Beginner

OpenGL ES & GLKit

Hands-On Challenges

Beginner OpenGL ES & GLKit Hands-On Challenges

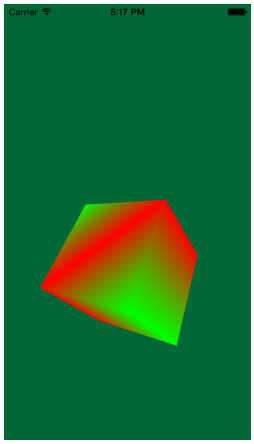
Challenge #5: Out of this World

At this point, you have seen how to generate geometry for a 3D cube and render it to the screen using 3D transformations.

But surely you can do a bit better than a cube – what about making a rocket ship and watching it blast off into space?

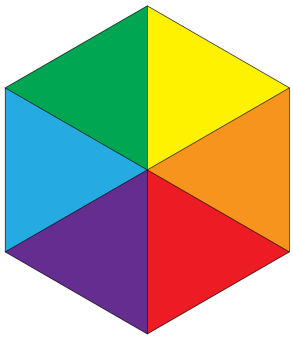
Part 1: Making a Cone

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 3D cube drawn to the screen. Look through the project and make sure you have a good understanding of how it works.



You are going to start by creating a new model which will be a rocket ship. To keep things simple, you will make a cone that you will stretch later into a rocket ship shape.

The cone you are trying to make will look like this (top down view):



The bottom center of the cone should be at (0, 0, 0), and the cone should be 1 unit tall and 2 units wide. You might find it helpful to sketch out the coordinates for each vertex of the cone – you'll need it later.

To do this, go to **File\New\File** and select the **iOS\Cocoa Touch\Objective-C class** template. Name the class **RWTCone** and make it a subclass of **RWTModel**.

Open **RWTCone.h** and replace it with the following:

```
#import "RWTModel.h"

@interface RWTCone : RWTModel

- (instancetype)initWithShader:(RWTBaseEffect *)shader;

@end
```

Nothing fancy here – it just derives from the `RWModel` class created in lecture and takes a shader class as a parameter.

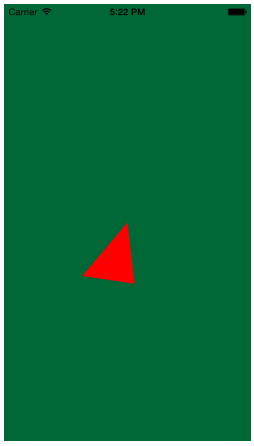
Next go to **RWTCone.m** and replace it with the following:

```
#import "RWTCone.h"

@implementation RWTCone

const static RWTVertex Vertices[] = {
    // Face 1: Red
    {{1, 0, 0.5}, {1, 0, 0, 1}},
    {{0, 1, 0}, {1, 0, 0, 1}},
    {{0, 0, 1}, {1, 0, 0, 1}},
};

const static GLubyte Indices[] = {
    0, 1, 2,
    3, 4, 5,
```

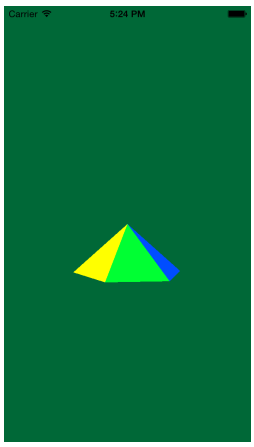
```
};

- (instancetype)initWithShader:(RWTBaseEffect *)shader {

  if ((self = [super initWithName:"cone" shader:shader
    vertices:(RWTVertex *)Vertices
    vertexCount:sizeof(Vertices) / sizeof(Vertices[0])
    inidices:(GLubyte *)Indices
    indexCount:sizeof(Indices) / sizeof(Indices[0])])) {

  }
  return self;
}

- (void)updateWithDelta:(GLfloat)aDelta {
    [super updateWithDelta:aDelta];
    self.rotationY += M_PI * aDelta;
}

@end
```

This is very similar to `RWCube` except it has different geometry. For now, I have only created one face of the cone for you – you will add the rest later.

Now for your first part of the challenge. Open **RWTViewController.m** and replace all references to `RWCube` with `RWCone` so that the cone shows up. Build and run, and you should see one face of the cone rotating like the following:



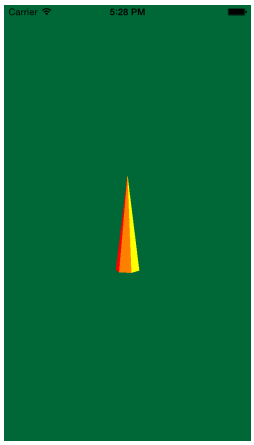
Now for your second part of the challenge, add the rest of the vertices and indices so that the rest of the cone renders. When you're done, you should see the following:



Part 2: Creating the Scene

Right now the cone doesn't modify it's position or scale; it only periodically updates its rotation. The scene itself has a translation though – it is moved -1 unit along the y axis, and -5 units along the z axis. This makes the cone seem further away and down a bit.

Your first challenge is set the rocket ships x scale to 0.25, z scale to 0.25, and y scale to 2.0. You'll have to modify `RWModel` to support this, as right now it only supports a uniform scale. When you're done, the cone should look more like a rocket ship:



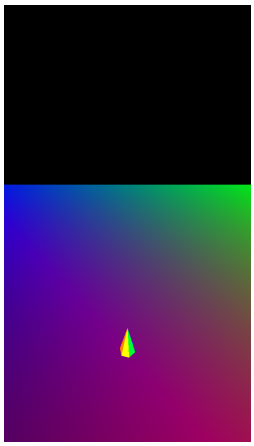
Next, you should set up your scene to the following:

- The rocket ship should be at 0, -5, 0.
- Add the square back to the scene. Rotate it and position it to be underneath the rocket ship. Also scale it 10x so it is very big, and acts as the "ground".
- By default you won't be able to see the full rocket ship anymore. To fix this, you want to "look" down at the rocket ship. You can do this by rotating the view

matrix 45 degrees along the x axis. But which comes first – the translate or the rotate?

- Also set the background color to black. This is outer space after all!

By the time you're done, you should see the following:



Part 3: We Have Liftoff!

Now it's time to make that rocket ship go!

To do this, add a new field to **RWModel.h**:

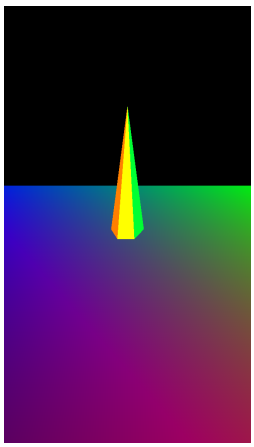
```
@property (nonatomic) GLKVector3 velocity;
```

And set the velocity in RWTViewController.m's setupScene method, right after setting the cone's position:

```
_cone.velocity = GLKVector3Make(0, 2, 0);
```

This sets the cone to move 2 units per second up along the y axis.

Your challenge is to update **RWModel.m** to update the position each frame based on the velocity. To do this, you will need to use some vector math. Hint: You'll need to use two GLKMath routines to do this.



Uber Haxx0r Challenge: Sky Watcher

Your final challenge is to add two more features to this scene:

1. Modify the scene to "look at" the rocket ship as it blasts off into the air. Basically, rotate the scene from -45 degrees to 45 degrees along the x axis at the rate of about 20 degrees per second. Alternatively, there's a methoded `GLKMatrix4LookAt` which you might find handy. ☺

2. Shake the entire scene several times for the first 3 seconds (by translating the scene up and down) to simulate the initial blast of the rocket ship.

Build and run, and watch the rocket ship fly away into the night sky!