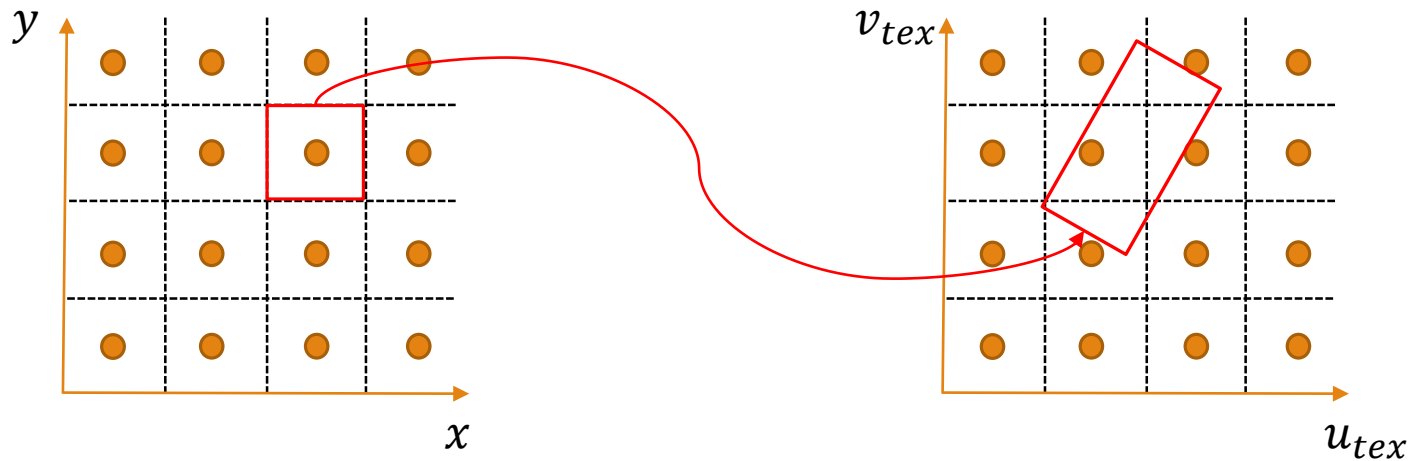
Image space



Texture space

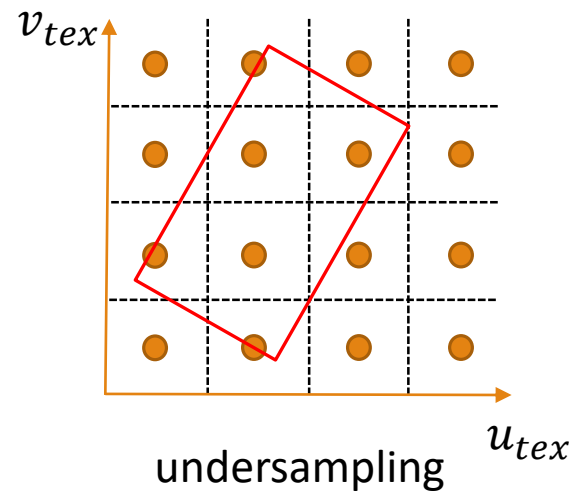
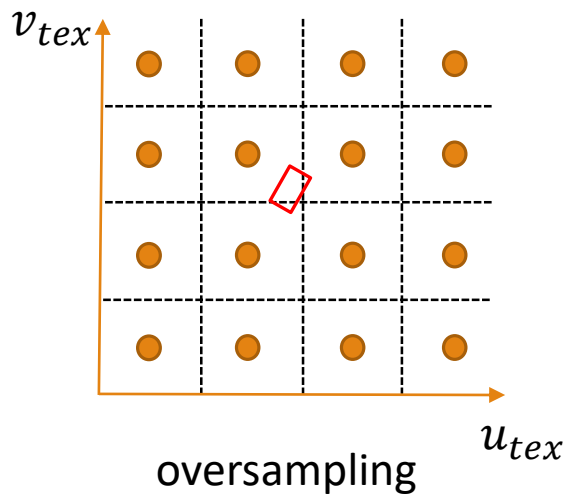
Texture Mapping

- Assign texture coordinates (normalized coordinates) at each vertex
 - (u, v) in the range $([0...1], [0...1])$
- Texture pixel (texel) is fetched given a (u, v) coordinate
 - e.g., $(u, v) \rightarrow (u_{tex}, v_{tex})$ in the range $([0 \dots width_{tex}], [0 \dots height_{tex}])$
 - As a result, $(x, y) \rightarrow (u, v) \rightarrow (u_{tex}, v_{tex})$



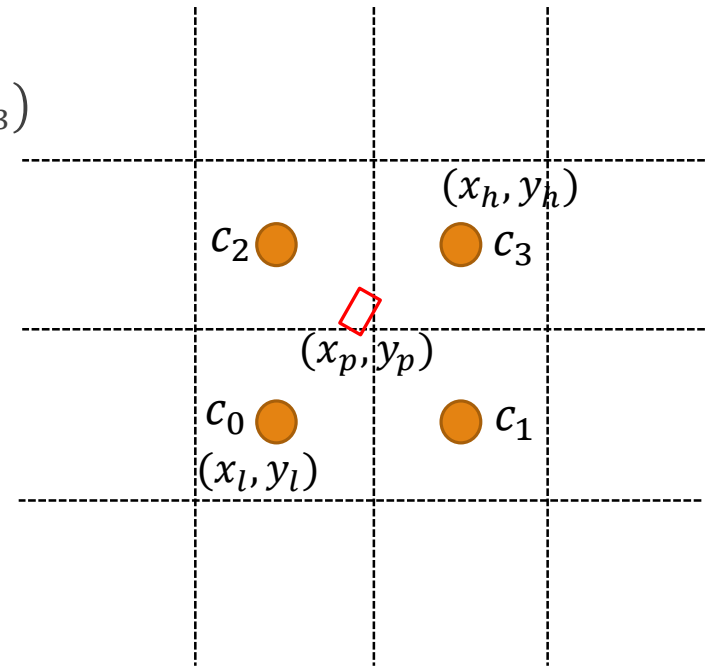
Issues of Texture Mapping

- Oversampling (magnification): one pixel is corresponding to less than a texel
- Undersampling (minification): one pixel is corresponding to more than a texel



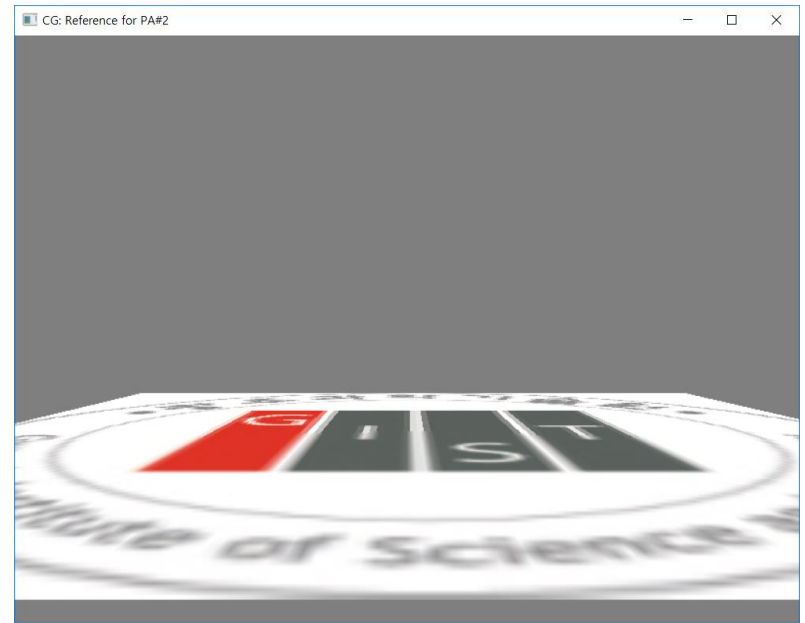
Texture Filtering for Oversampling

- Filtering methods
 - Nearest neighbor: take the color of the closest texel
 - Bilinear interpolation:
 - $\alpha = \frac{x_p - x_l}{x_h - x_l}$
 - $\beta = \frac{y_p - y_l}{y_h - y_l}$
 - $c_p = (1 - \beta)((1 - \alpha)c_0 + \alpha c_1) + \beta((1 - \alpha)c_2 + \alpha c_3)$



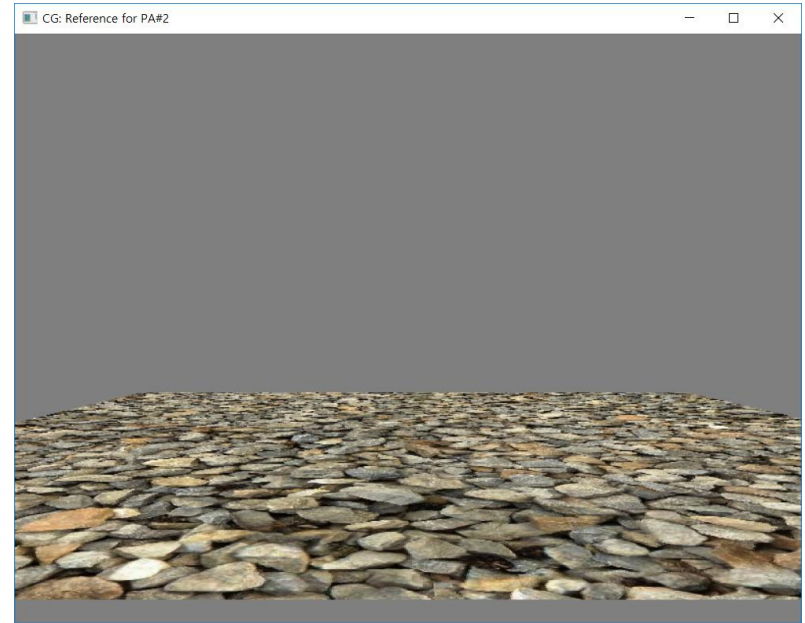
Texture Filtering for Oversampling

- Filtering methods
 - Nearest neighbor: take the color of the closest texel
 - Bilinear interpolation:



Texture Filtering for Undersampling

- High-frequency details in a small region introduce an image artifact (e.g., aliasing)
 - Should integrate multiple texels on the fly
 - Requires multiple read operations (expensive)



Texture Filtering for Undersampling

- High-frequency details in a small region introduce an image artifact (e.g., aliasing)
 - Should integrate multiple texels on the fly
 - Requires multiple read operations (expensive)
 - MIP Mapping: Prepare multiple-resolution (pre-filtered) images in preprocessing and select a texel from a MIP level



$width \times height$

$$\frac{width}{2} \times \frac{height}{2}$$



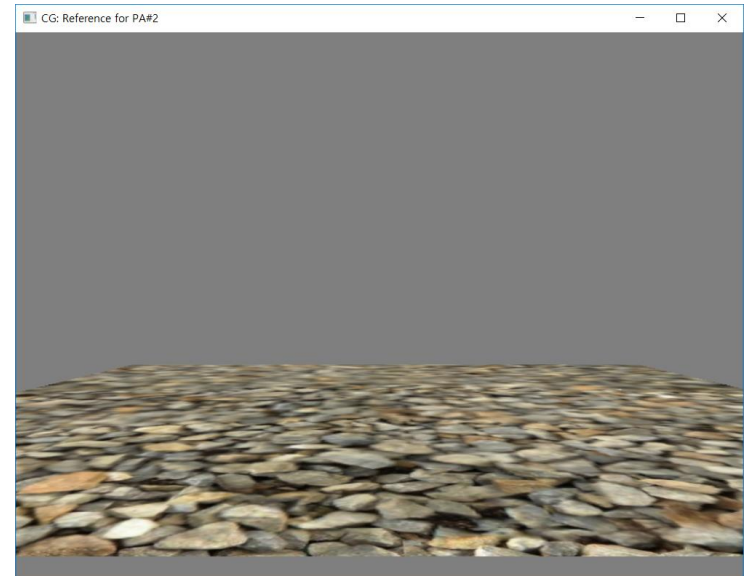
$$\frac{width}{8} \times \frac{height}{8}$$



$$\frac{width}{4} \times \frac{height}{4}$$

Texture Filtering for Undersampling

- High-frequency details in a small region introduce an image artifact (e.g., aliasing)
 - Should integrate multiple texels on the fly
 - Requires multiple read operations (expensive)
 - MIP Mapping: Prepare multiple-resolution (pre-filtered) images in preprocessing and select a texel from a MIP level
 - Two adjacent MIP levels can be interpolated



Texture Filtering in OpenGL

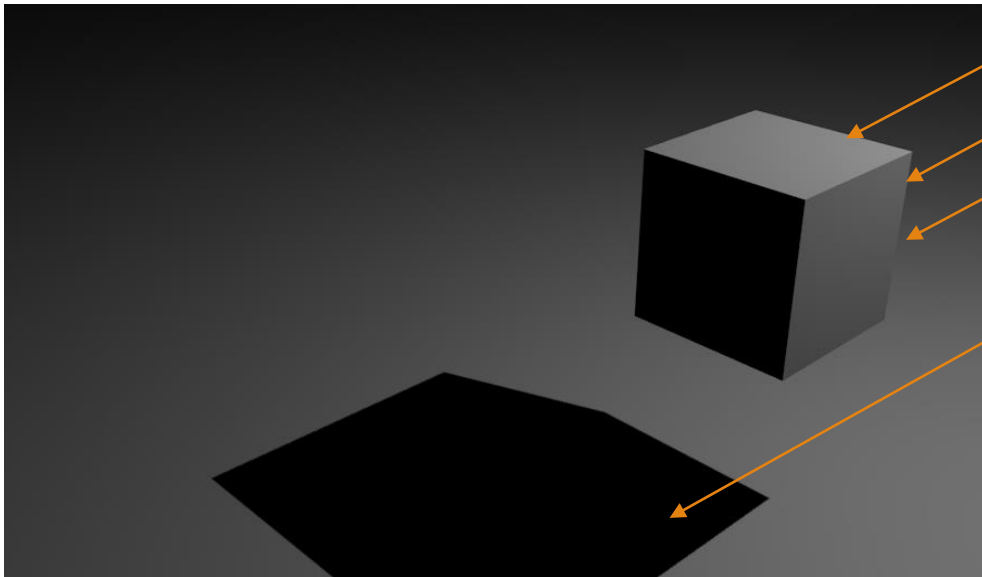
- MIP map generation
 - `gluBuild2DMipmaps(GL_TEXTURE_2D, 3, width, height, GL_RGB, GL_UNSIGNED_BYTE, image data);`
- Filtering methods
 - `glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, filter);`
 - `glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, filter);`
- Filters for magnification
 - `GL_NEAREST, GL_LINEAR`
- Filters for minification
 - `GL_NEAREST, GL_LINEAR, GL_NEAREST_MIPMAP_NEAREST, GL_LINEAR_MIPMAP_NEAREST, GL_NEAREST_MIPMAP_LINEAR, GL_LINEAR_MIPMAP_LINEAR`
 - Note: `GL_XX_MIPMAP_LINEAR` // choose the two MIP maps that closely match the size of pixels for an interpolation
 - Note: `GL_LINEAR_MIPMAP_NEAREST` // use the `GL_LINEAR` interpolation within a MIP level

Applications: Material Parameters

- Phong illuming model
 - $I = \sum_{i=1}^{\# \text{ of lights}} L_a^i k_a + L_d^i k_d \max(0, \mathbf{n} \cdot \mathbf{l}^i) + L_s^i k_s \max(0, \mathbf{r}^i \cdot \mathbf{v})^s$
- Q. Which parameters can be changed from the texture mapping?
 - k_a, k_d, k_s
- Lights can be textured as well
 - e.g., TV screen or your monitor (area lights)

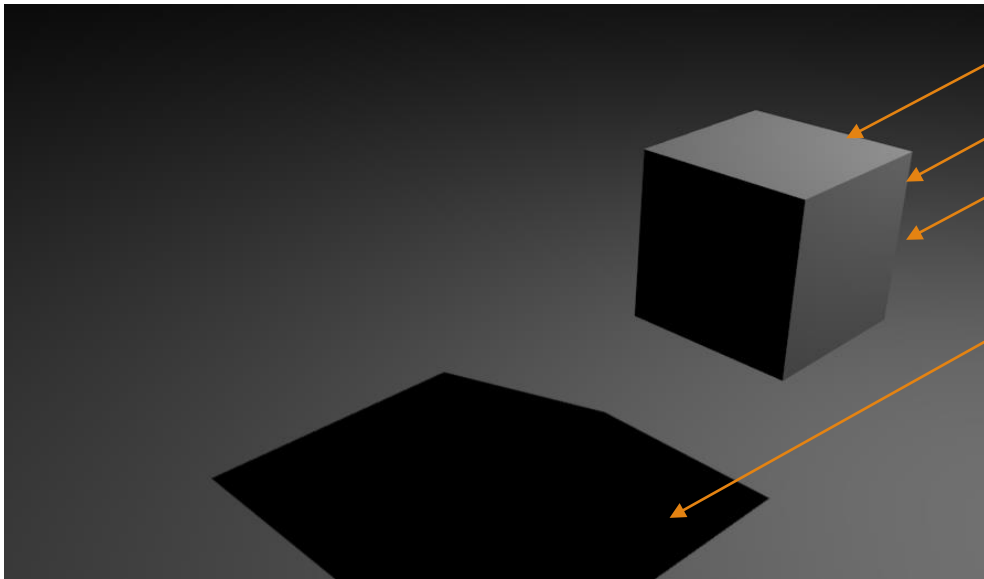
Applications: Shadow Maps

- Shadow mapping
 - 1. Pre-render a scene from a light source and store depths in a shadow map
 - 2. Render a scene from a view point while performing an extra test
 - If (distance between the point (i.e., fragment) and a light $>$ the stored depth)
 - This point is in shadow.



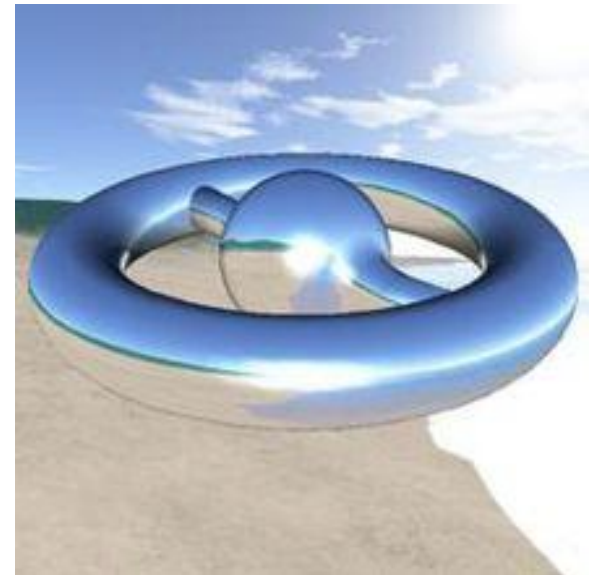
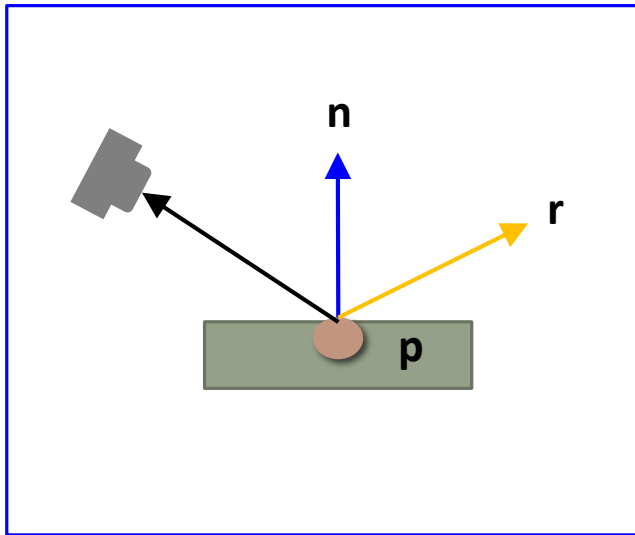
Applications: Shadow Maps

- Issues with shadow mapping
 - Area lights
 - Multiple lights
 - ...



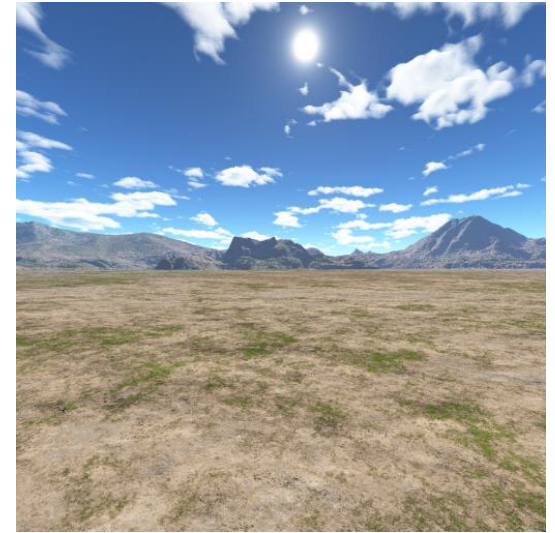
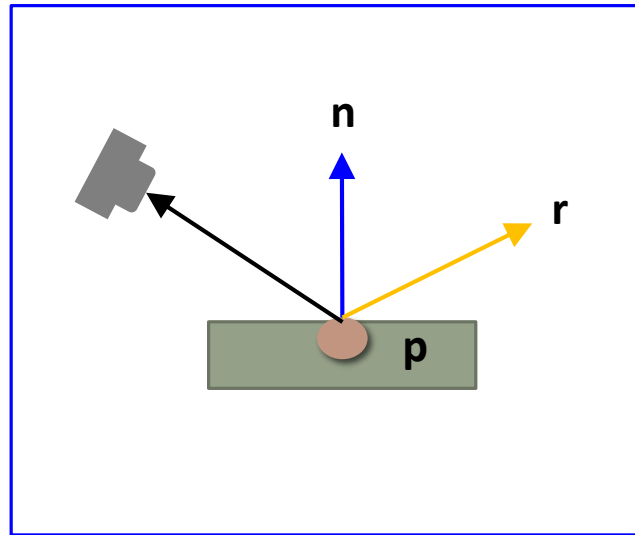
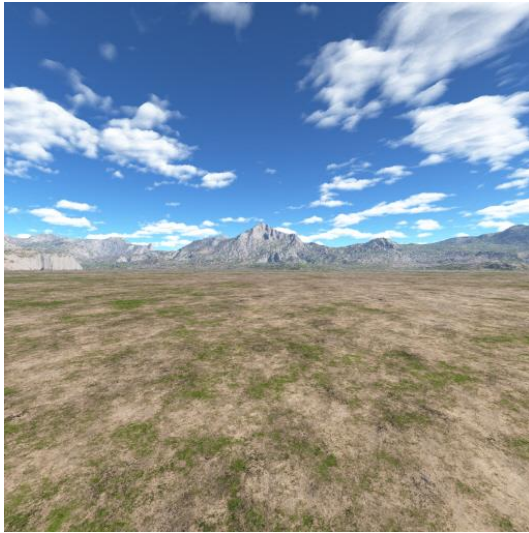
Applications: Environment Maps

- Environment mapping (reflection mapping) is an image-based lighting that approximates reflections (e.g., indirect illumination) on surfaces, by using pre-computed textures
 - Simple geometries (e.g., sphere, cube) are usually used to approximate the environment
 - The geometries are intermediate objects for texture mapping



Applications: Environment Maps

- Environment mapping (reflection mapping) is an image-based lighting that approximates reflections (e.g., indirect illumination) on surfaces, by using pre-computed textures
 - Simple geometries (e.g., sphere, cube) are usually used to approximate the environment
 - The geometries are intermediate objects for texture mapping



Applications: Bump Maps

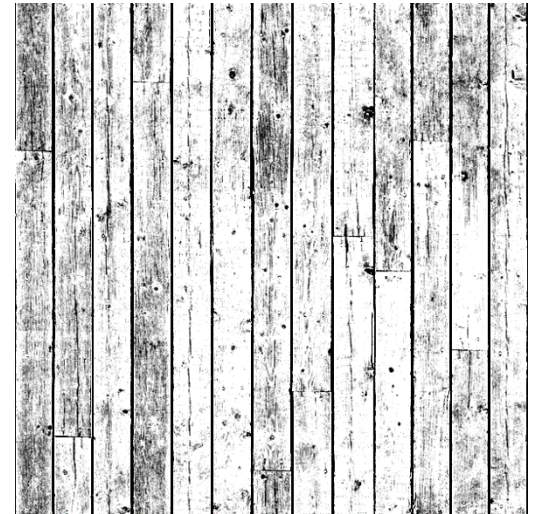
- Approaches to model rough (bumpy) surfaces
 - Add complex geometries
 - Perturb surface normal based on a texture image



Rendering result



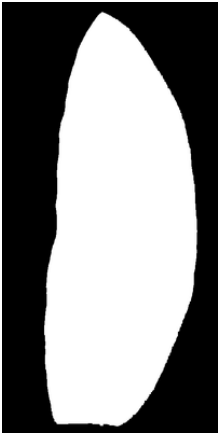
Diffuse texture map



Bump map

Applications: Other Maps

- Normal mapping
 - Replace the normal at a point with a pre-computed normal (r, g, b) at texel
- Opacity maps
 - Use black and white (or alpha channels) to make some areas of a surface transparent



Further Readings

- Chapter 11