Interactive Computer Graphics

CS 418 – Spring 2011

MP3 Projection, Hidden Surface Removal and Texturing Mapping

TA: Mahsa Kamali

Email: mkamali2 at illinois.edu

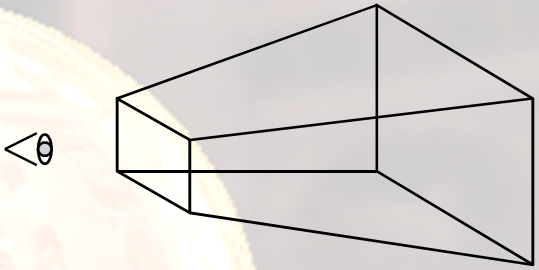
Office Hours

Location: 0207 Siebel Center

Time: Thursdays 2-3PM

Projection Transformation

- Shape of viewing frustum



- Perspective projection

```
gluPerspective( fovy, aspect, zNear, zFar )
glFrustum( left, right, bottom, top, zNear, zFar )
```

- Orthographic parallel projection

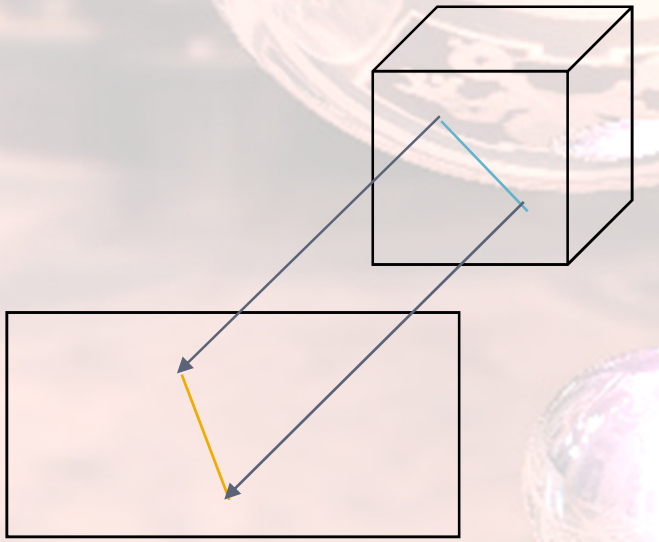
```
glOrtho( left, right, bottom, top, zNear, zFar )
gluOrtho2D( left, right, bottom, top )
```

  - calls **glOrtho** with z values near zero

Applying Projection Transformations

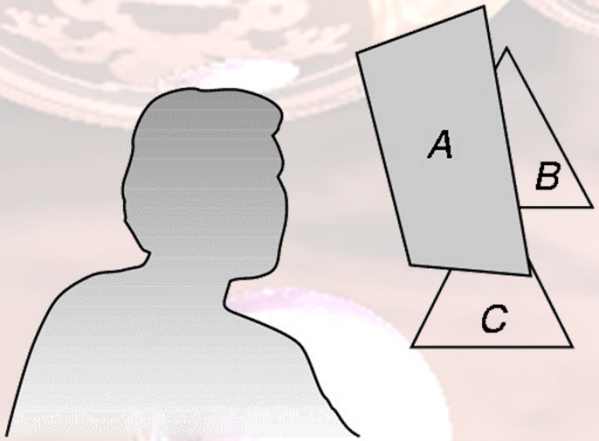
- Typical use (orthographic projection)

```
glMatrixMode( GL_PROJECTION );
glLoadIdentity();
glOrtho( left, right, bottom, top, zNear, zFar );
```



Hidden-Surface Removal

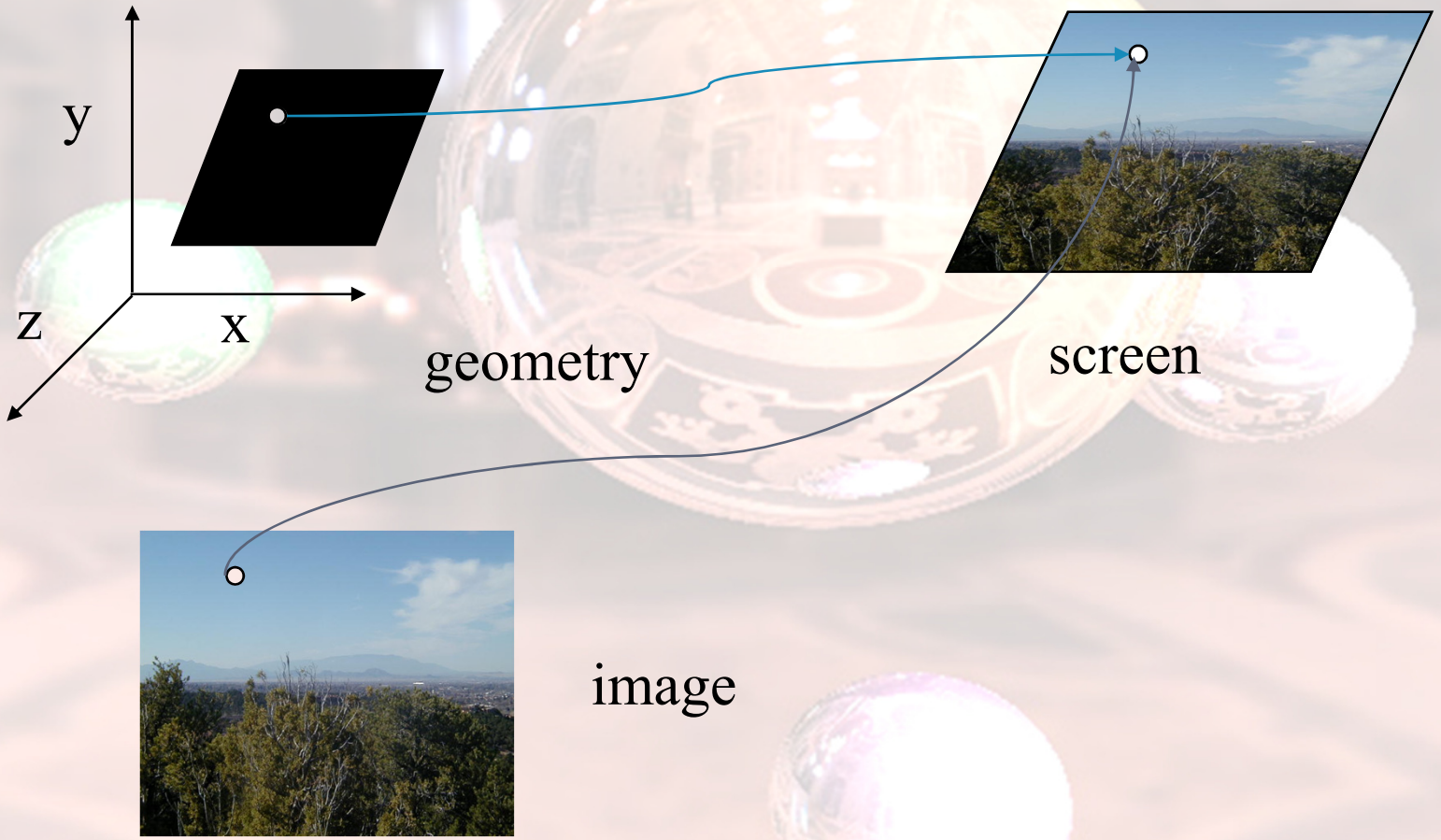
- We want to see only those surfaces in front of other surfaces
- OpenGL uses a *hidden-surface* method called the *z*-buffer algorithm that saves depth information as objects are rendered so that only the front objects appear in the image



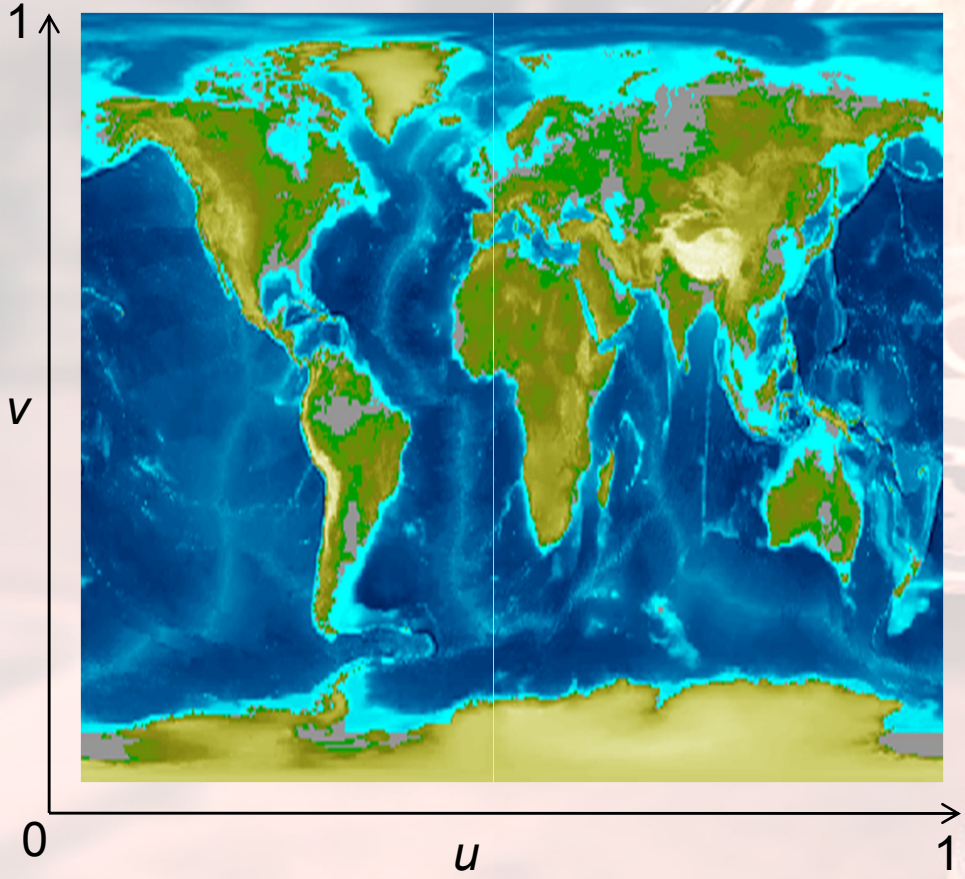
Using the *z*-buffer algorithm

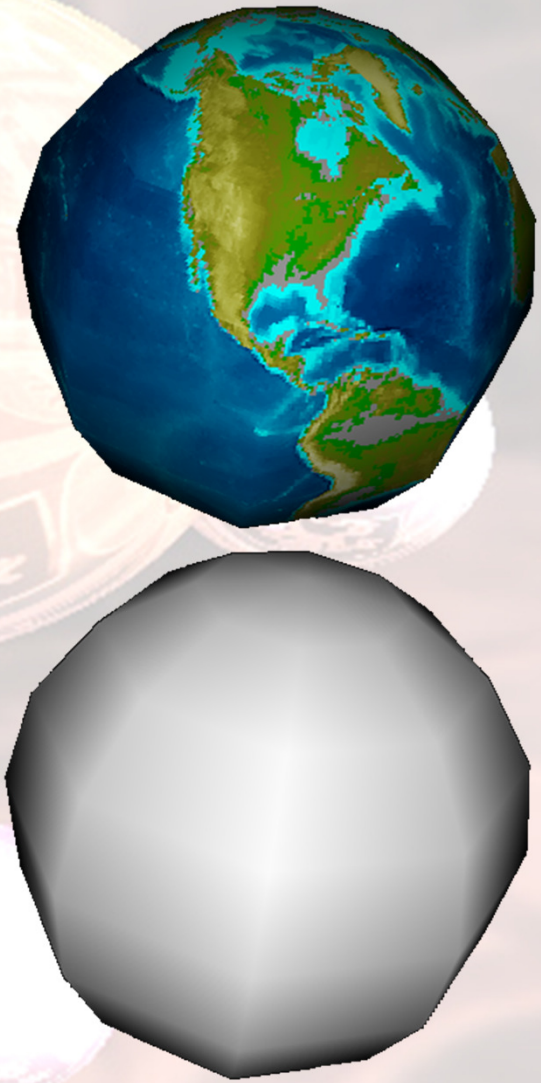
- The algorithm uses an extra buffer, the z-buffer, to store depth information as geometry travels down the pipeline
- It must be
  - Requested in **`main.c`**
    - **`glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH)`**
  - Enabled in **`init.c`**
    - **`glEnable(GL_DEPTH_TEST)`**
  - Cleared in the display callback
    - **`glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)`**

Texture Mapping

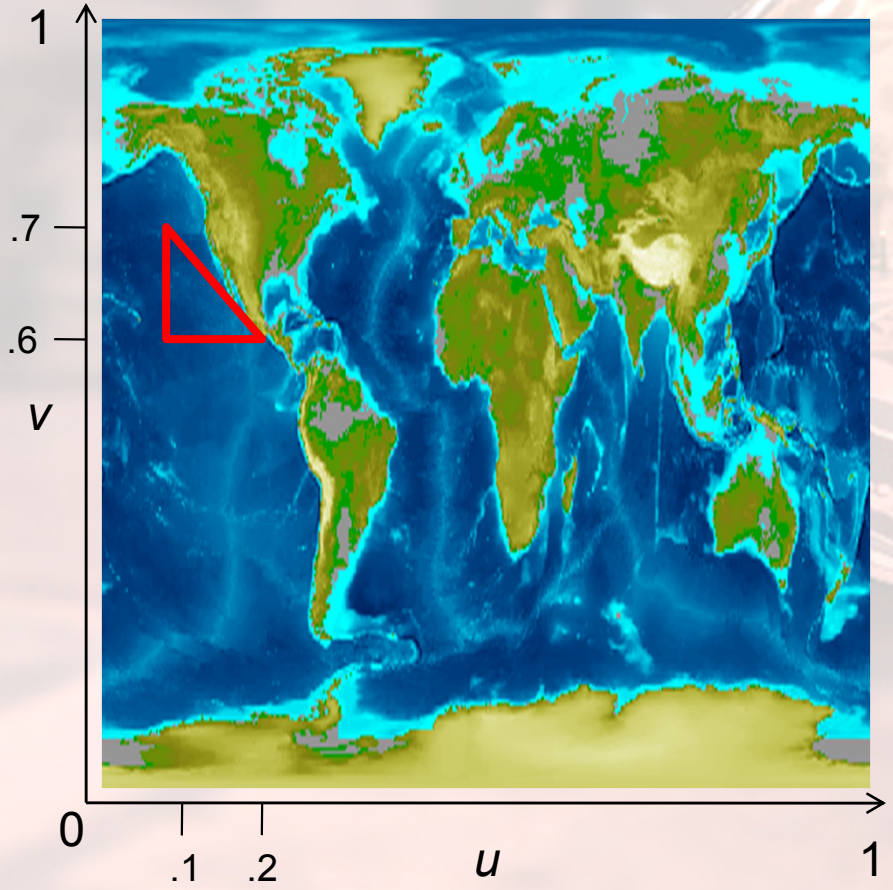


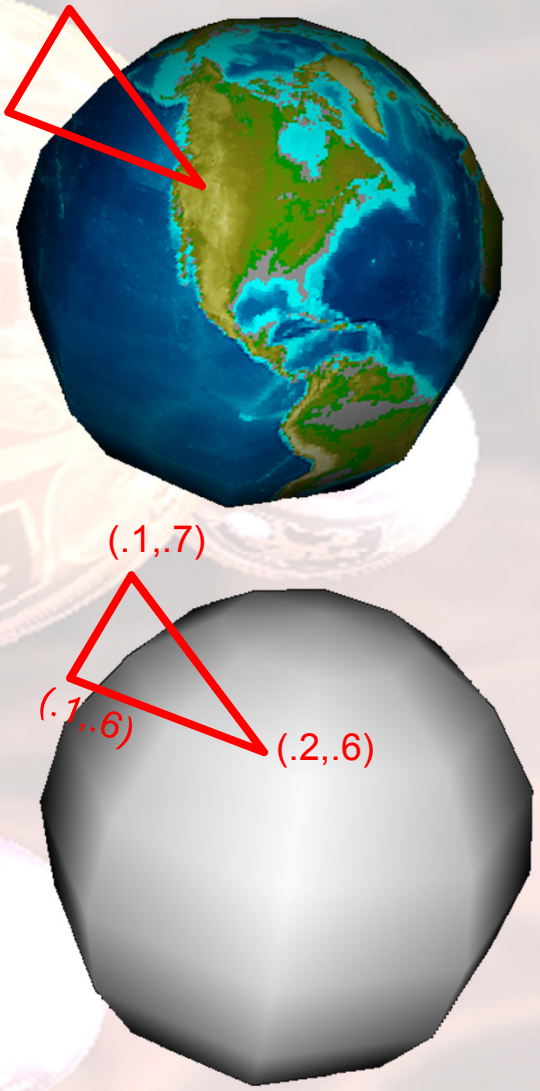
Texture Mapping





Texture Mapping





Texture Mapping

- Steps
  - Specify texture
    - Read or generate image
    - Link image to texture
  - Set texturing parameters.
  - Assign texture coordinates to vertices.
  - Draw the scene with texture mapping.

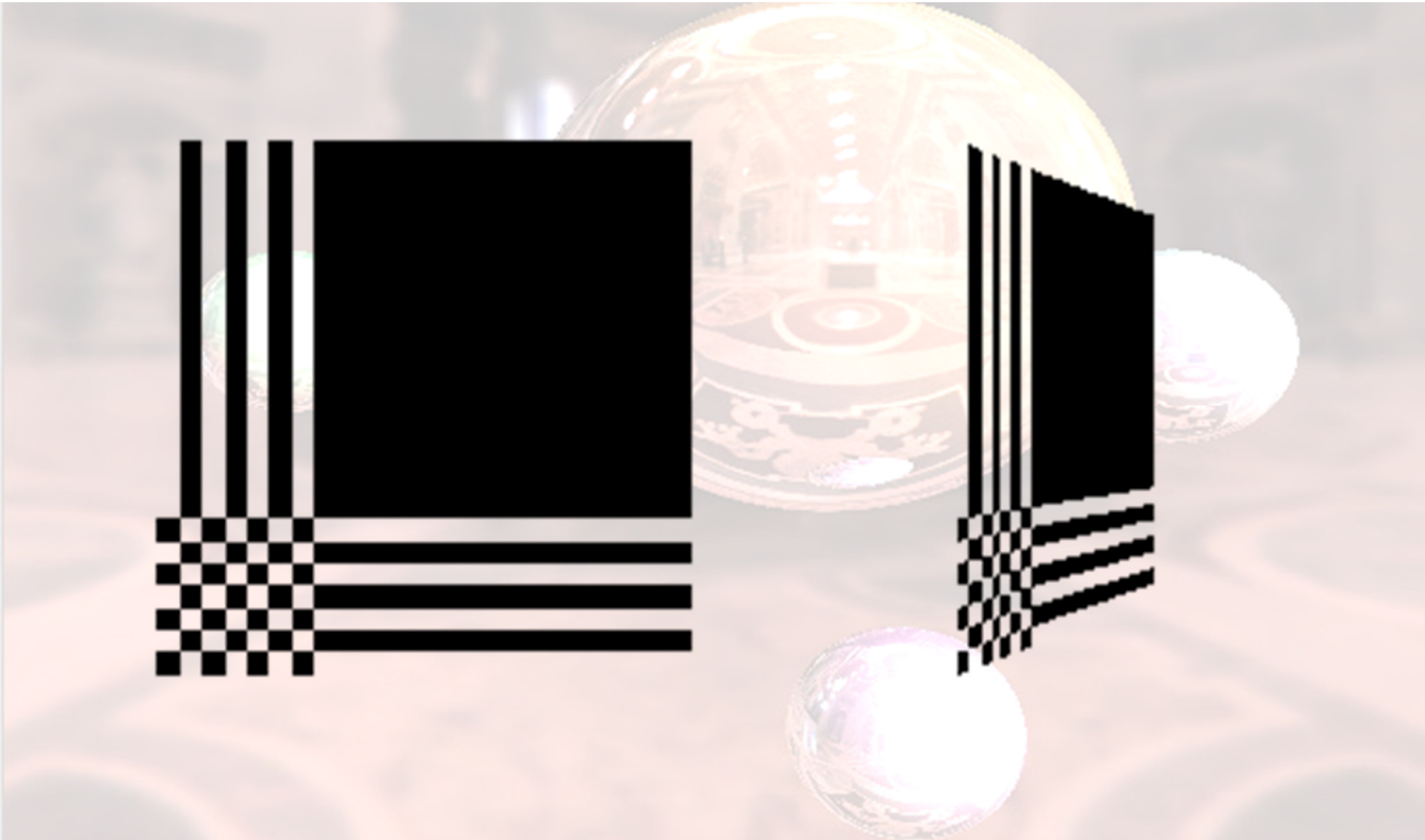
Texture Mapping

- Texture Objects
  - one image per texture object
  - Faster to use an existing object than reload a texture image
- Generate texture ***names***
  - **`glGenTextures( n, *texIds );`**
  - Only the name(s) is created. texId = zero is reserved.
  - No information about its data/dimension is created.
- Create/Bind textures with the ***name*** before using
  - **`glBindTexture( target, id );`**
  - Create a new texture object if called for first time
  - Activate the texture object with the same ID already exist.
  - If (id==0), unbind the current texture object

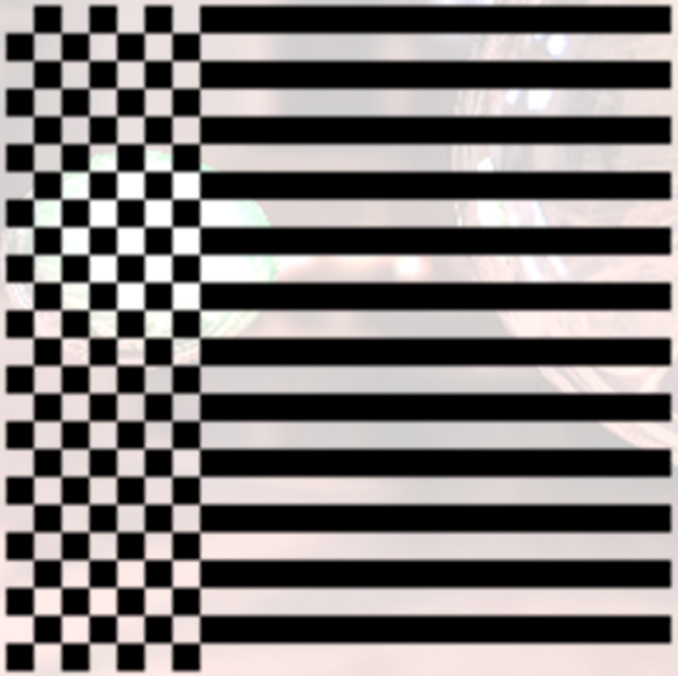
Set texturing parameters

```
// Set texturing parameters
glBindTexture(GL_TEXTURE_2D, texName);

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_MAG_FILTER, GL_LINEAR);

glTexParameteri(GL_TEXTURE_2D,
   GL_TEXTURE_MIN_FILTER, GL_LINEAR);
```

Texturing Clamping/Repeating



Texturing Clamping/Repeating

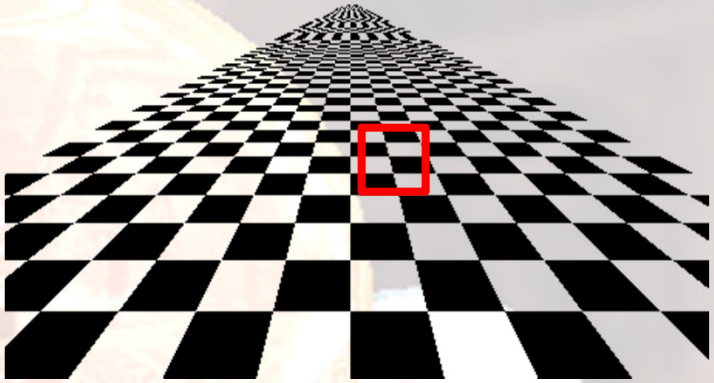


Texturing Clamping/Repeating

Parameter	Values
GL_TEXTURE_WRAP_S	GL_CLAMP, GL_REPEAT
GL_TEXTURE_WRAP_T	GL_CLAMP, GL_REPEAT

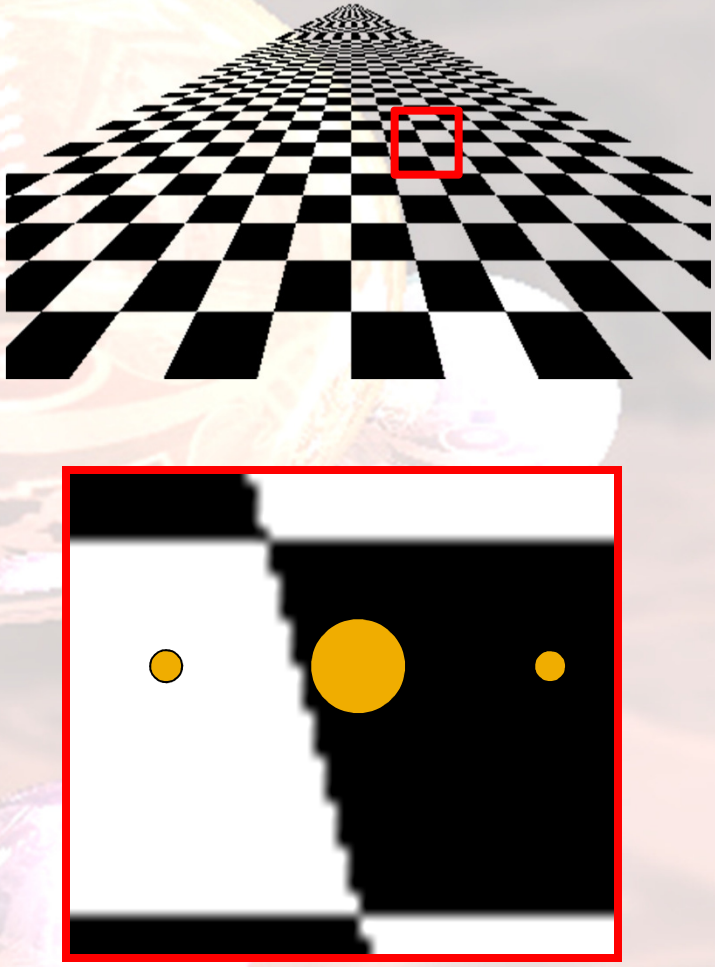
Magnification Aliasing

- "Jaggies" – lines have a staircased edge appearance
- Occur when a single texture sample (texels) projects to multiple screen pixels
- (Also occurs when rasterizing lines or polygon edges)



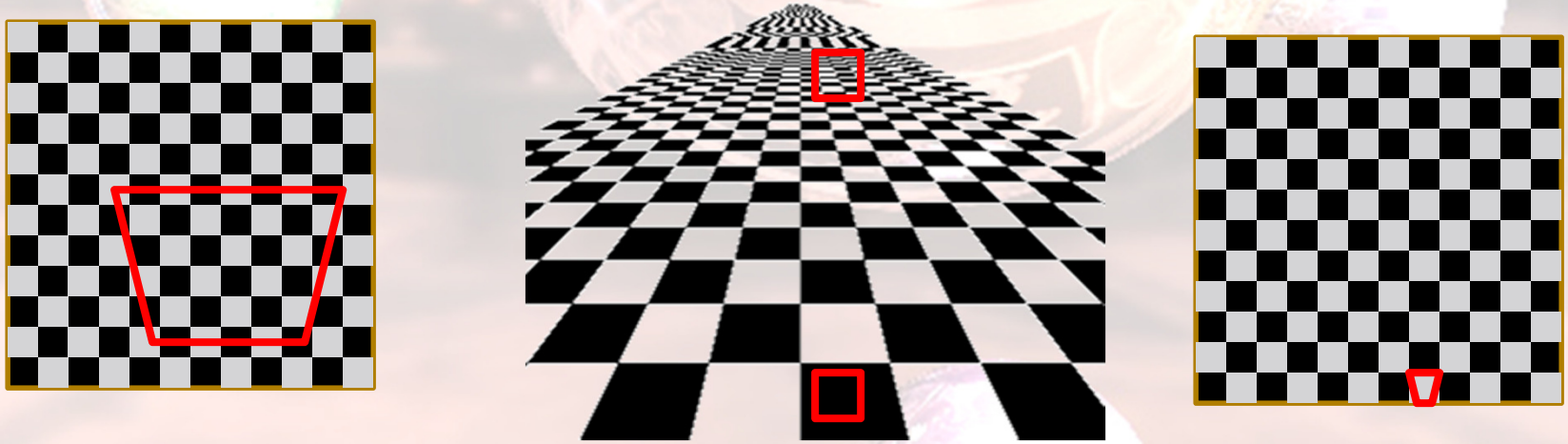
Bilinear Filtering

- "Jaggies" – lines have a staircased edge appearance
- Occur when a single texture sample (texels) projects to multiple screen pixels
- (Also occurs when rasterizing lines or polygon edges)
- Fixed by averaging neighboring samples to find the value between samples



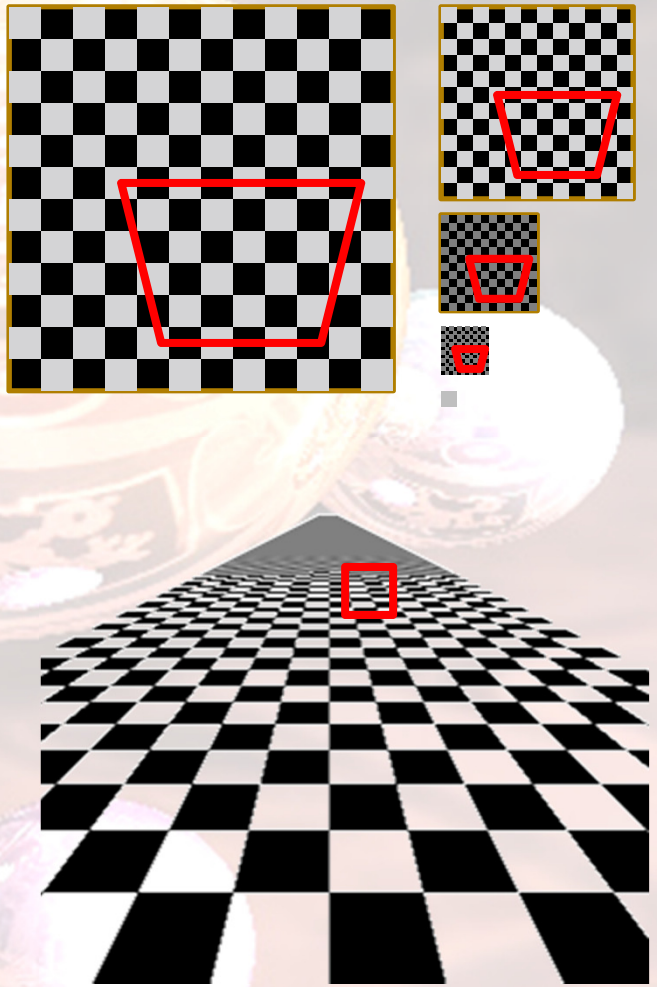
Minification Aliasing

- Many texture pixels (texels) map into a single screen pixel



MIP Mapping

- Many texture pixels (texels) map into a single screen pixel
- Cannot simply add them up because some pixels would take longer than others to add
- Create an image pyramid from the initial texture
- Each level of the pyramid half the resolution of the one below it
- Choose the texture resolution whose projected texel size most closely matches pixel size



Texture Filtering

Parameter	Values
GL_TEXTURE_MAG_FILTER	GL_NEAREST or GL_LINEAR
GL_TEXTURE_MIN_FILTER	GL_NEAREST, GL_LINEAR, GL_NEAREST_MIPMAP_NEAREST, GL_NEAREST_MIPMAP_LINEAR, GL_LINEAR_MIPMAP_NEAREST, or GL_LINEAR_MIPMAP_LINEAR

Caveat

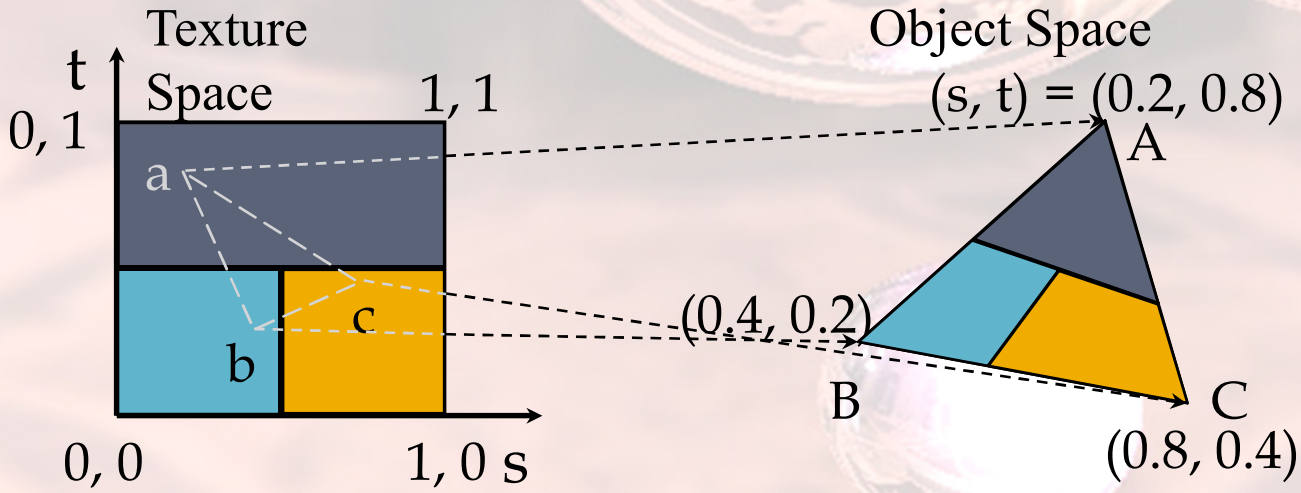
- Default setting for Minification filter is GL_NEAREST_MIPMAP_LINEAR
  - Change it to GL_LINEAR/GL_NEAREST since you don't have mipmap yet.
  - Otherwise your texture binding will fail.

Texture Mapping

- Now we have a texture object, but it contains no image.
- Define a texture image from an array of data in CPU memory
  - `glTexImage2D( target, level, components, w, h, border, format, type, *texels );`
  - dimensions of image are usually powers of 2
  - Target is usually "GL_TEXTURE_2D".
  - Variations : glTexImage1D, glTexImage3D

Texture Mapping

- Applying Texture
  - Remember to call **`glBindTexture & glEnable(GL_TEXTURE2D)`**
  - Based on parametric texture coordinates
  - **`glTexCoord*()`** specified at each vertex



Draw

```
glBindTexture(GL_TEXTURE_2D, texName);

glBegin(--------);
   glTexCoord2f(0.0, 0.0);
   glVertex3f(-2.0, -1.0, 0.0);
   …
glEnd();
```

Applications

- Environment Mapping
- Bump/Normal Mapping

Q&A