Beginner

OpenGL ES & GLKit

Hands-On Challenges

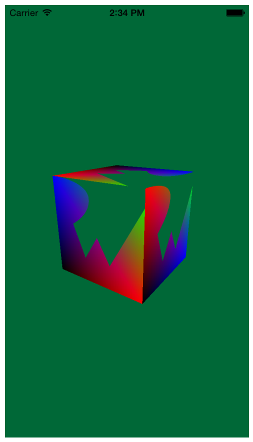
Beginner OpenGL ES & GLKit Hands-On Challenges

Challenge #6: Textures and Masks

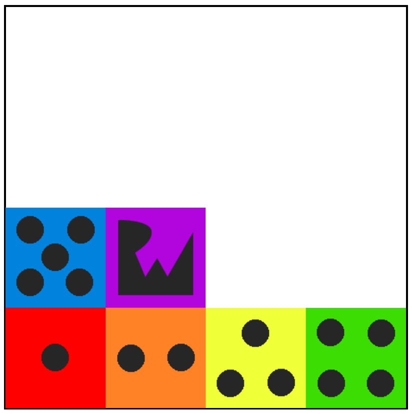
In the lecture, you saw how to map a texture to each side of a cube. But there's a lot more you can do with texturing than that – and in this lab you'll explore two of the most useful techniques.

Part 1: Rolling the Dice

In the resources for this challenge, you will find a starter project. This is a project that is equivalent to where things left off in the lecture, with a textured 3D cube drawn to the screen. Look through the project and make sure you have a good understanding of how it works.



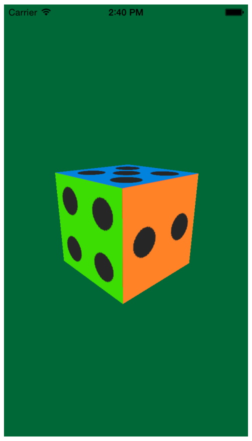
The starter project also includes a texture you'll need: **dice.png**. Open it up to take a look:



Your job in this challenge is to modify **RWCube.m** to use this texture, such that eac face of the cube is mapped to one of the parts of this image. Also, modify your vertex shader to use just the texture color (instead of also multiplying by the vertex color):

```
gl_FragColor = texture2D(u_Texture, frag_TexCoord);
```

When you're done, you should have the following:

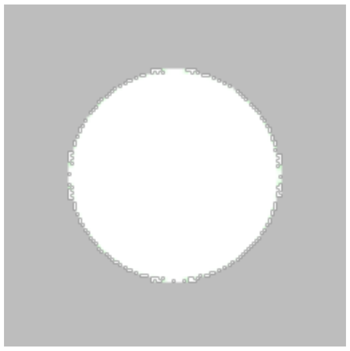


Uber Haxx0r Challenge: Masking Nodes

One of the most useful techniques in games is masking an image with another image. For example, say you have this zombie image:

And you want to put it in this frame:

To do that, you want to make the parts of the zombie image show up that are inside the frame, but not the parts that are outside the frame. You can accomplish this by applying a mask to the zombie image. For example, say you have an image that defines the center of the picture frame:



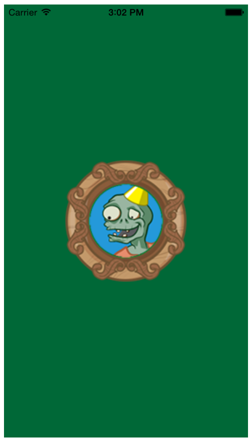
You color the parts where you want the zombie to show up white, and make the others transparent. You could pass this texture in to your fragment shader as a second texture:

```
uniform sampler2D u_Mask;
```

And then your fragment shader's main would look something like this:

```
void main(void) {
  lowp vec4 texColor = texture2D(u_Texture, frag_TexCoord);
  lowp vec4 maskColor = texture2D(u_Mask, frag_TexCoord);
  gl_FragColor = vec4(texColor.r, texColor.g, texColor.b,
maskColor.a * texColor.a);
}
```

Your challenge is to modify your shaders to add masking support, and a new "Masked Texture" model object that uses them in order to generate the following image to the screen:



Hint: If you're having trouble making the frame show up in front of the zombie, try setting the z position of the frame slightly bigger than the z position of the zombie.

This is one of the trickiest challenges yet, so if you get this working you can rest assured that you have mastered the material covered so far! ☺