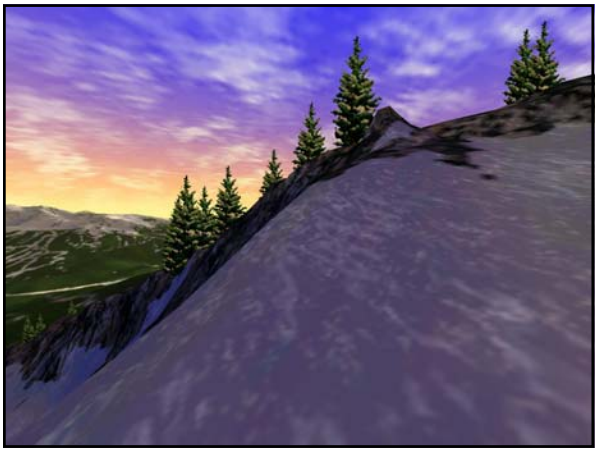
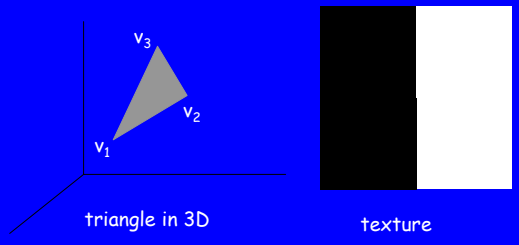
Texture Mapping & OpenGl



Textures

- Textures are cool
- Except sometimes when they're not
- Why that happens & what to do
- Other cool things to do with textures

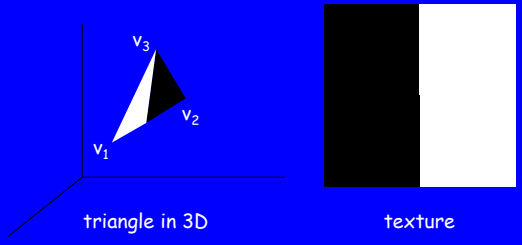
We want to "glue" the image onto the triangle.



triangle in 3D

texture

We want to "glue" the image onto the triangle.

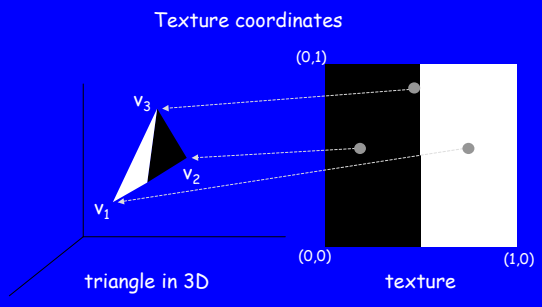


triangle in 3D

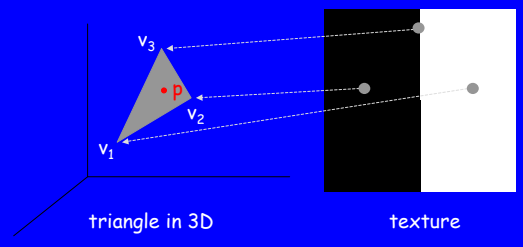
texture

texture demo

How should the texture be situated?

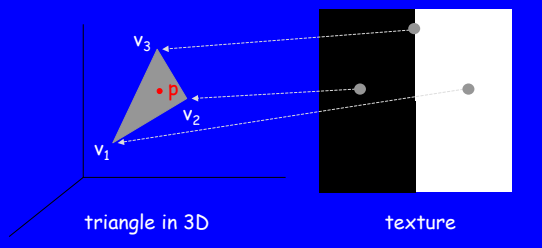


Texture Mapping Algorithm takes texture coordinates, geometric coordinates, and a point p



Texture Mapping Algorithm answers this question:

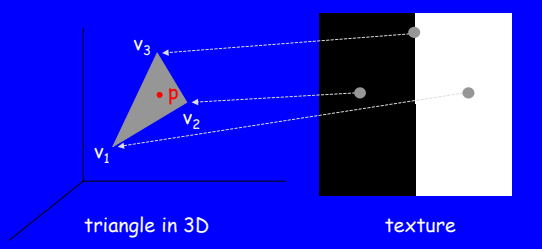
"What color is point p on the texture mapped triangle?"



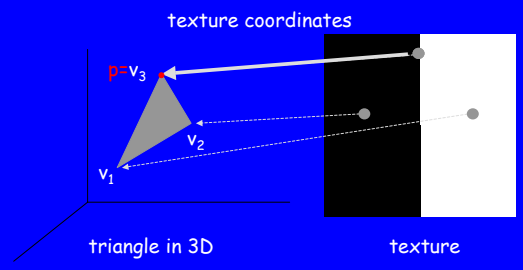
Algorithm

1. Find the texture coordinates for p.
2. Find the color of the texture at the texture coordinates for p.

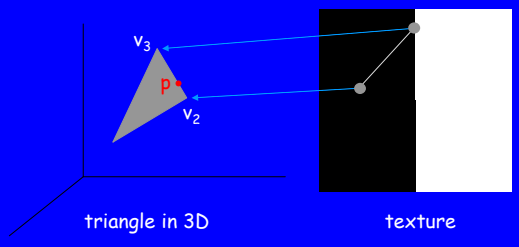
1. Find the texture coordinates for p.



If p is a vertex the answer is easy.

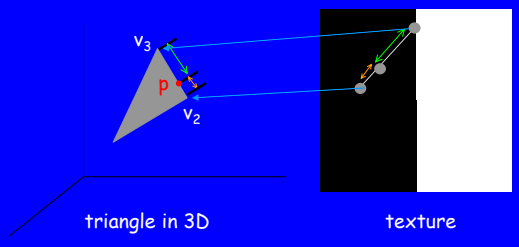


Suppose p lies on an edge ...



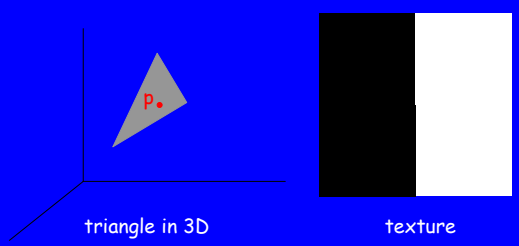
triangle in 3D texture

interpolate based on endpoints of edge



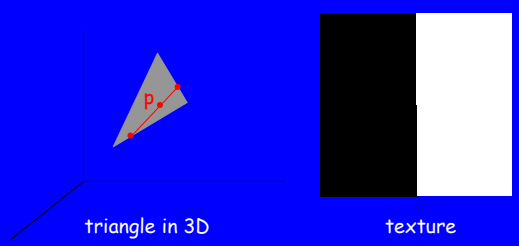
triangle in 3D texture

Suppose p is an interior point.



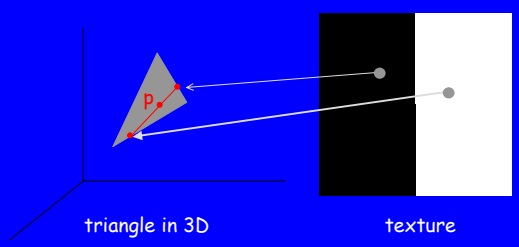
triangle in 3D texture

Draw a line through p and find its intersection points with the triangle edges.



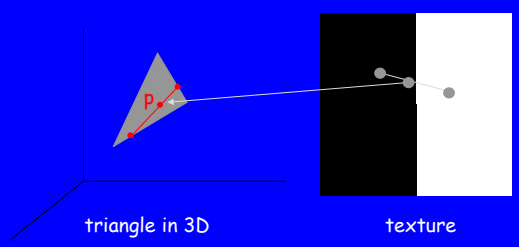
triangle in 3D texture

Compute the texture coordinates of the intersection points.



triangle in 3D texture

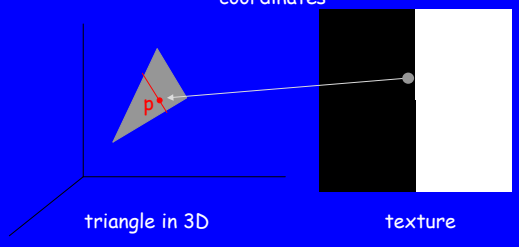
Interpolate along the line to find the texture coordinates of p.



triangle in 3D texture

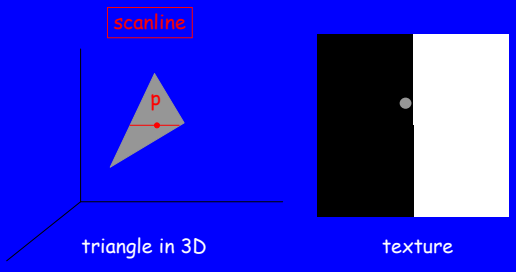
What if we had chosen a different line?

Doesn't matter … we'd end up with the same texture coordinates



triangle in 3D texture

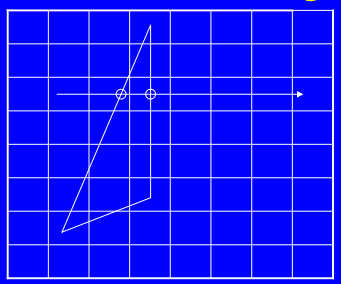
pipeline: what line should we choose?



triangle in 3D texture

where are we? 3D or 2D?

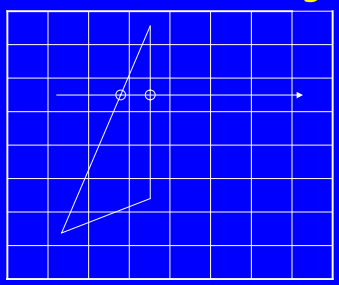
scan line algorithm



for each scan line

1. **find edge/scan line intersection points**
2. order by x-coordinate
3. use odd-even test to turn on pixels

scan line algorithm



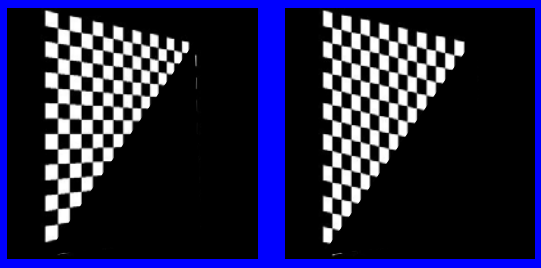
for each scan line

2D

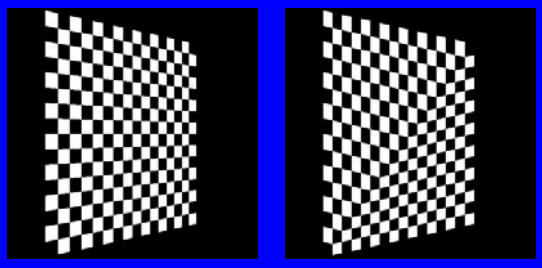
1. **find edge/scan line intersection points**
2. order by x-coordinate
3. use odd-even test to turn on pixels
4. interpolate to compute color/texture

perspective demo

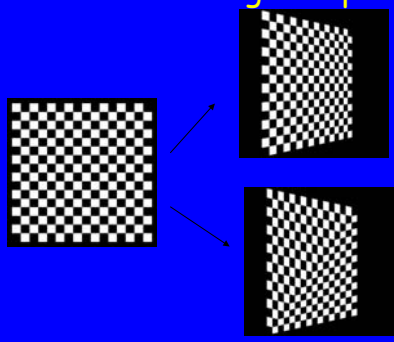
correct vs. incorrect



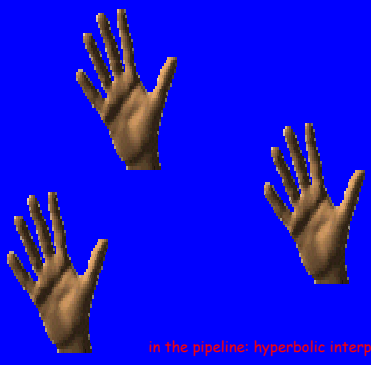
correct vs. incorrect



distortion is angle-dependent



interpolate in **3D** or 2D?

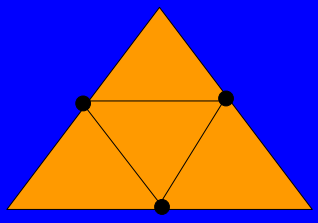


in the pipeline: hyperbolic interpolation

SO OPENGL TAKES CARE OF THIS FOR US ...RIGHT?

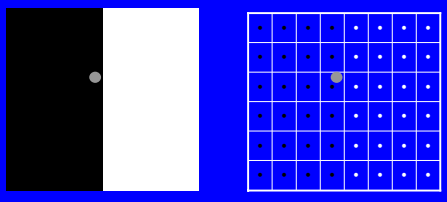
so what can you do?

increase polygon count



Algorithm

1. Find the texture coordinates for p.
2. Find the color of the texture at the texture coordinates for p.



continuous image

discrete image

resample

Algorithm

- Find the texture coordinates for p.
- Find the color of the texture at the texture coordinates for p.

Ta Da!

beyond triangles

- OpenGL typically converts general polygons into triangle fans or triangle strips.

blending

Texel color = pixel color
= ambient response
= diffuse response

etc.

opengl howto: woo ch 9

- create a texture object
- specify resampling method, etc.
- specify the texture
- enable texture mapping
- draw scene supplying texture coordinates for each vertex

opengl howto: woo ch 9

- create a texture object

```
unsigned int hTexture;
glGenTextures(1, &hTexture);
glBindTexture(GL_TEXTURE_2D, hTexture);
```

opengl howto: woo ch 9

- specify resampling method, etc.

```
glTexParameteri(GL_TEXTURE_2D, GL_MIN_FILTER, GL_LINEAR);
glTexParameteri(GL_TEXTURE_2D,GL_MAX_FILTER, GL_NEAREST);
```

opengl howto: woo ch 9

- specify the texture
  - read texture
  - height and width must be a power of 2

```
glTexImage2D(..., image)
```

opengl howto: woo ch 9

- enable texture mapping
- draw scene

```
glEnable(GL_TEXTURE_2D);
glTexEnvi(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_REPLACE);
glBegin(GL_TRIANGLES);
glTexCoord2f(0,0);
glVertex3f(0,0,0);
glTexCoord2f(1,0);
glVertex3f(10,0,0);
glTexCoord2f(.5,1);
glVertex3fr(5,5,0);
glEnd();
```

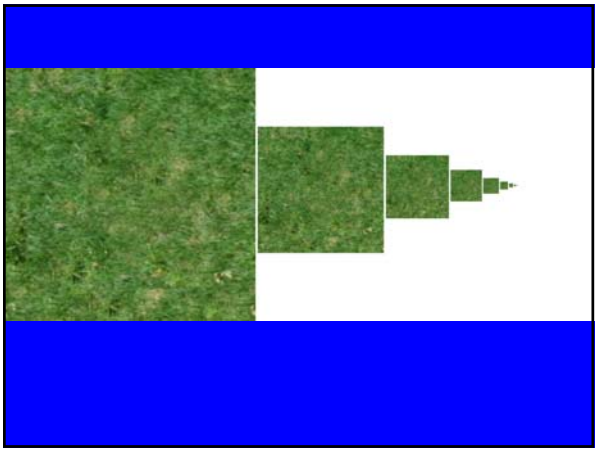
interpolation demo

- openGL supports nearest and bilinear interpolation
- can choose different interpolation method for minification and magnification of textures

ideal interpolation

neighborhood size increases with distance from the viewer

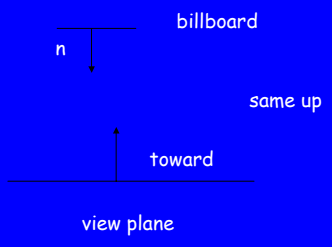
mipmapping



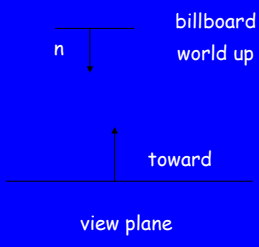
billboarding

use picture
of complicated
object instead of complicated geometry

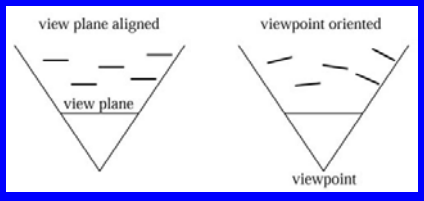
screen aligned billboard

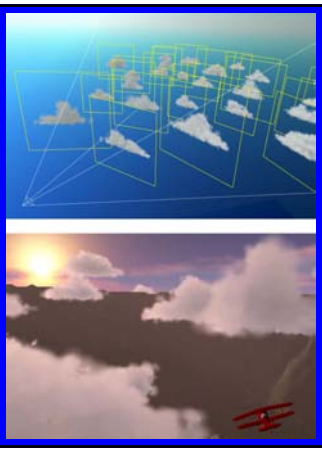


world oriented billboard



viewplane vs. viewpoint aligned





exercise

determine the rotation transforms for each of these billboarding techniques

LAB TONIGHT: Implement