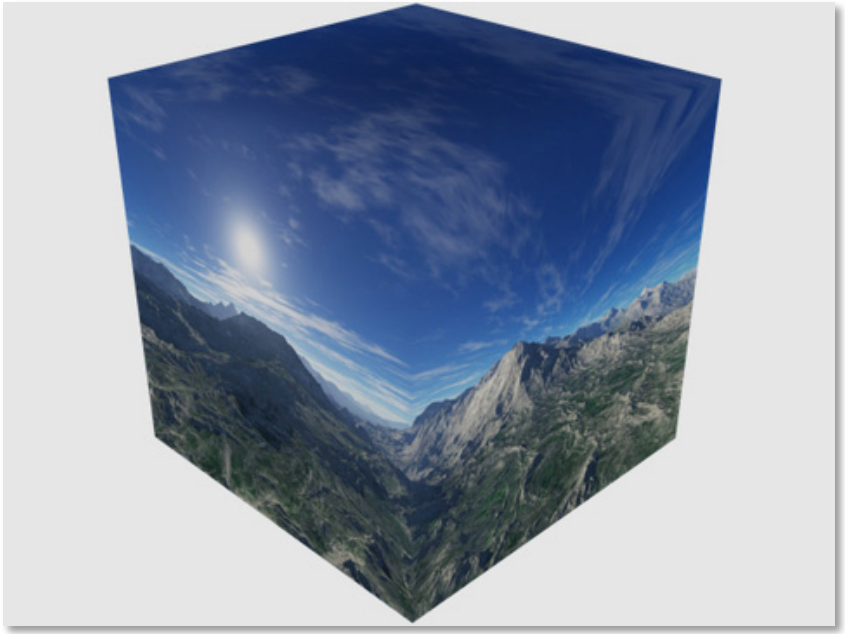
Getting fancy with texture mapping (Part 2)

CS559 – Spring 2017

6 Apr 2017

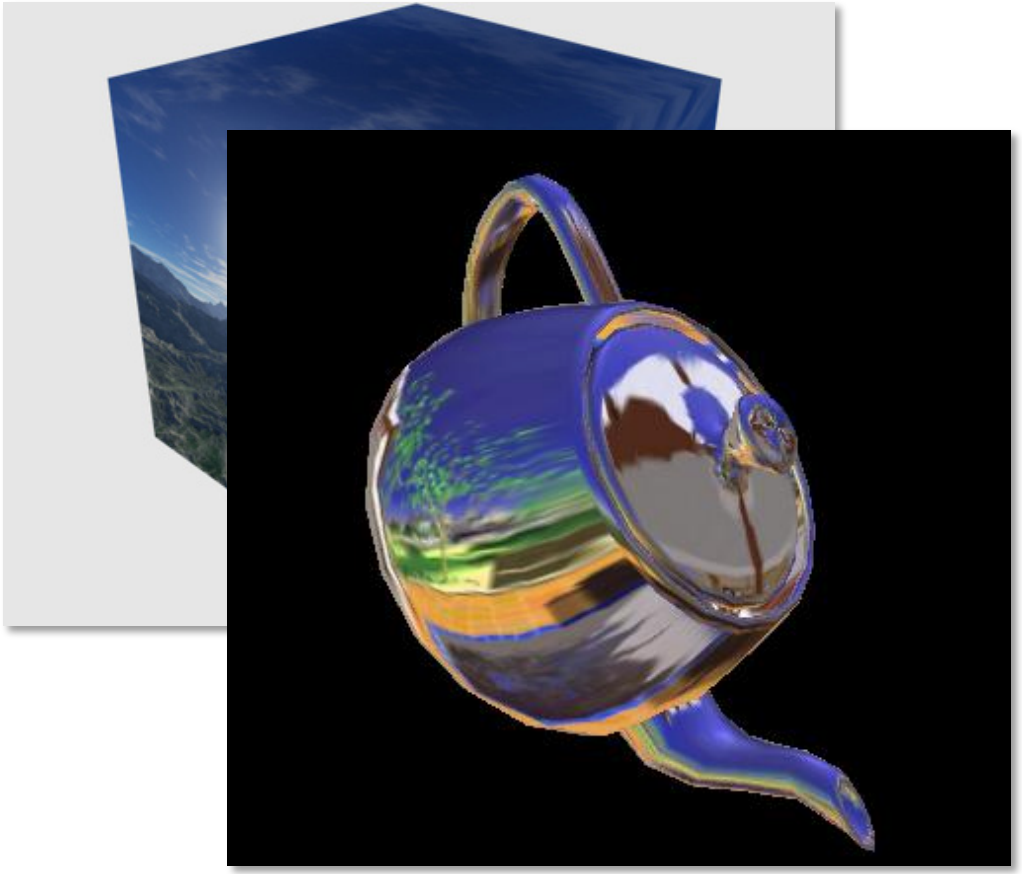
Review ...



Skyboxes as backdrops

Credits : Flipmode 3D

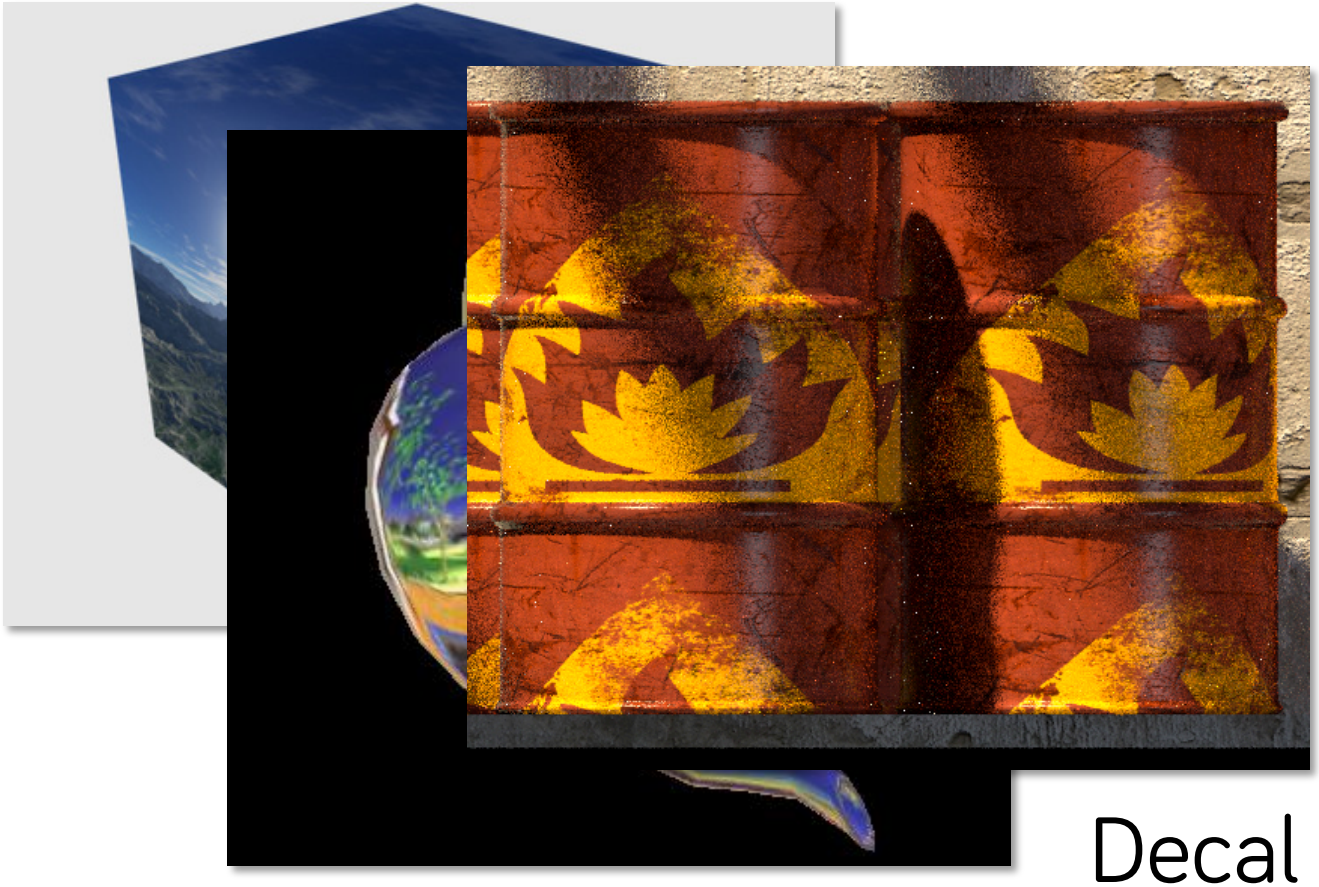
Review ...



Reflection maps

Credits : NVidia

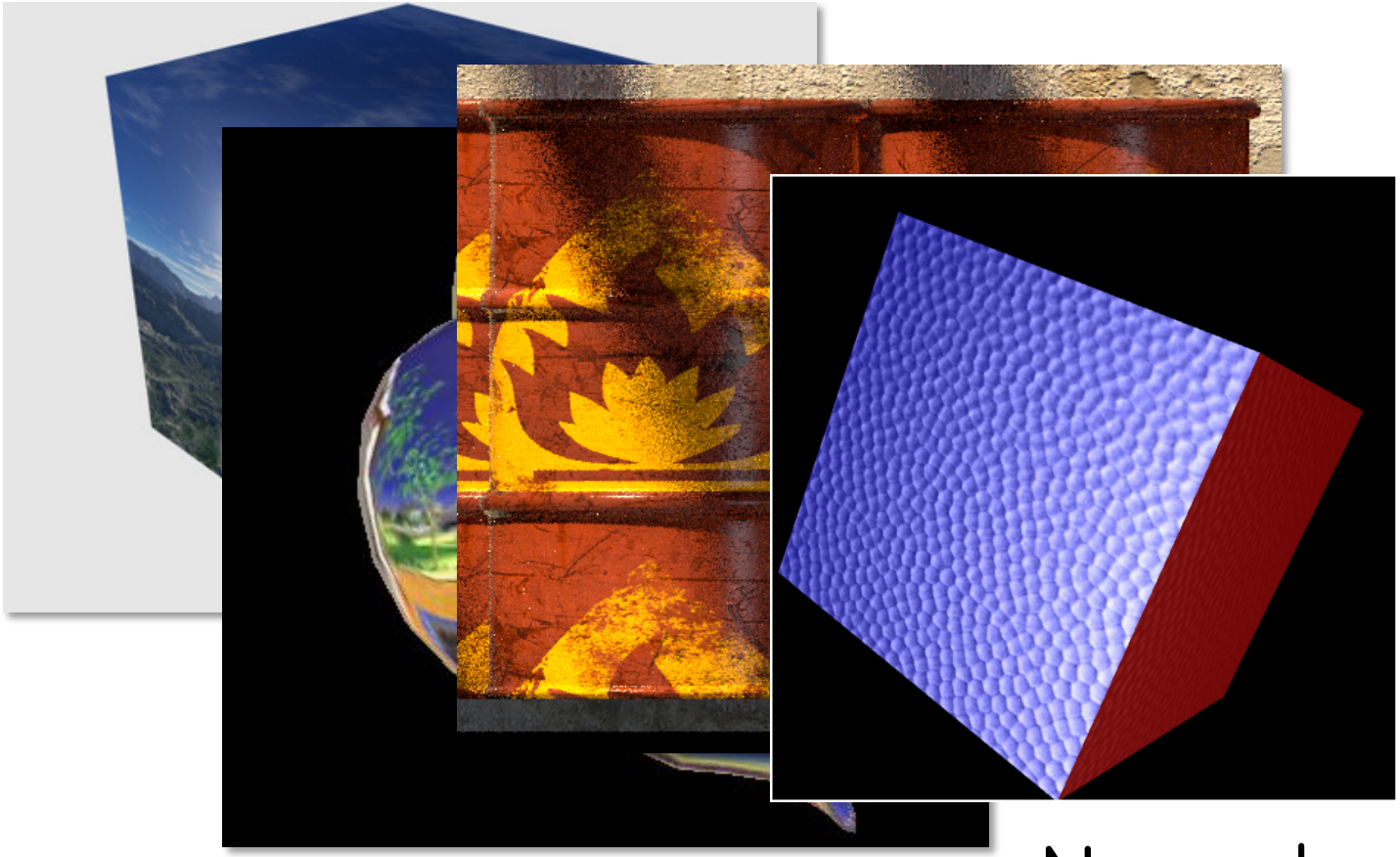
Review ...



Decal textures

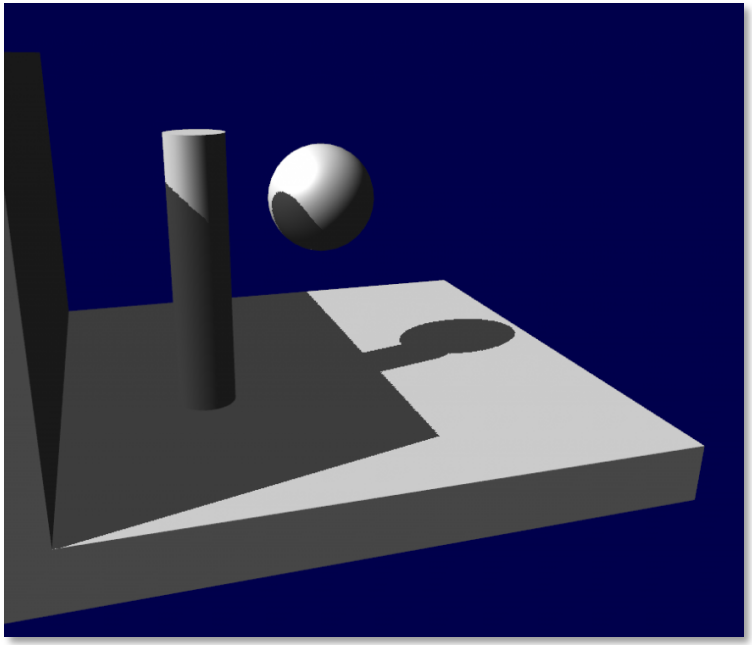
Credits : andreucabre.com

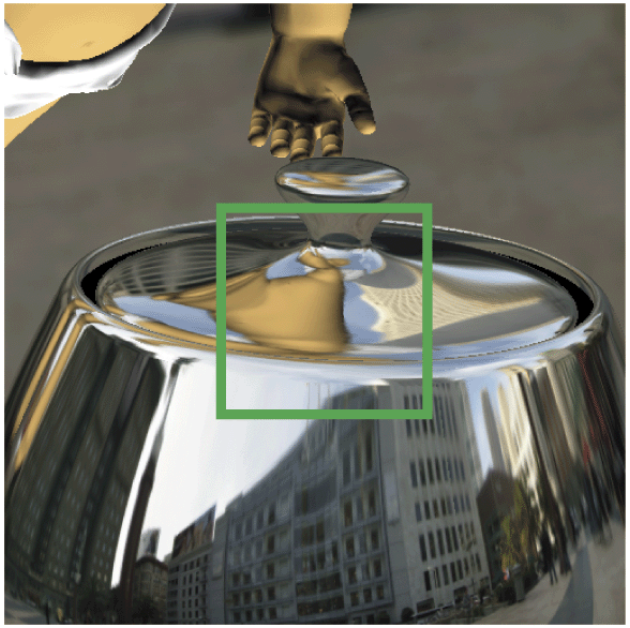
Review ...



Normal maps

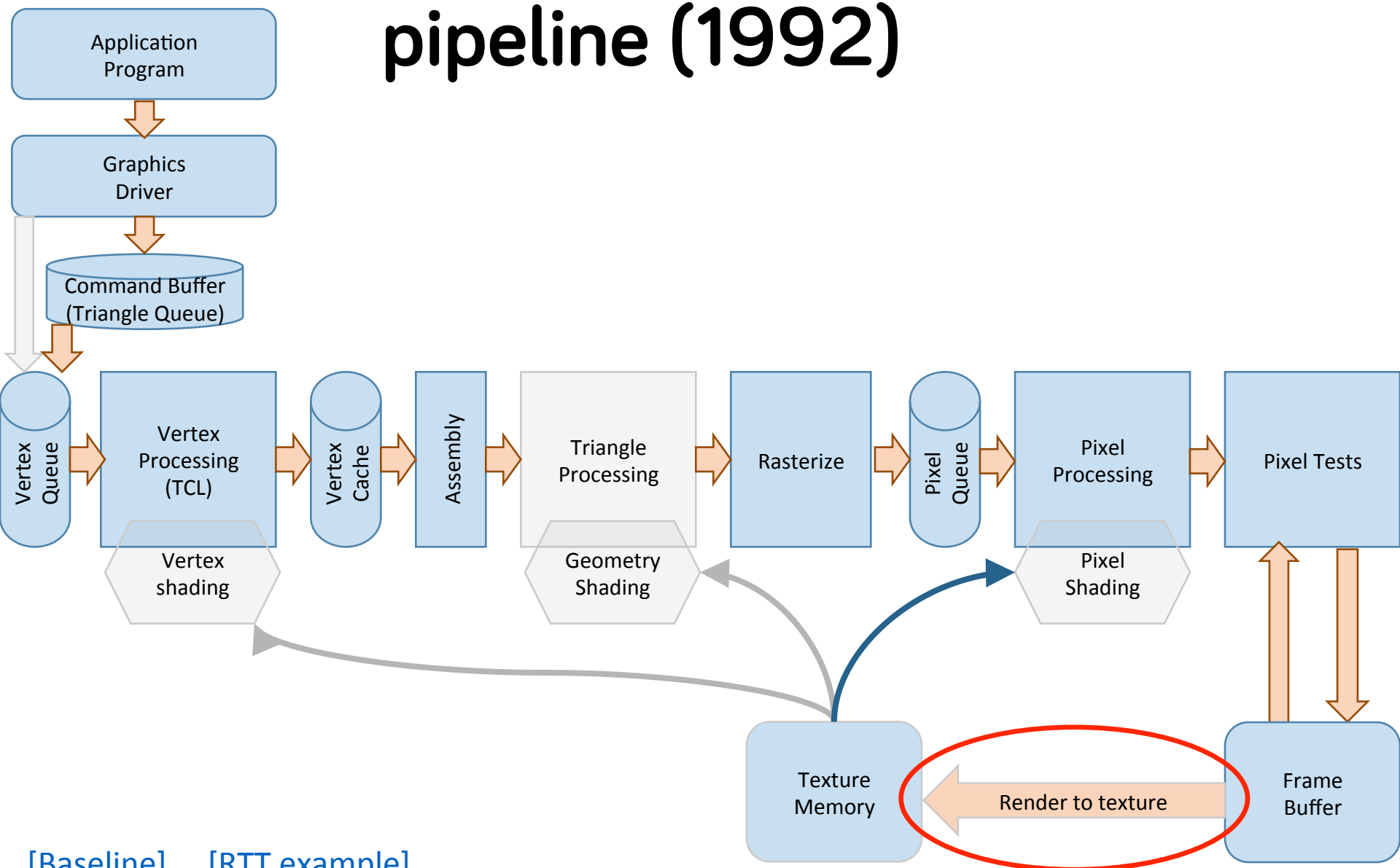
Two-pass effects ...





(b)

The full fixed-function pipeline (1992)



[Baseline] [RTT example]

jsbin.com/quxudovesu/edit

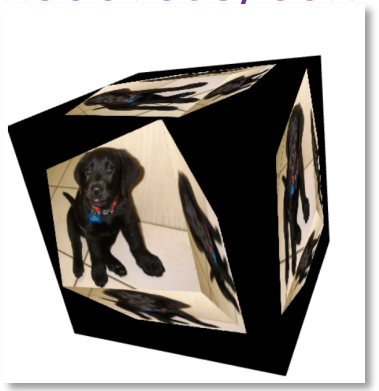
Render-to-texture

Framebuffer objects

- A destination buffer for offscreen rendering
- Separate rendering pass per framebuffer (including on-screen)
- Prior passes available to the shaders (just use the results as textures)

jsbin.com/quxudovesu/edit

Render-to-texture



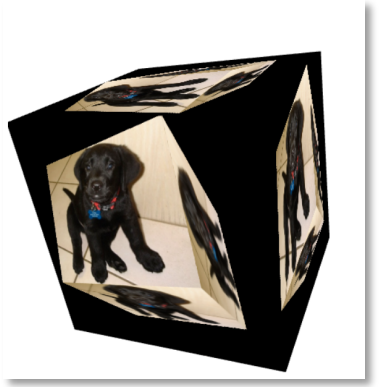
.js
```
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

```
// Set up a texture (initially blank) to bind to the framebuffer object
var FBtexture = gl.createTexture();
gl.bindTexture(gl.TEXTURE_2D, FBtexture);
gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, 512, 512, 0, gl.RGBA, gl.UNSIGNED_BYTE, null);
gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_MIN_FILTER, gl.LINEAR);

// Now create a framebuffer object and attach the texture
var framebuffer = gl.createFramebuffer();
gl.bindFramebuffer(gl.FRAMEBUFFER,framebuffer);
gl.framebufferTexture2D(gl.FRAMEBUFFER, gl.COLOR_ATTACHMENT0, gl.TEXTURE_2D, FBtexture, 0);
gl.bindFramebuffer(gl.FRAMEBUFFER,null);  // restore default FB (onscreen rendering)
```

jsbin.com/quxudovesu/edit

Render-to-texture



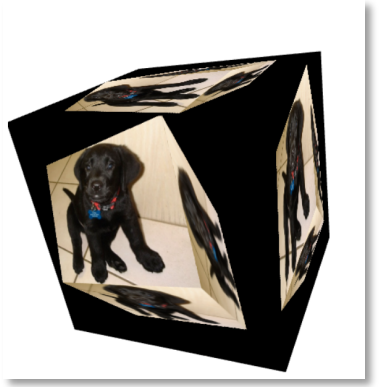
```
.js
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

```
// Set up a texture (initially blank) to bind to the framebuffer object
var FBtexture = gl.createTexture();
gl.bindTexture(gl.TEXTURE_2D, FBtexture);
gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, 512, 512, 0, gl.RGBA, gl.UNSIGNED_BYTE, null);
gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_MIN_FILTER, gl.LINEAR);

// Now create a framebuffer object and attach the texture
var framebuffer = gl.createFramebuffer();
gl.bindFramebuffer(gl.FRAMEBUFFER,framebuffer);
gl.framebufferTexture2D(gl.FRAMEBUFFER, gl.COLOR_ATTACHMENT0, gl.TEXTURE_2D, FBtexture, 0);
gl.bindFramebuffer(gl.FRAMEBUFFER,null);  // restore default FB (onscreen rendering)
```

jsbin.com/quxudovesu/edit

Render-to-texture



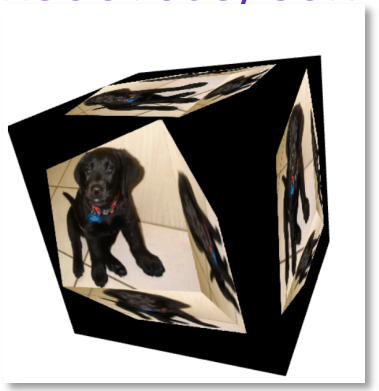
```
.js
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

```
// Set up a texture (initially blank) to bind to the framebuffer object
var FBtexture = gl.createTexture();
gl.bindTexture(gl.TEXTURE_2D, FBtexture);
gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, 512, 512, 0, gl.RGBA, gl.UNSIGNED_BYTE, null);
gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_MIN_FILTER, gl.LINEAR);

// Now create a framebuffer object and attach the texture
var framebuffer = gl.createFramebuffer();
gl.bindFramebuffer(gl.FRAMEBUFFER,framebuffer);
gl.framebufferTexture2D(gl.FRAMEBUFFER, gl.COLOR_ATTACHMENT0, gl.TEXTURE_2D, FBtexture, 0);
gl.bindFramebuffer(gl.FRAMEBUFFER,null);  // restore default FB (onscreen rendering)
```

jsbin.com/quxudovesu/edit

Render-to-texture



```
.js
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

```
// Set up a texture (initially blank) to bind to the framebuffer object
var FBtexture = gl.createTexture();
gl.bindTexture(gl.TEXTURE_2D, FBtexture);
gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, 512, 512, 0, gl.RGBA, gl.UNSIGNED_BYTE, null);
gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_MIN_FILTER, gl.LINEAR);

// Now create a framebuffer object and attach the texture
var framebuffer = gl.createFramebuffer();
gl.bindFramebuffer(gl.FRAMEBUFFER,framebuffer);
gl.framebufferTexture2D(gl.FRAMEBUFFER, gl.COLOR_ATTACHMENT0, gl.TEXTURE_2D, FBtexture, 0);
gl.bindFramebuffer(gl.FRAMEBUFFER,null);  // restore default FB (onscreen rendering)
```

jsbin.com/quxudovesu/edit

Render-to-texture

.js

```
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

```
// Set up a texture (initially blank) to bind to the framebuffer object
var FBtexture = gl.createTexture();
gl.bindTexture(gl.TEXTURE_2D, FBtexture);
gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGBA, 512, 512, 0, gl.RGBA, gl.UNSIGNED_BYTE, null);
gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_MIN_FILTER, gl.LINEAR);

// Now create a framebuffer object and attach the texture
var framebuffer = gl.createFramebuffer();
gl.bindFramebuffer(gl.FRAMEBUFFER,framebuffer);
gl.framebufferTexture2D(gl.FRAMEBUFFER, gl.COLOR_ATTACHMENT0, gl.TEXTURE_2D, FBtexture, 0);
gl.bindFramebuffer(gl.FRAMEBUFFER,null);  // restore default FB (onscreen rendering)
```

jsbin.com/quxudovesu/edit

Render-to-texture

```
gl.bindBuffer(gl.ARRAY_BUFFER, textureBuffer);
gl.vertexAttribPointer(shaderProgram.texcoordAttribute, textureBuffer.ite
  gl.FLOAT, false, 0, 0);

// ------------------------------------------------------
// PASS 1 -- DRAW TO THE FRAMEBUFFER (with original texture)
// ------------------------------------------------------

gl.bindFramebuffer(gl.FRAMEBUFFER,framebuffer);

// Clear screen, prepare for rendering
gl.clearColor(0.0, 0.0, 0.0, 1.0);
gl.disable(gl.DEPTH_TEST);
gl.clear(gl.COLOR_BUFFER_BIT);

// Bind texture
gl.bindTexture(gl.TEXTURE_2D, texture);

// Do the drawing
gl.drawElements(gl.TRIANGLES, triangleIndices.length, gl.UNSIGNED_BYTE, 0);
```

.js

```
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

jsbin.com/quxudovesu/edit

Render-to-texture

```
gl.bindBuffer(gl.ARRAY_BUFFER, textureBuffer);
gl.vertexAttribPointer(shaderProgram.texcoordAttribute, textureBuffer.ite
  gl.FLOAT, false, 0, 0);

// ------------------------------------------------------
// PASS 1 -- DRAW TO THE FRAMEBUFFER (with original texture)
// ------------------------------------------------------

gl.bindFramebuffer(gl.FRAMEBUFFER,framebuffer);

// Clear screen, prepare for rendering
gl.clearColor(0.0, 0.0, 0.0, 1.0);
gl.disable(gl.DEPTH_TEST);
gl.clear(gl.COLOR_BUFFER_BIT);

// Bind texture
gl.bindTexture(gl.TEXTURE_2D, texture);

// Do the drawing
gl.drawElements(gl.TRIANGLES, triangleIndices.length, gl.UNSIGNED_BYTE, 0);
```

.js

```
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

jsbin.com/quxudovesu/edit

Render-to-texture

```
gl.bindBuffer(gl.ARRAY_BUFFER, textureBuffer);
gl.vertexAttribPointer(shaderProgram.texcoordAttribute, textureBuffer.ite
  gl.FLOAT, false, 0, 0);

// ------------------------------------------------------
// PASS 1 -- DRAW TO THE FRAMEBUFFER (with original texture)
// ------------------------------------------------------

gl.bindFramebuffer(gl.FRAMEBUFFER,framebuffer);

// Clear screen, prepare for rendering
gl.clearColor(0.0, 0.0, 0.0, 1.0);
gl.disable(gl.DEPTH_TEST);
gl.clear(gl.COLOR_BUFFER_BIT);

// Bind texture
gl.bindTexture(gl.TEXTURE_2D, texture);

// Do the drawing
gl.drawElements(gl.TRIANGLES, triangleIndices.length, gl.UNSIGNED_BYTE, 0);
```

.js

```
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

jsbin.com/quxudovesu/edit

Render-to-texture

```
gl.bindBuffer(gl.ARRAY_BUFFER, textureBuffer);
gl.vertexAttribPointer(shaderProgram.texcoordAttribute, textureBuffer.ite
  gl.FLOAT, false, 0, 0);

// ------------------------------------------------------
// PASS 1 -- DRAW TO THE FRAMEBUFFER (with original texture)
// ------------------------------------------------------

gl.bindFramebuffer(gl.FRAMEBUFFER,framebuffer);

// Clear screen, prepare for rendering
gl.clearColor(0.0, 0.0, 0.0, 1.0);
gl.disable(gl.DEPTH_TEST);
gl.clear(gl.COLOR_BUFFER_BIT);

// Bind texture
gl.bindTexture(gl.TEXTURE_2D, texture);

// Do the drawing
gl.drawElements(gl.TRIANGLES, triangleIndices.length, gl.UNSIGNED_BYTE, 0);
```

.js

```
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

jsbin.com/quxudovesu/edit

Render-to-texture

```
// ------------------------------------------------
// PASS 2 -- DRAW ONSCREEN (with framebuffer texture)
// ------------------------------------------------

gl.bindFramebuffer(gl.FRAMEBUFFER,null);

// Clear screen, prepare for rendering
gl.clearColor(1.0, 1.0, 1.0, 1.0);
gl.enable(gl.DEPTH_TEST);
gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

// Bind texture
gl.bindTexture(gl.TEXTURE_2D, FBtexture);

// Do the drawing
gl.drawElements(gl.TRIANGLES, triangleIndices.length, gl.UNSIGNED_BYTE, 0);
```

.js

```
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

jsbin.com/quxudovesu/edit

Render-to-texture

```
// ------------------------------------------------
// PASS 2 -- DRAW ONSCREEN (with framebuffer texture)
// ------------------------------------------------

gl.bindFramebuffer(gl.FRAMEBUFFER,null);

// Clear screen, prepare for rendering
gl.clearColor(1.0, 1.0, 1.0, 1.0);
gl.enable(gl.DEPTH_TEST);
gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

// Bind texture
gl.bindTexture(gl.TEXTURE_2D, FBtexture);

// Do the drawing
gl.drawElements(gl.TRIANGLES, triangleIndices.length, gl.UNSIGNED_BYTE, 0);
```

.js

```
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

jsbin.com/quxudovesu/edit

Render-to-texture

.js
```
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

```
// ------------------------------------------------
// PASS 2 -- DRAW ONSCREEN (with framebuffer texture)
// ------------------------------------------------

gl.bindFramebuffer(gl.FRAMEBUFFER,null);

// Clear screen, prepare for rendering
gl.clearColor(1.0, 1.0, 1.0, 1.0);
gl.enable(gl.DEPTH_TEST);
gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

// Bind texture
gl.bindTexture(gl.TEXTURE_2D, FBtexture);

// Do the drawing
gl.drawElements(gl.TRIANGLES, triangleIndices.length, gl.UNSIGNED_BYTE, 0);
```

jsbin.com/quxudovesu/edit

Render-to-texture

```
// ------------------------------------------------
// PASS 2 -- DRAW ONSCREEN (with framebuffer texture)
// ------------------------------------------------

gl.bindFramebuffer(gl.FRAMEBUFFER,null);

// Clear screen, prepare for rendering
gl.clearColor(1.0, 1.0, 1.0, 1.0);
gl.enable(gl.DEPTH_TEST);
gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

// Bind texture
gl.bindTexture(gl.TEXTURE_2D, FBtexture);

// Do the drawing
gl.drawElements(gl.TRIANGLES, triangleIndices.length, gl.UNSIGNED_BYTE, 0);
```

.js

```
start(){
  initShaders();
  sendData();
  initTextures();
  draw()
}
```

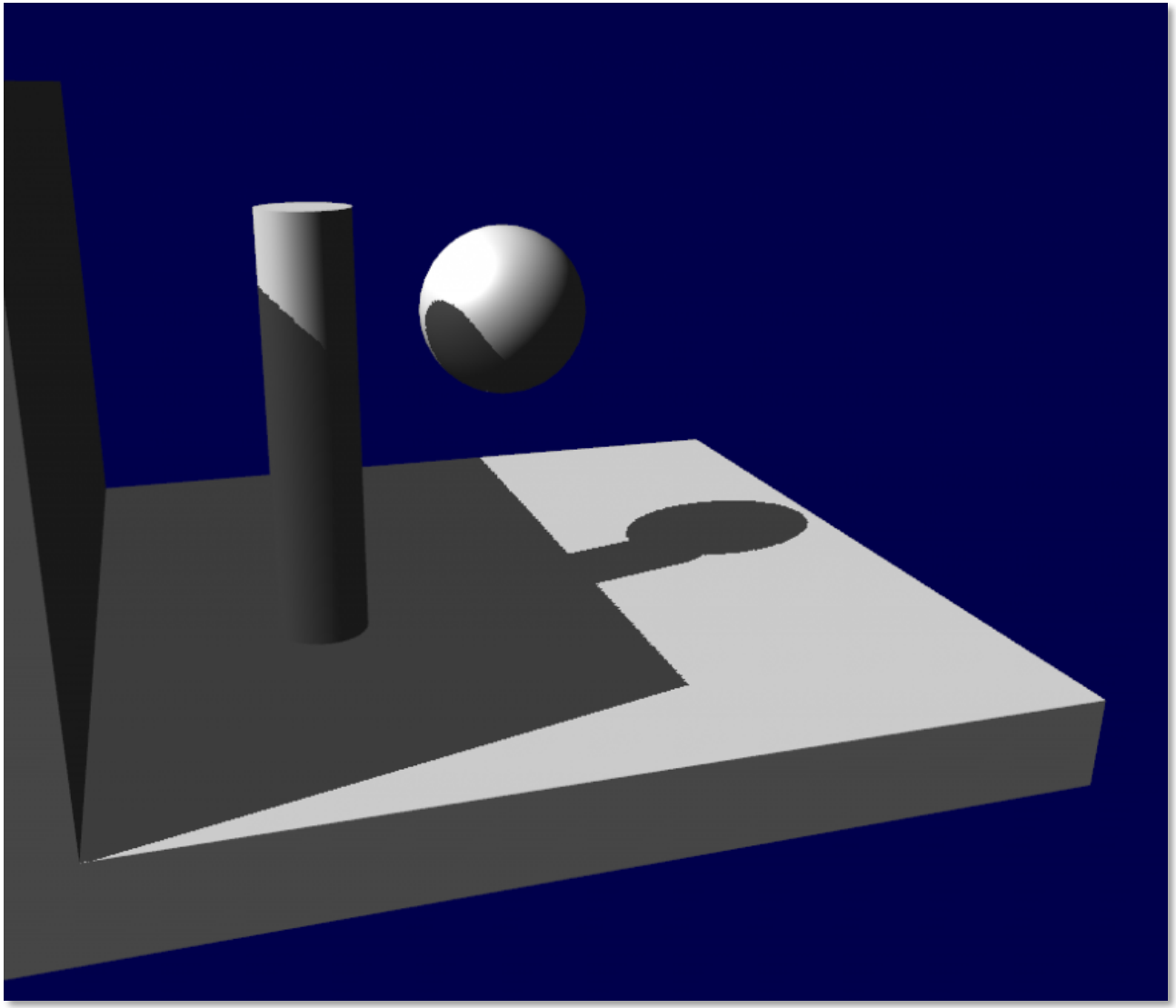
jsbin.com/quxudovesu/edit

Render-to-texture

In *real* use you may have to …

- Use different shaders per pass …
- Use depth buffer …
- Selectively draw objects …
- (and many other adjustments, depending on the algorithm implemented via RTT)

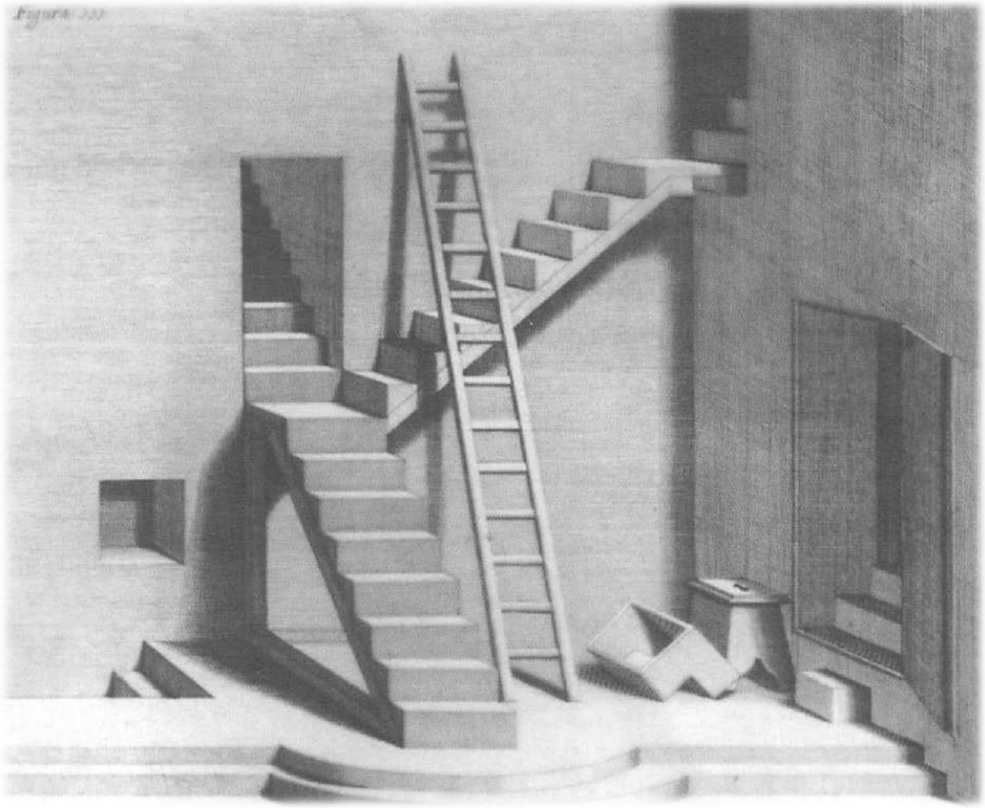
Shadow mapping



Shadow mapping

Why shadows?

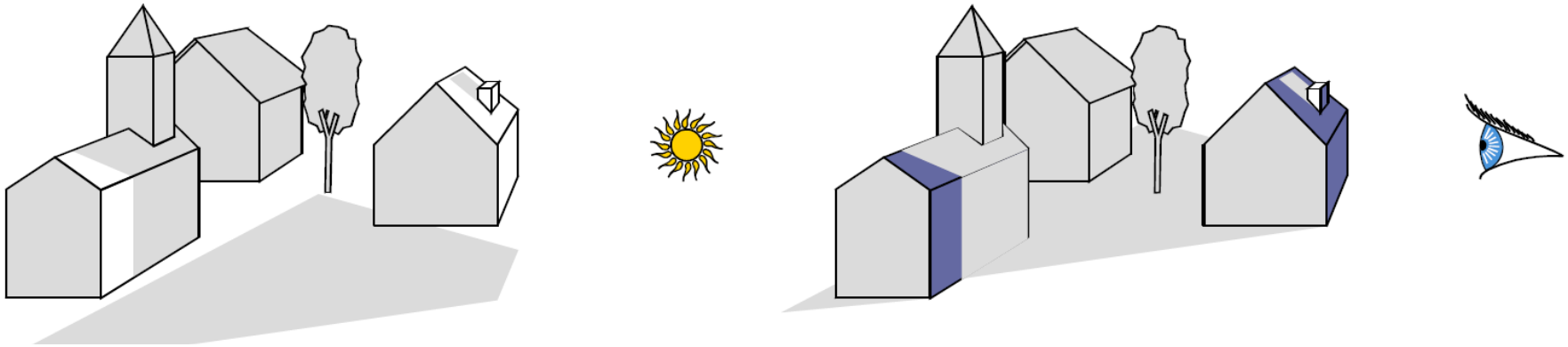
- Enrich lighting
- Provide significant cues on scene (order, depth ...)



Shadow mapping

Intuition:

Locations that are *lit* are *visible* from the light source's viewpoint

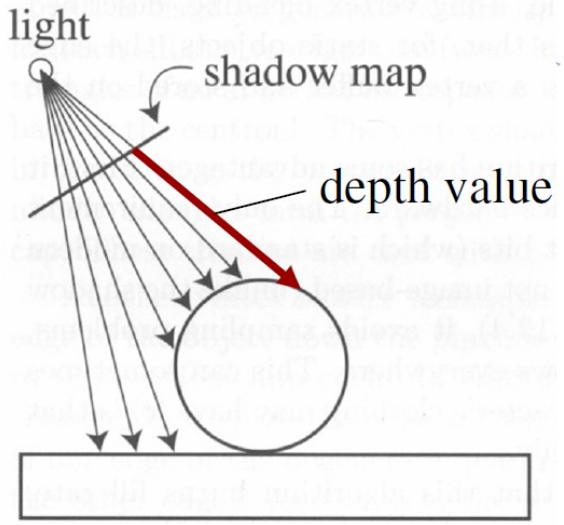


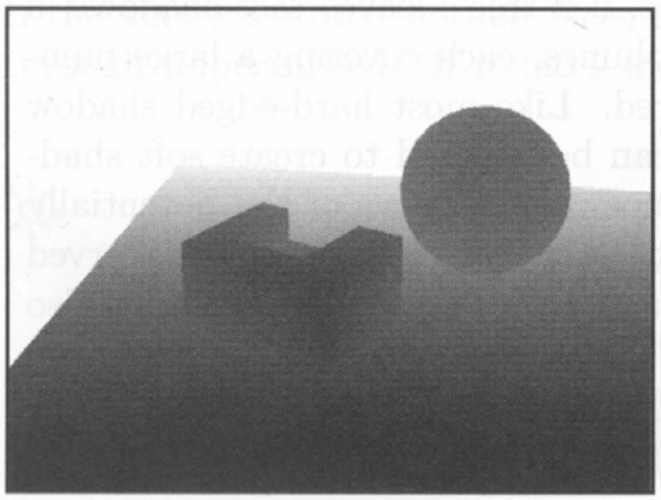
Shadow mapping

First step:

Place camera at light's location

Render & store *depth* image *(shadow map)*



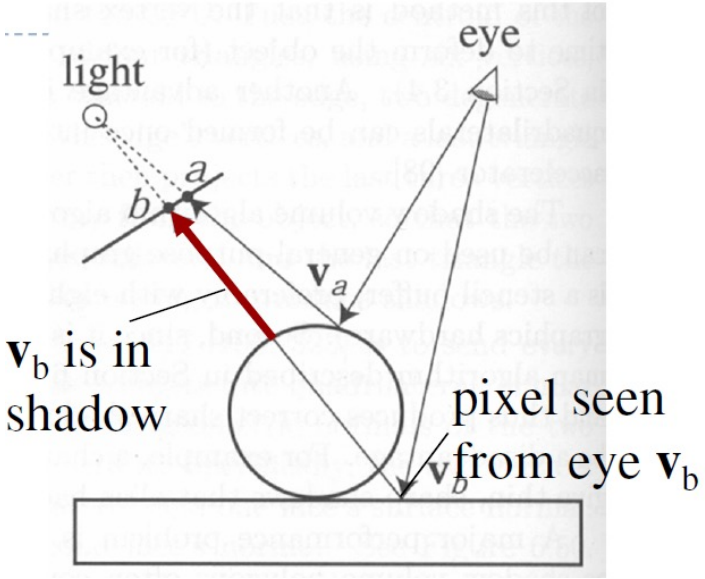


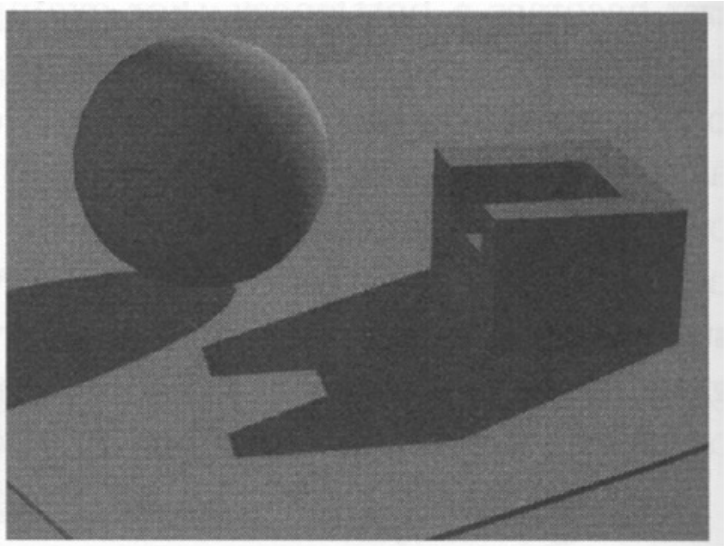
Shadow mapping

Second step:

Render from camera's position

(Per fragment) compare distance to light with shadow map value



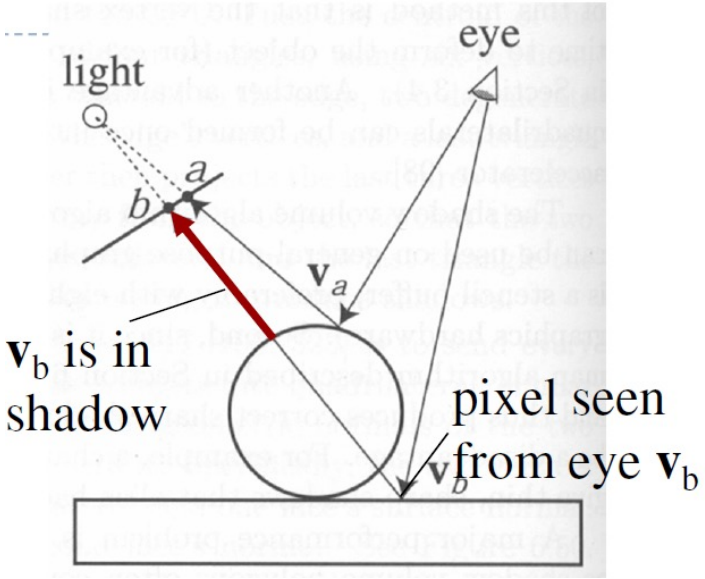


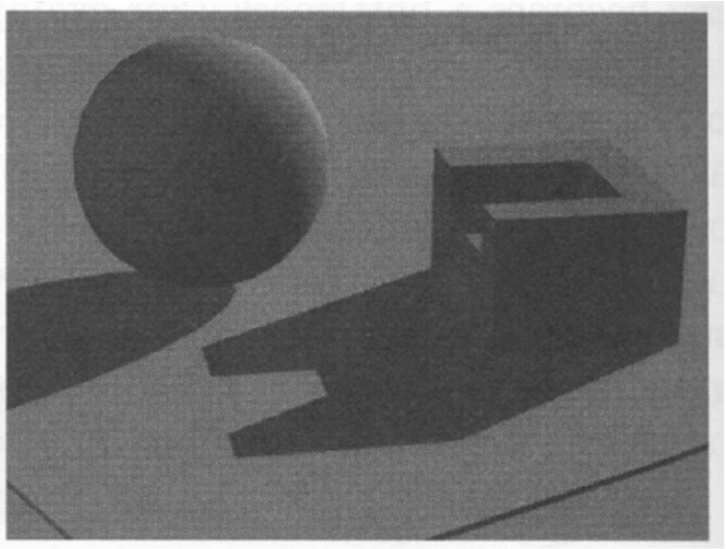
Shadow mapping

Second step:

Shadow map value larger -> Fragment shadowed

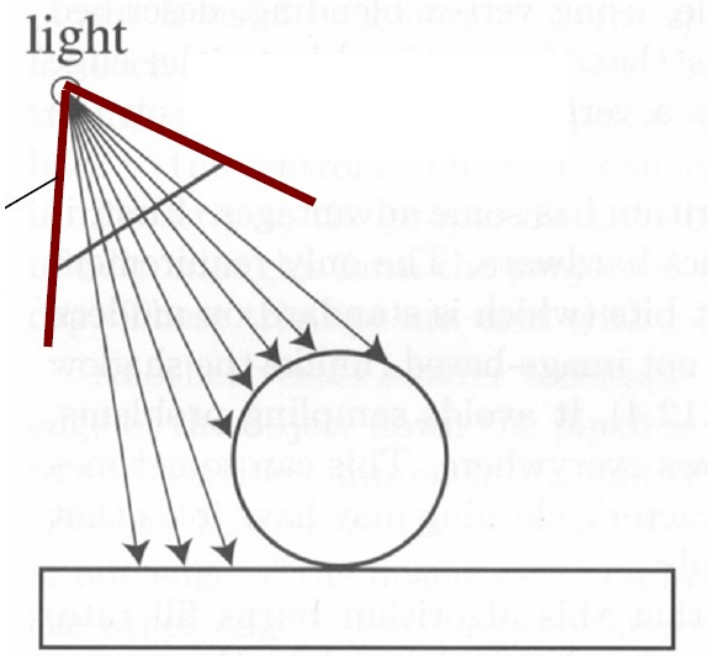
Shadow map value less/equal -> Fragment lit

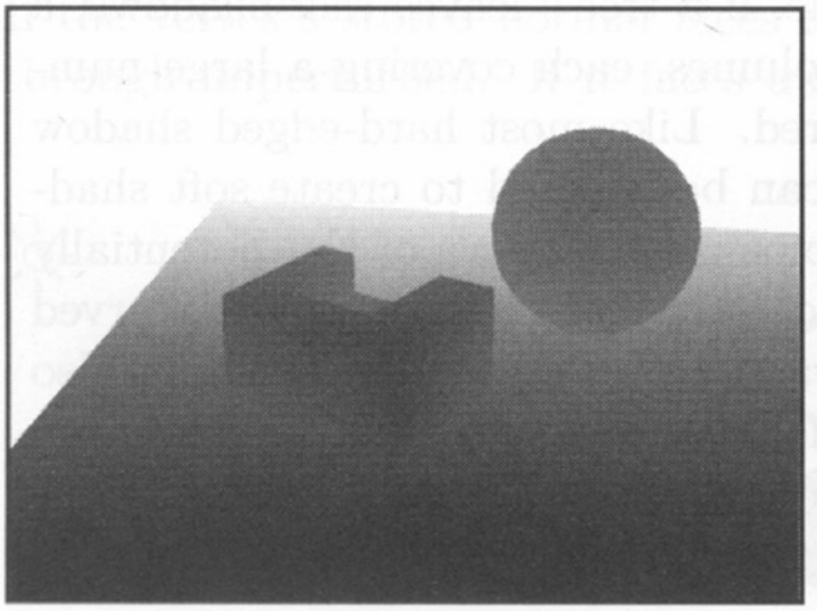




Shadow mapping

Issues : Field of view

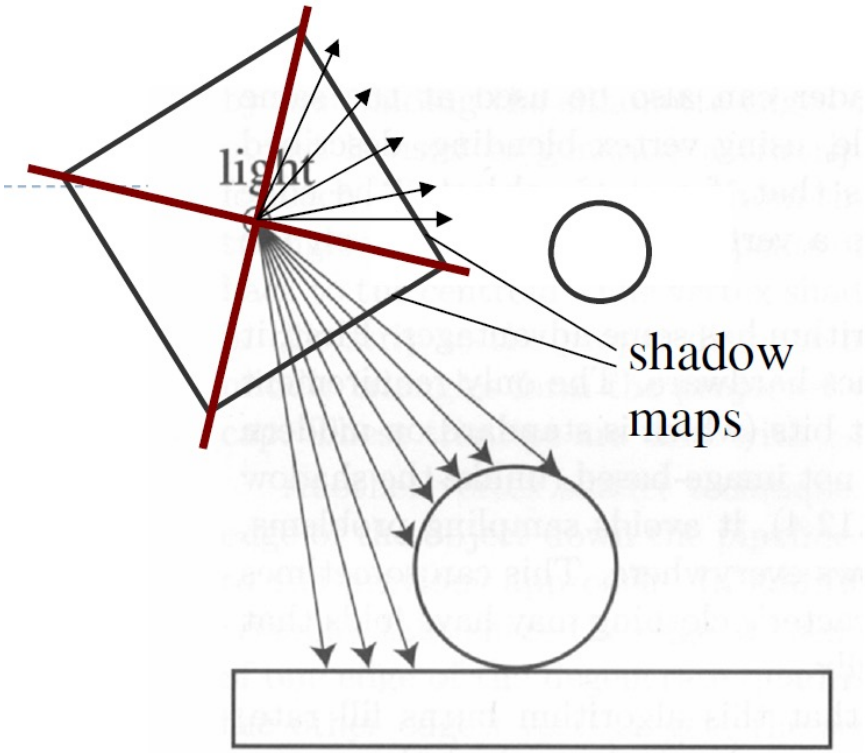




Shadow mapping

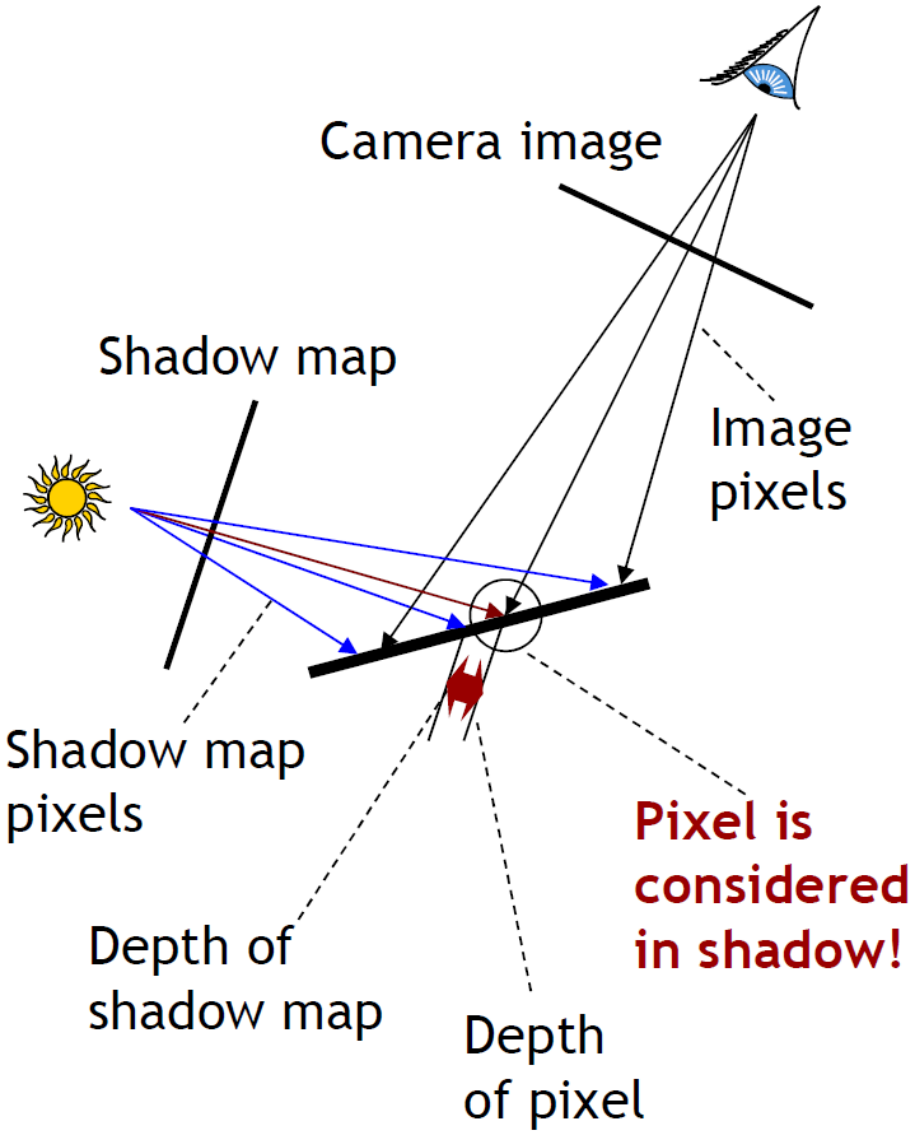
Issues : Field of view

Remedy … render 6 shadow maps in a cube



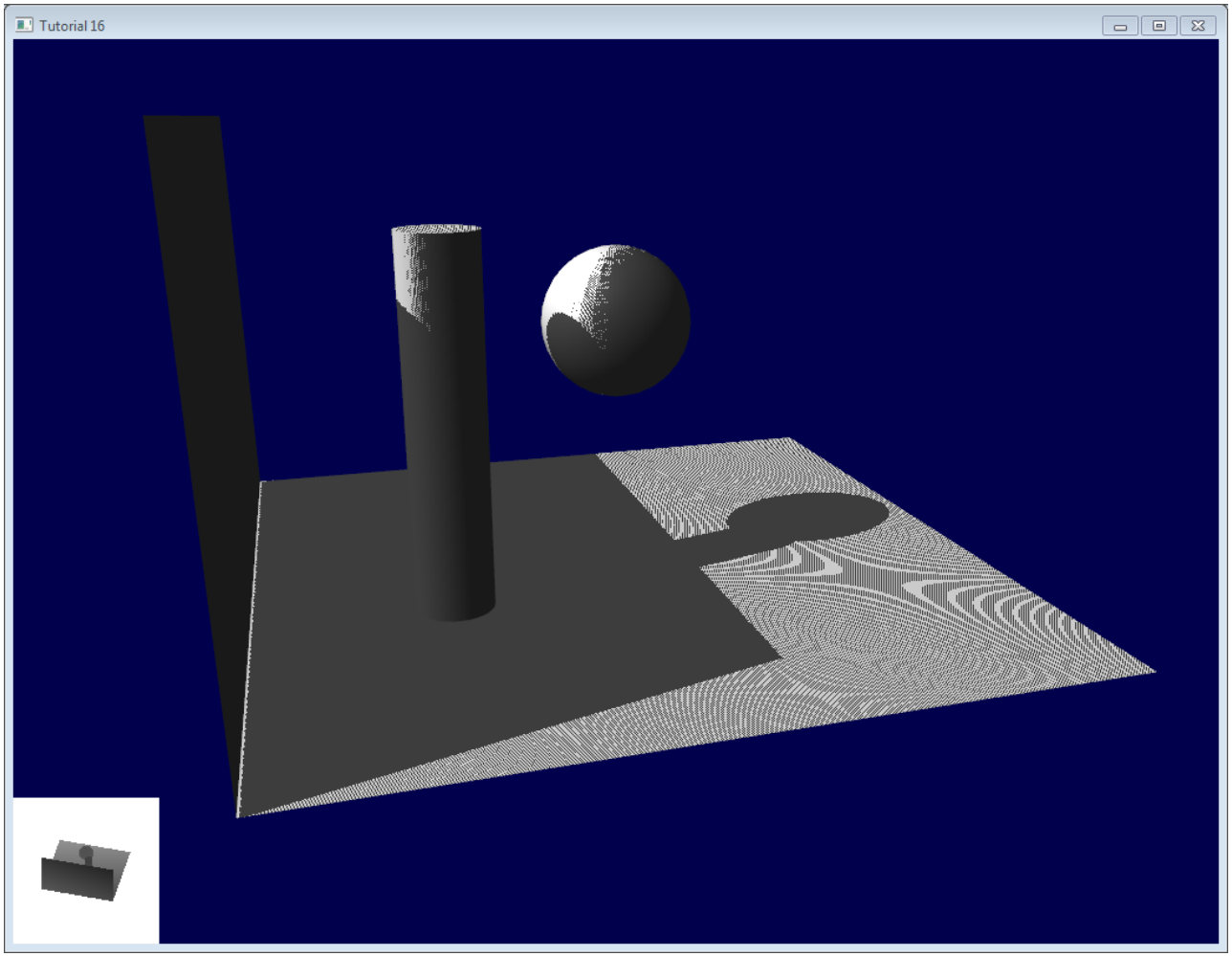
Shadow mapping

Issues : Z-fighting



Shadow mapping

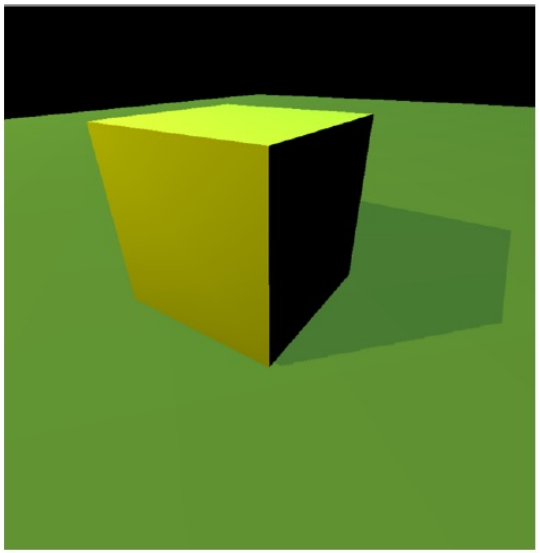
Issues : Z-fighting



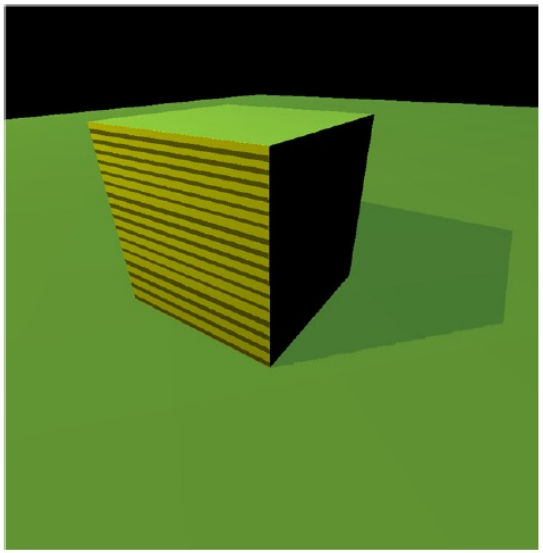
Shadow mapping

Issues : Z-fighting

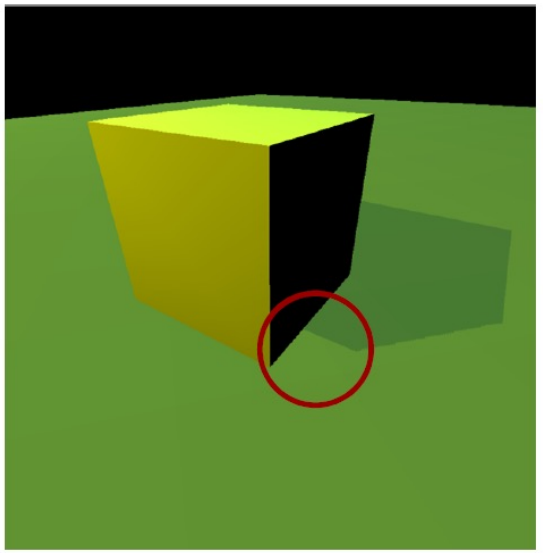
Solution : Bias (be careful how much!)



Correct bias



Not enough bias



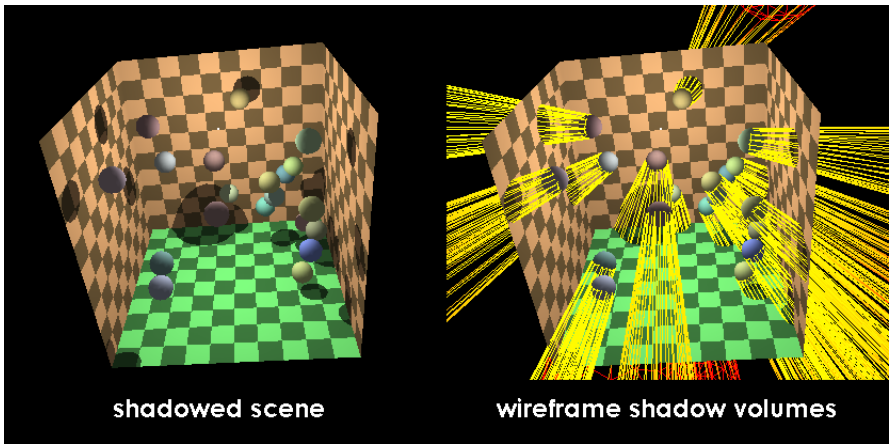
Too much bias

Shadow mapping

Additional issues:

- Sampling (resolution of shadow map)
- Thin objects
- Multiple light sources

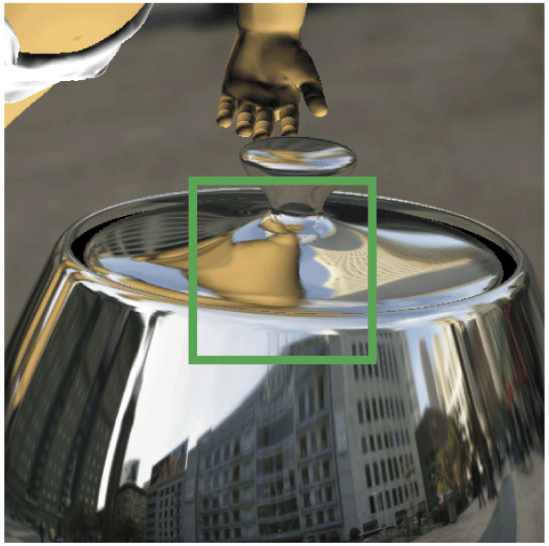
Other methods exist (e.g. shadow volumes)
(but may require CPU intervention)

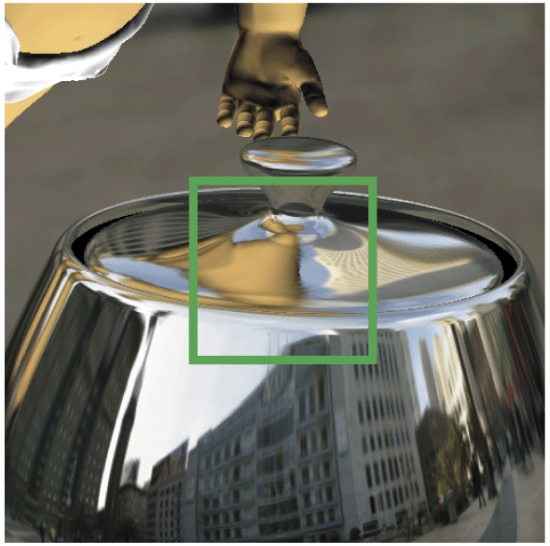


Dynamic environment maps

First pass : Render scene into the 6 faces of the environment cubemap

Second pass : Normal environment mapping with rendered texture



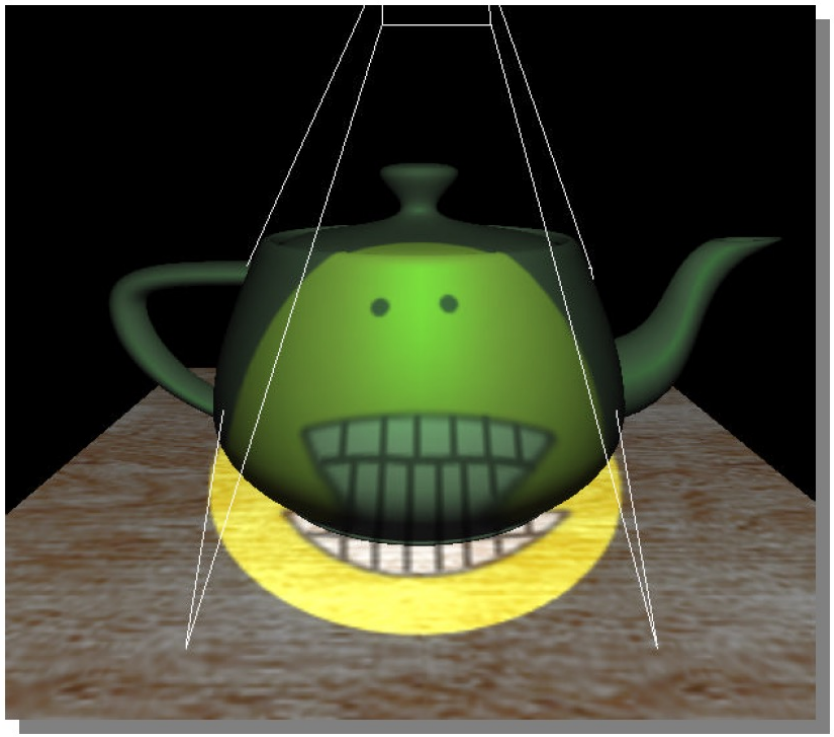


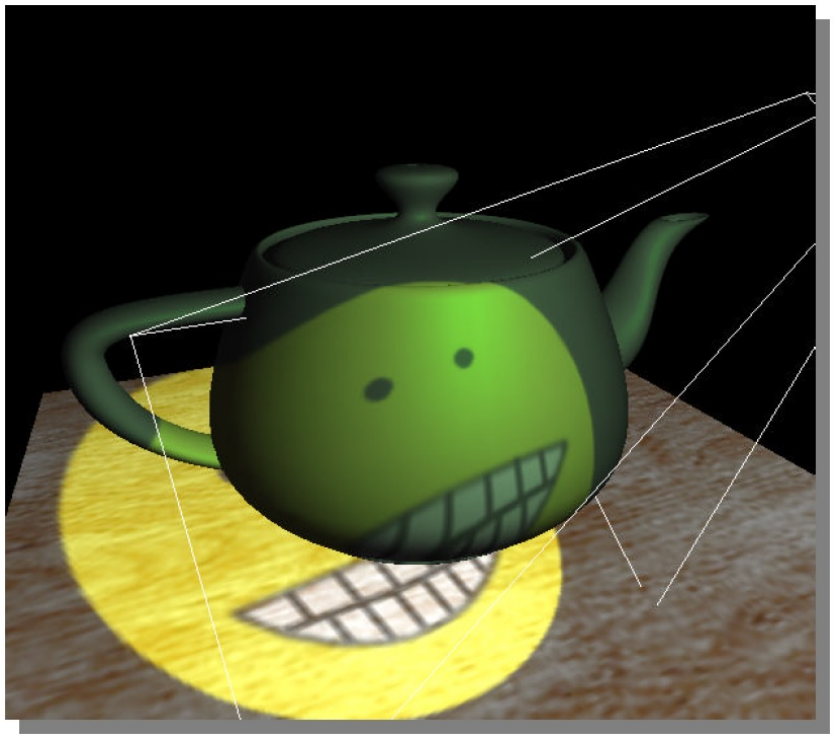
[Example] [Example]

Projector textures

"Project" texture coordinates on scene,
to use with secondary texture

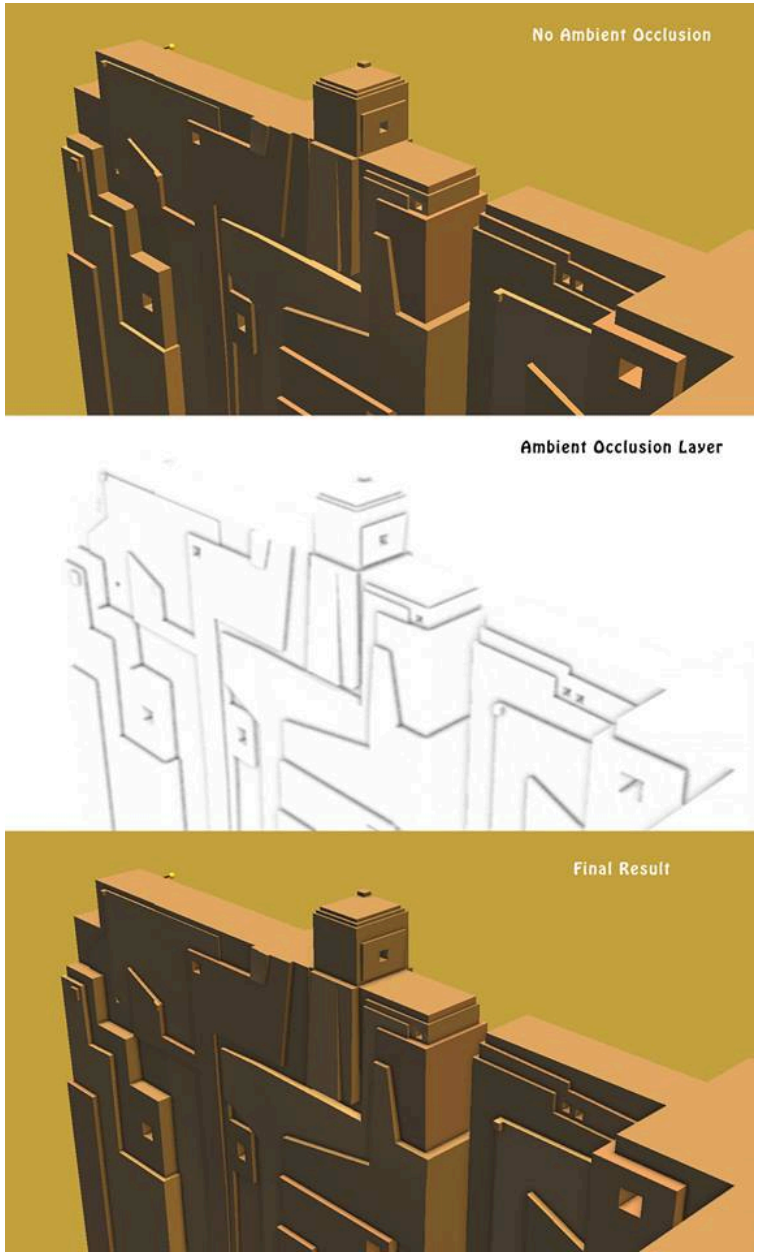
Need to worry about sampling issues all over
again (no direct mipmap help)





Ambient occlusion

Darken ambient lighting component near folds



Ambient occlusion

Can be very expensive (and very cool)

Screen-space approximation :
  Uses depth field, estimates “occlusion factor”