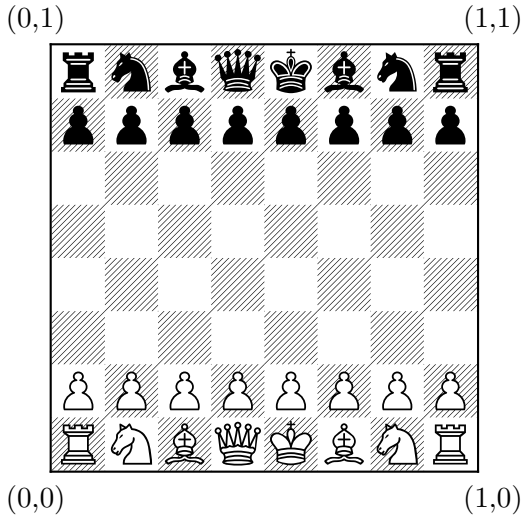
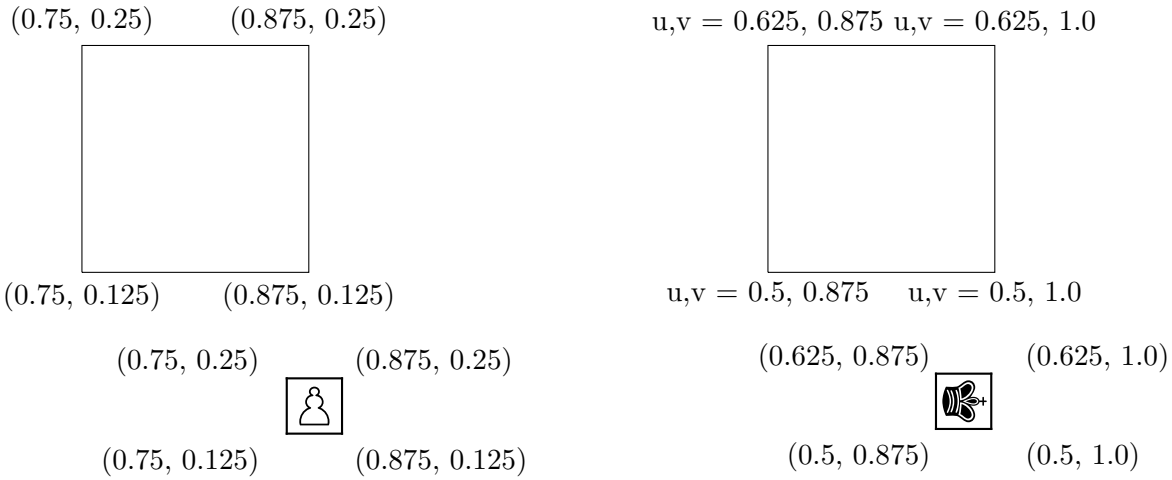
Homework 3 - Textures
Solution

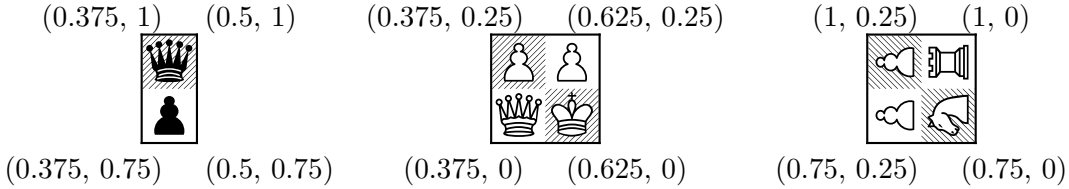
1. **Considering the following chessboard texture with (u,v) coordinates:**



(a) **Draw the approximate mapping on each quad if they were textured using the above image.**



(b) **Label the (u,v) texture coordinates for each of the 4 vertices that would result in the image below (just write them next to the corner of each square).**



2. **Describe how and when one could use Spherical Parameterization to map a texture onto a 3D model.**

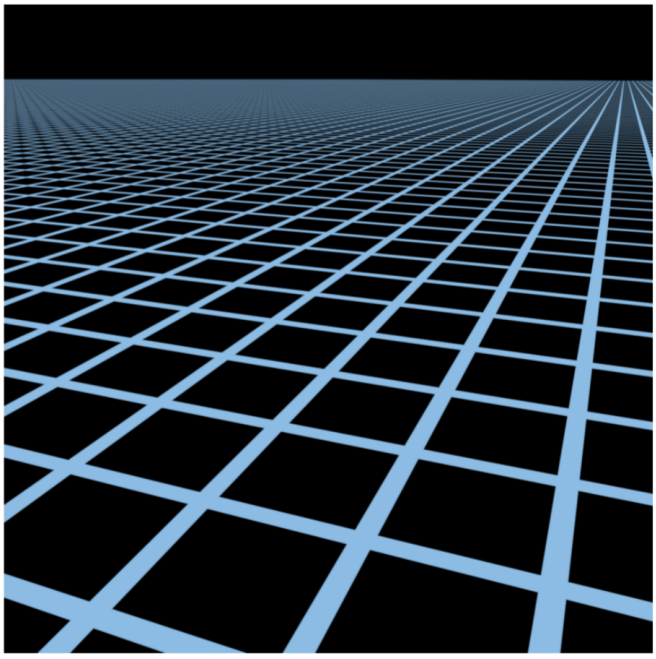
   You can apply a spherical parameterization by converting each vertex (x, y, z) position into polar (ρ, θ, r). Then just assign the UV coordinates to each vertex as (U,V)= (ρ, θ). The question is whether you would want to. This might make sense to texture a sphere. It would make less sense to texture a torus since two points on the torus would have the same (ρ, θ) coordinates and thus have to have the same color. On the other hand, the equation for a torus does in fact have two parameters, so you could use that equation to assign UV. The situation is harder with teapots and bunnies since no simple mathematical parameterization exists.

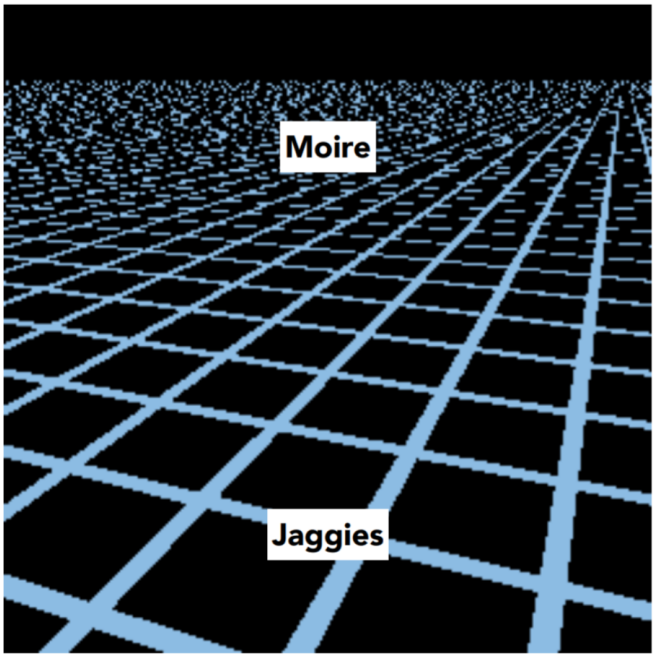
3. **In texture mapping, what is the difference between magnification and minification?**

   Magnification is when the texture is being zoomed in. Each texel (texture pixel) in the texture map is needed for multiple screen pixels. Equivalently, each screen pixel is mapping to less than a single texel.

   Minification is when a texel is being mapped to less than a single screen pixel. Equivalently, each screen pixel is mapping to many texels (and rendering without aliasing requires figuring out an average of all those texels).

4. **We textured a floor in our world and wish it looked like the left image, instead we got the right image, with speckled Moire patterns in the back and Jaggies in the front.**





   (a) **Explain why these Moire patterns appear and how to fix them.**

   These Moire patterns are a form of aliasing. Aliasing results whenever there are insufficient samples to reproduce a high frequency signal (whenever the image switches color rapidly), in this case the lines. We can increase the sampling rate using anti-aliasing (also known as super sampling with perhaps 4x4 samples per pixel) or we can pre-filter the texture (average together a bunch of texels (texture pixel) producing a blurred image, so there is no high frequencies the texture). OpenGL has mipmapping built in, which is one

method of pre-filtering, so we could turn that option on by setting the minification filter to LINEAR_MIPMAP_LINEAR. We would also need to create the mipmaps.

(b) **Explain why these Jaggies patterns appear and how to fix them.**

Jaggies are a result of insufficient resolution in our texture map relative to the screen rendering. To get rid of the jaggies, bilinear interpolation must be turned on by setting the magnification filter to LINEAR.

5. **What is the following fragment shader doing?**

```
varying vec2 uv;

uniform sampler2D tex;
uniform vec4 baseColor;

void main() {
    vec4 texColor = texture2D(tex, uv);
    gl_FragColor = 0.5 * texColor + 0.5 * baseColor;
}
```

This fragment shader is assigning the pixel color as the average of a color and a texture. The color is coming from the uniform variable baseColor, and the texture value is being retrieved with the call texture2D(tex,uv).