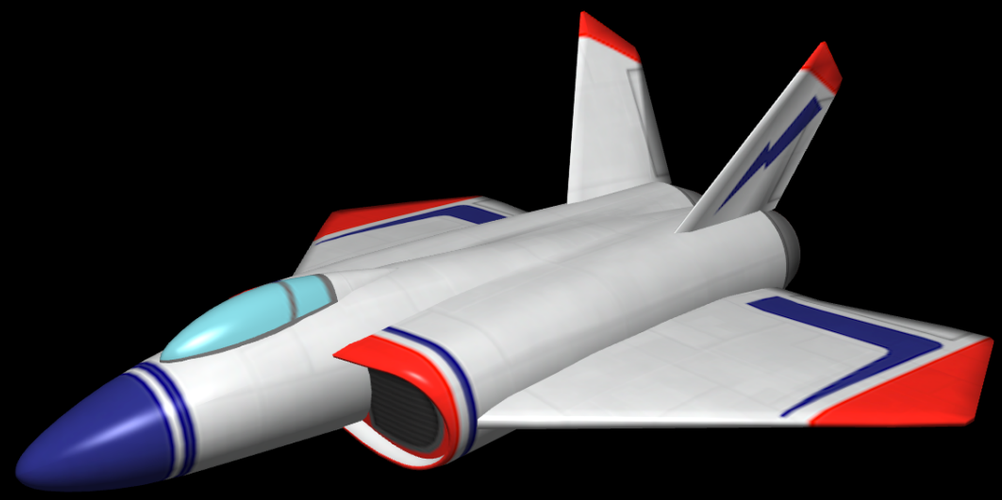
Three Aspects of SceneKit

John T Nelson

9/10/2016

Default SceneKit Project

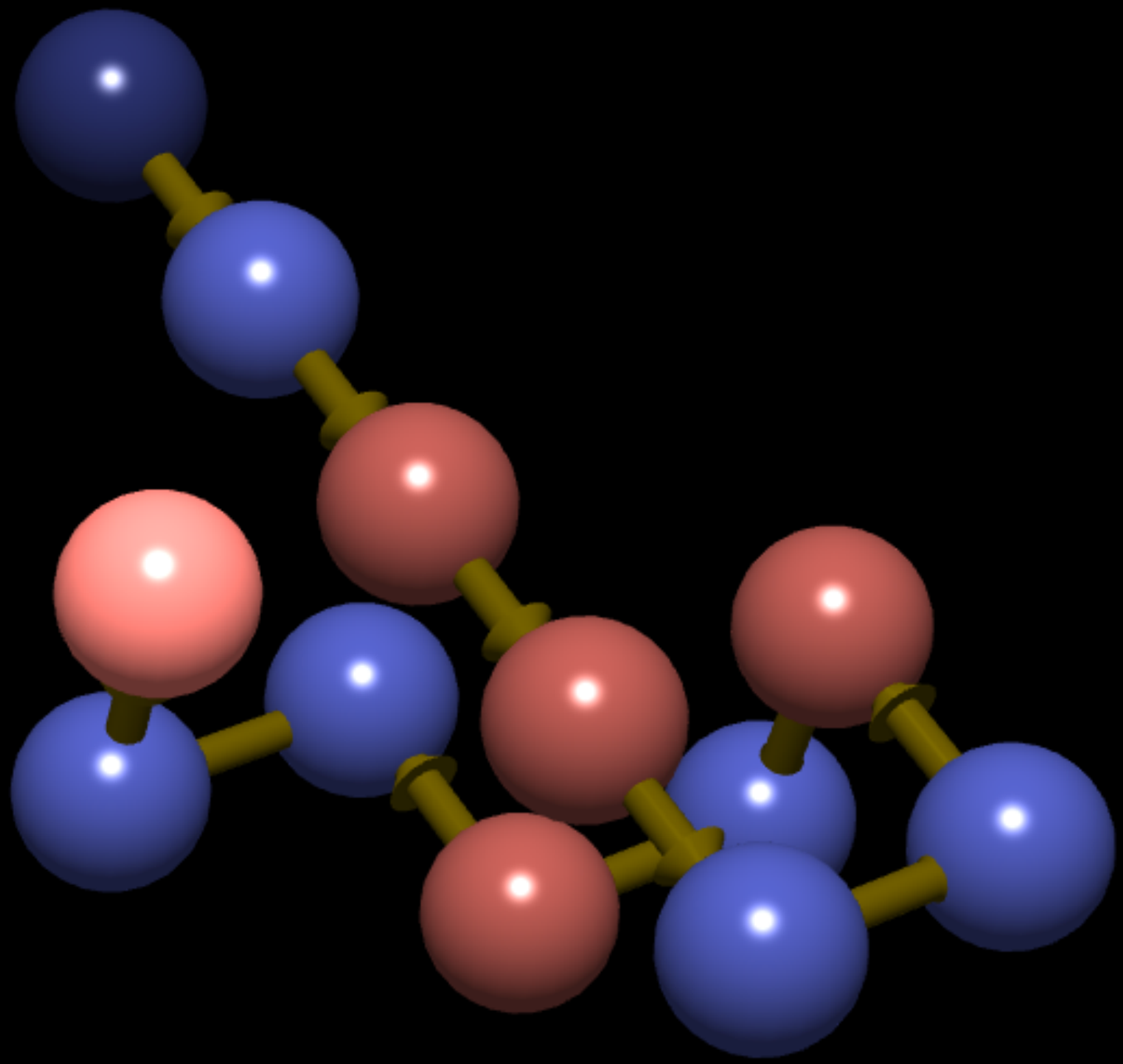


SceneKit Features

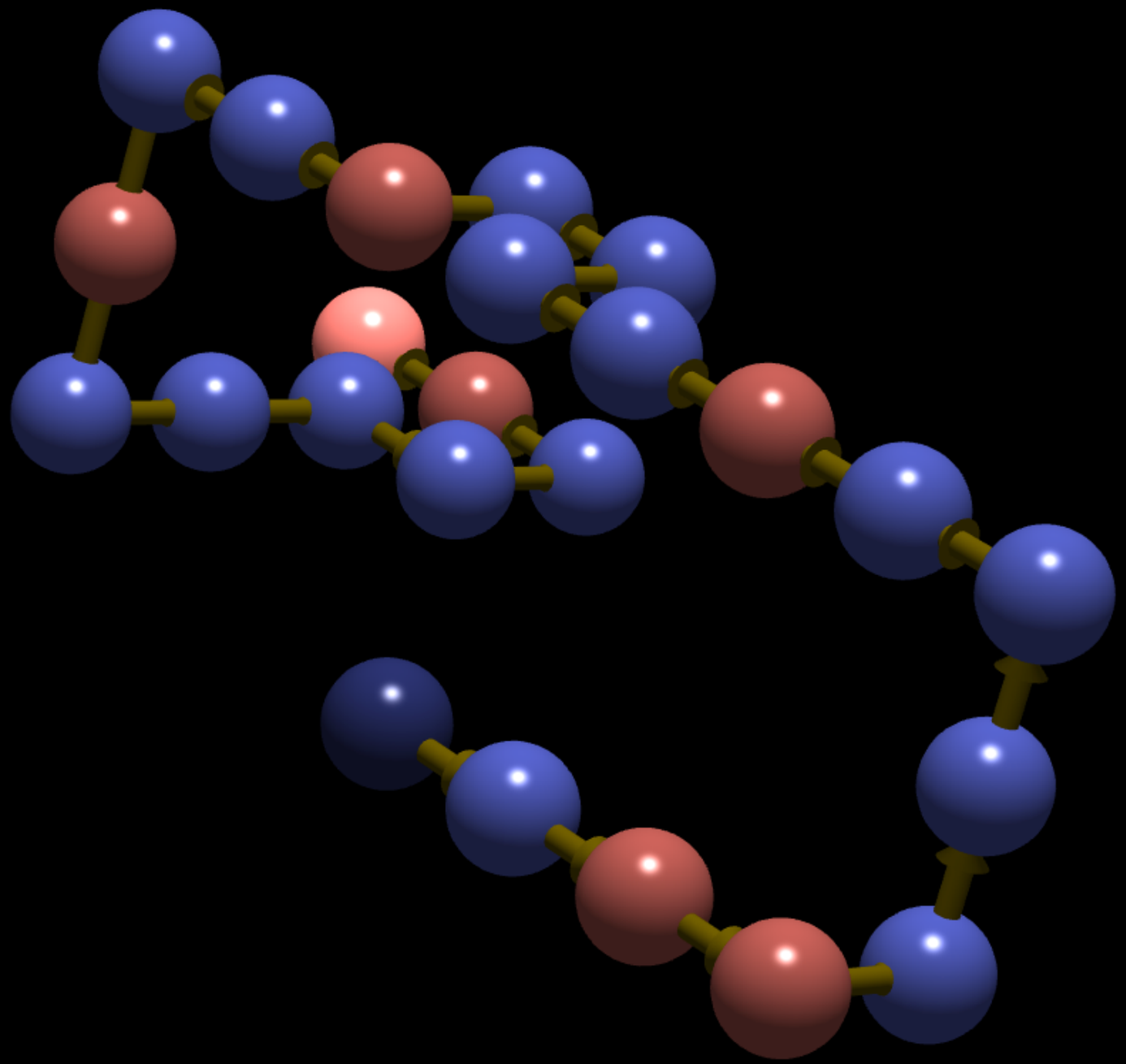
- Scene Graph Based
- Animation
- Lighting
- Geometric Primitives
- Texture Maps
- Physics
- Collision Detection
- Sound
- Atmospherics
- Emitters!
- Custom Geometry
- Model Loading

Visualization

PSP Visualization

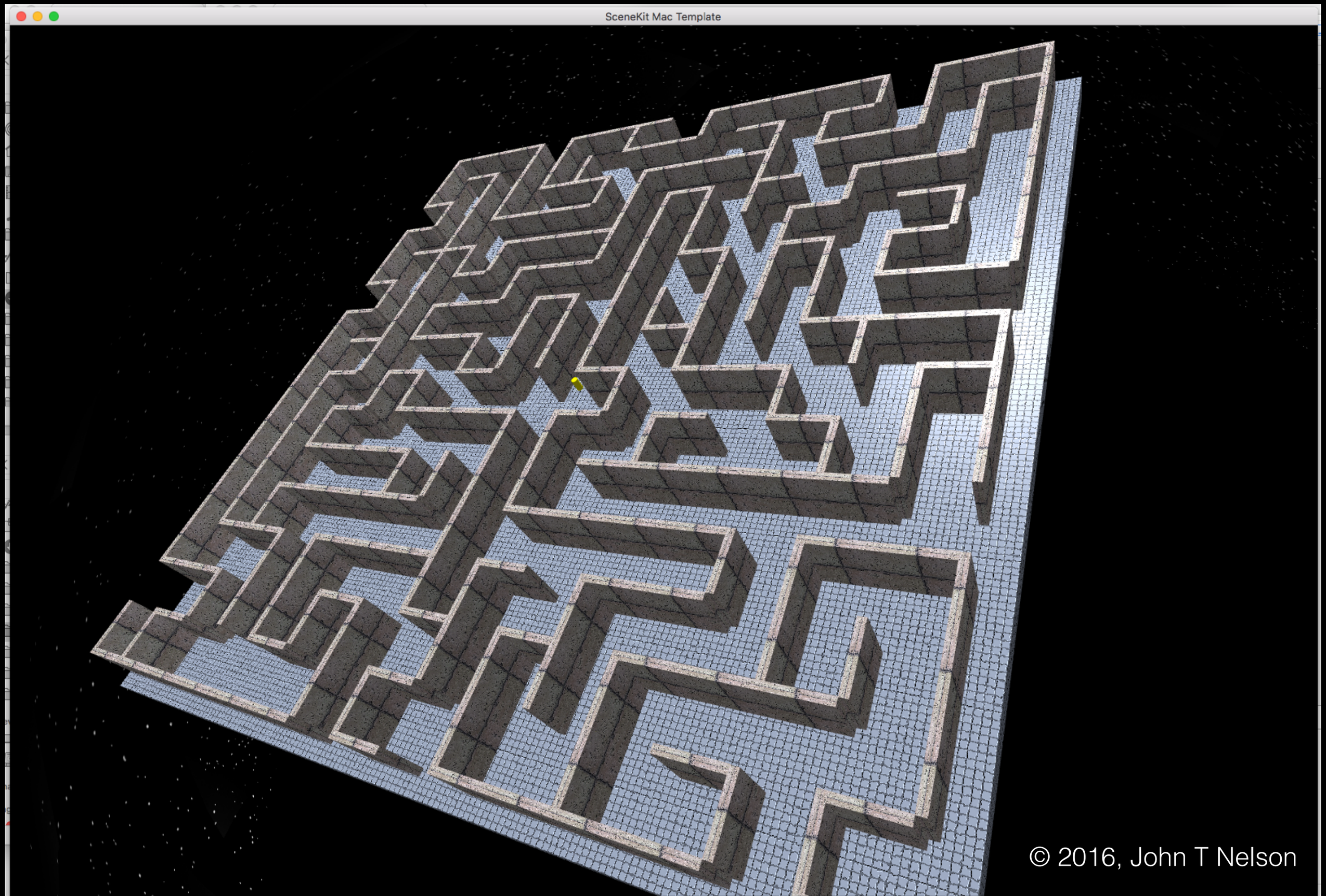


PSP Visualization

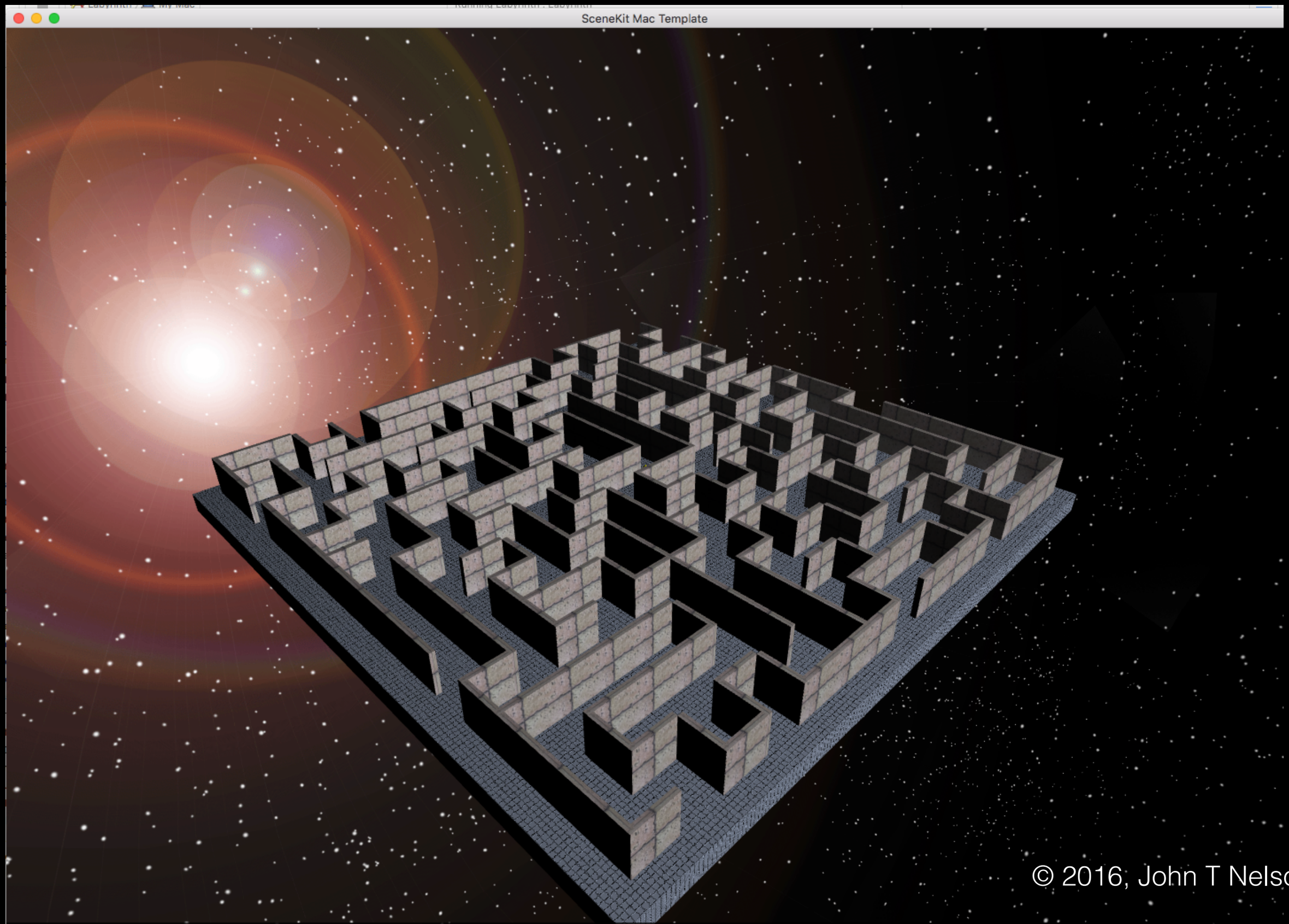


Game Design

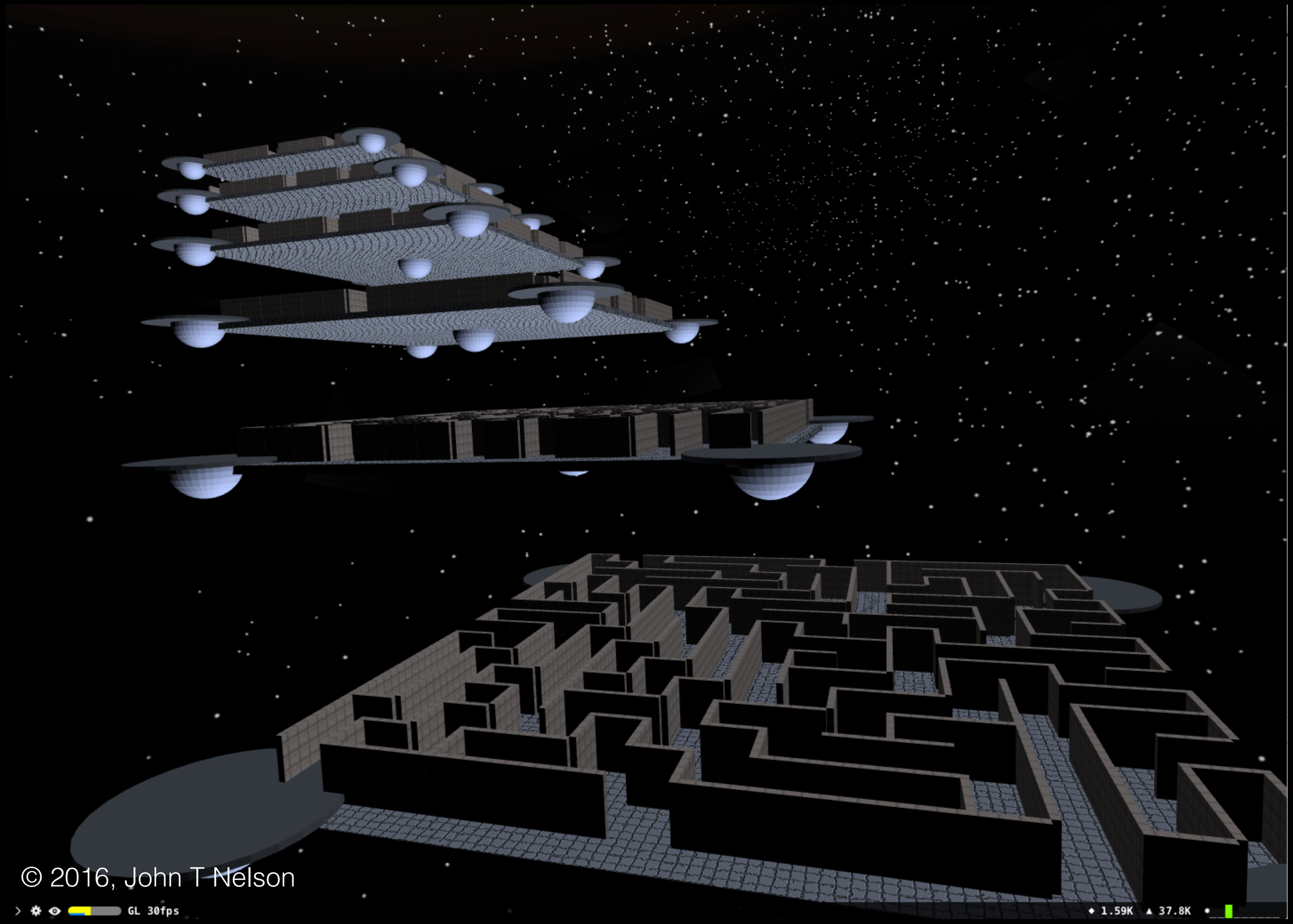
Algorithmic Maze Creation



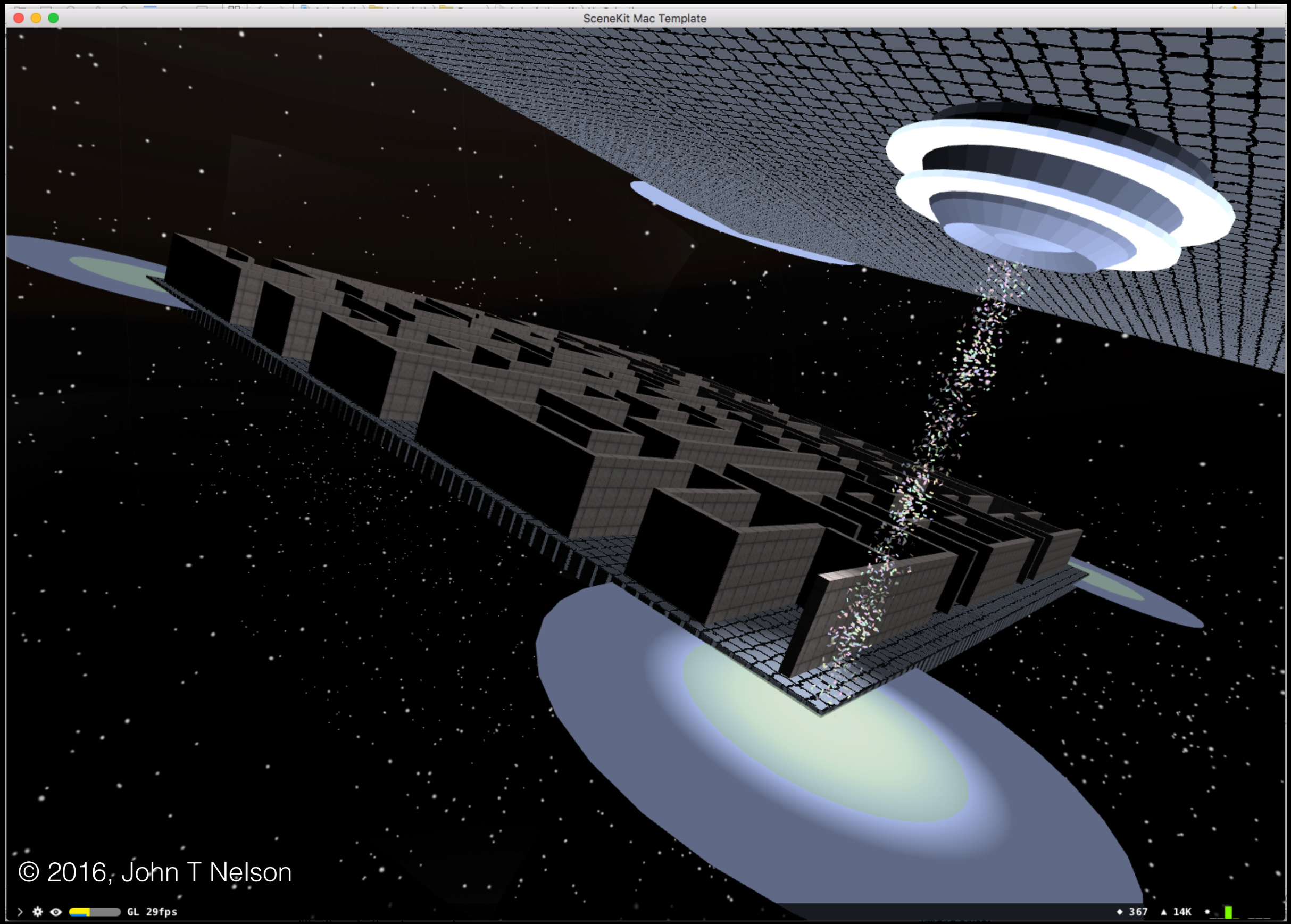
Lighting & Sky Box



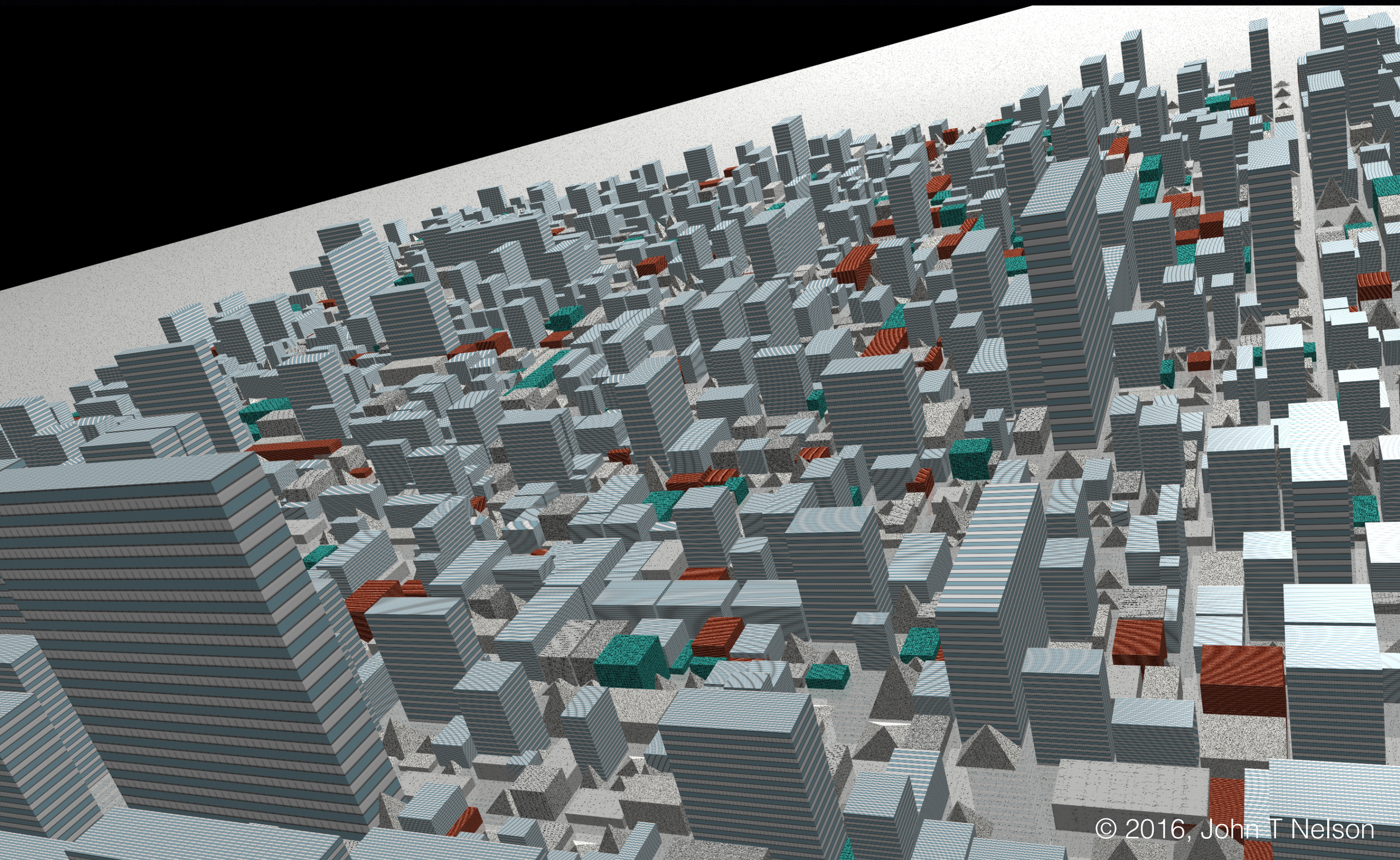
Repetition



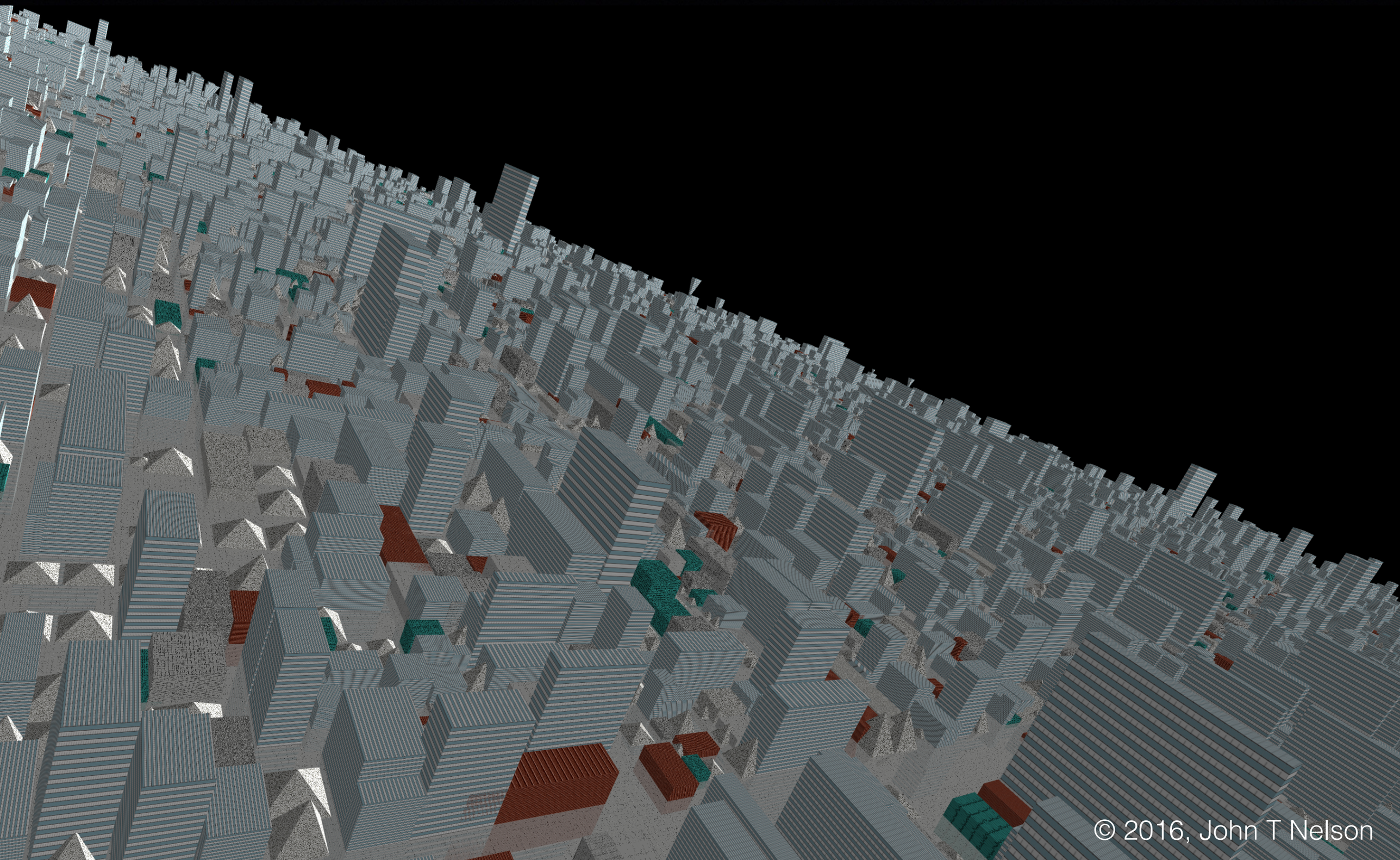
Emitters



Primitives



Scale & Performance

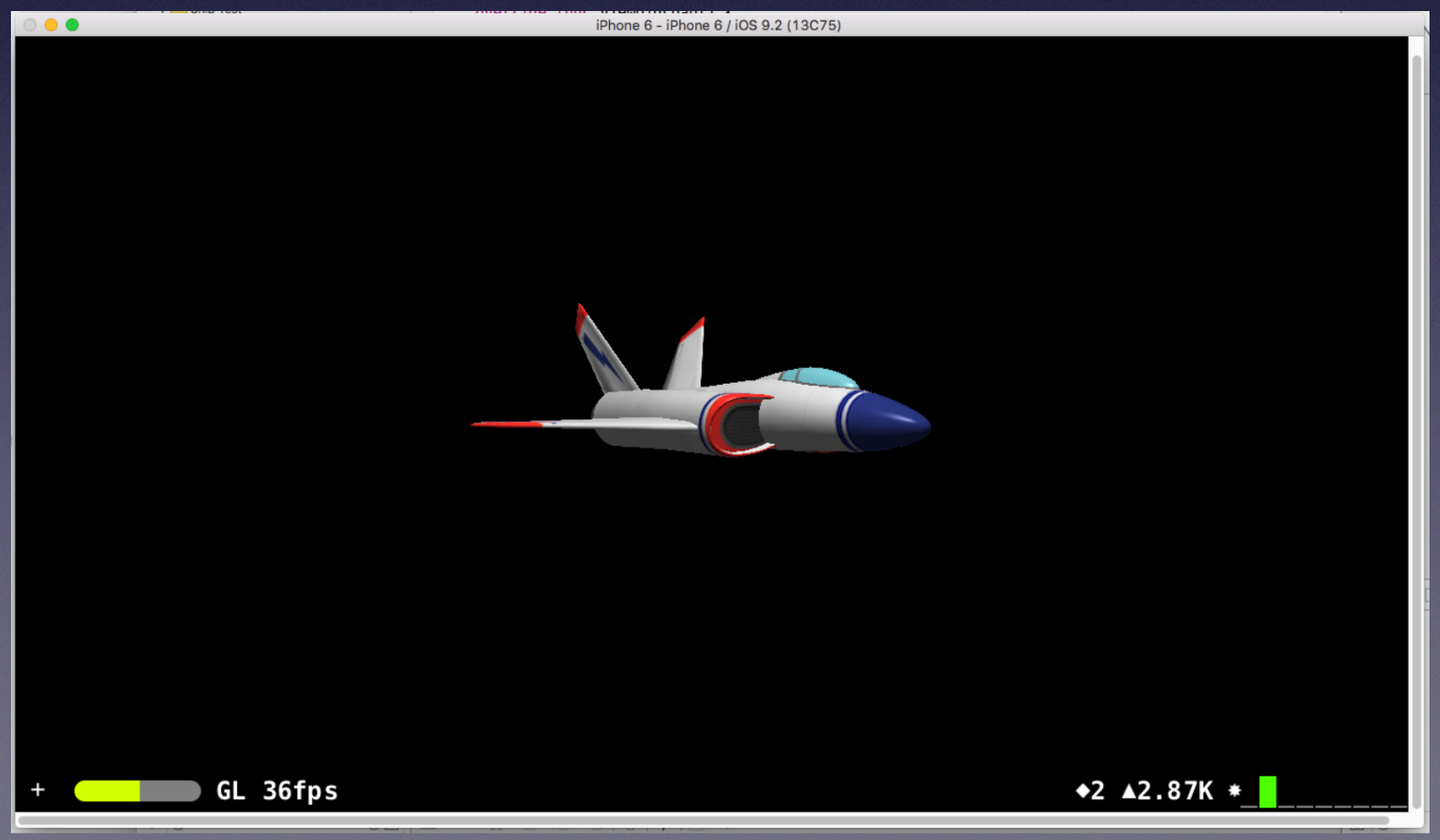


Virtual Reality

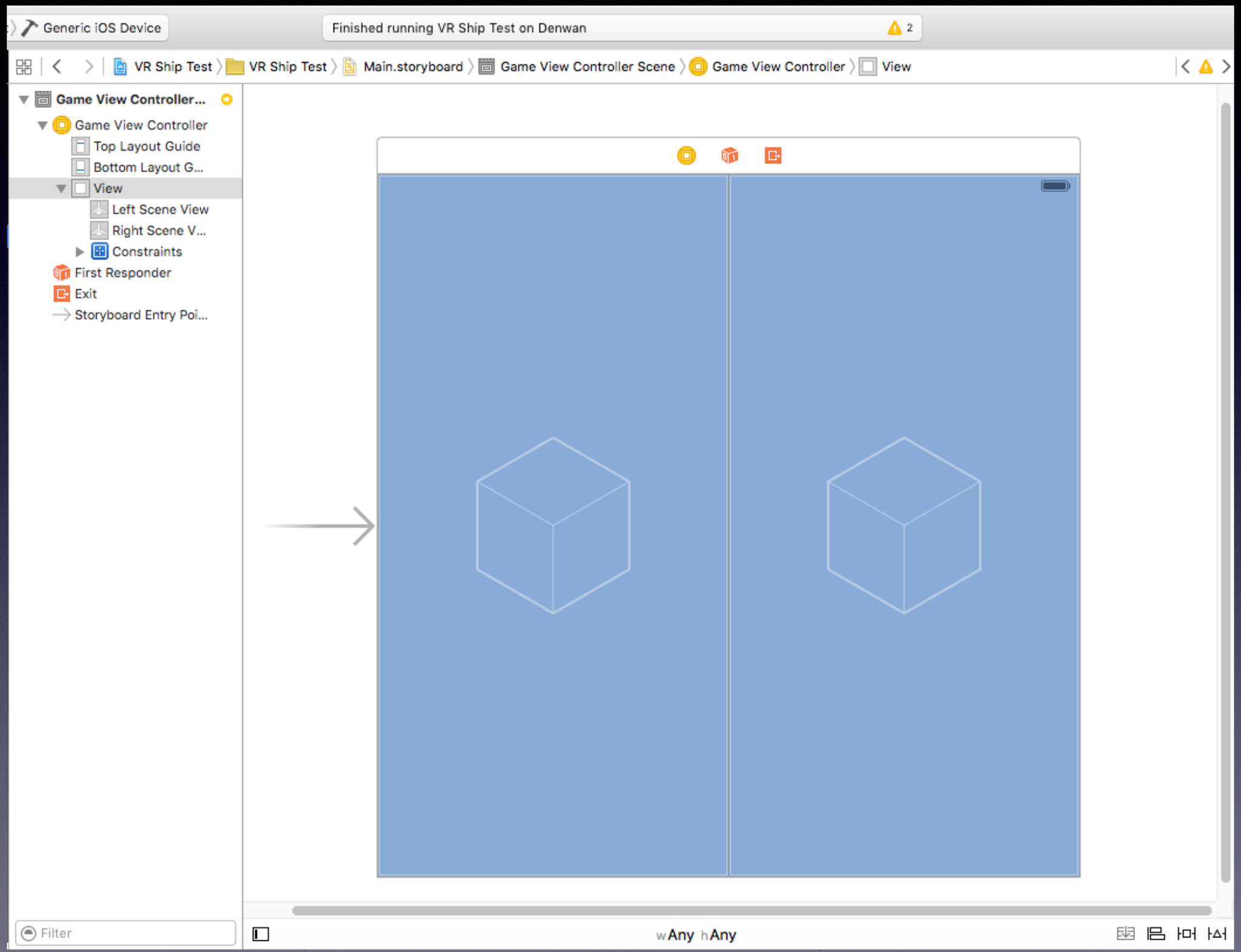
VR Camera Pattern

Single Camera

```
// create and add a camera to the scene
let cameraNode = SCNNode()
cameraNode.camera = SCNCamera()
scene.rootNode.addChildNode(cameraNode)

// place the camera
cameraNode.position = SCNVector3(x: 0, y: 0, z: 15)
```



VR Two-Camera Storyboard



VR Camera

```
import CoreMotion

class GameViewController: UIViewController, SCNSceneRendererDelegate {

    @IBOutlet var leftSceneView : SCNView?
    @IBOutlet var rightSceneView : SCNView?

    var motionManager : CMMotionManager?
    var cameraRollNode : SCNNode?
    var cameraPitchNode : SCNNode?
    var cameraYawNode : SCNNode?
    var scene: SCNScene!

    override func viewDidLoad() {
```

Two Camera Camera

```
// Create cameras
let leftCamera = SCNCamera()
let rightCamera = SCNCamera()

// JTN
leftCamera.zFar = 1000
rightCamera.zFar = 1000

let leftCameraNode = SCNNode()
leftCameraNode.camera = leftCamera
leftCameraNode.position = SCNVector3(x: -0.5, y: 0.0, z: 0.0)

let rightCameraNode = SCNNode()
rightCameraNode.camera = rightCamera
rightCameraNode.position = SCNVector3(x: 0.5, y: 0.0, z: 0.0)

let camerasNode = SCNNode()
camerasNode.position = SCNVector3(x: 0, y: 0, z: 15)

camerasNode.addChildNode(leftCameraNode)
camerasNode.addChildNode(rightCameraNode)
```

Two Scenes View

```
// create a new scene
let scene = SCNScene(named: "art.scnassets/ship.scn")!

// Space, the final frontier...

leftSceneView?.backgroundColor = UIColor.blackColor()
rightSceneView?.backgroundColor = UIColor.blackColor()

//scene = SCNScene()

leftSceneView?.scene = scene
rightSceneView?.scene = scene
```

Configure Camera

```
 // The user will be holding their device up (i.e. 90 degrees roll
 // from a flat orientation) so roll the cameras by -90 degrees to
 // orient the view correctly.

//camerasNode.eulerAngles = SCNVector3Make(degreesToRadians(-90.0), 0, 0)

cameraRollNode = SCNNode()
cameraRollNode!.addChildNode(camerasNode)

cameraPitchNode = SCNNode()
cameraPitchNode!.addChildNode(cameraRollNode!)

cameraYawNode = SCNNode()
cameraYawNode!.addChildNode(cameraPitchNode!)

scene.rootNode.addChildNode(cameraYawNode!)

leftSceneView?.pointOfView = leftCameraNode
rightSceneView?.pointOfView = rightCameraNode
```

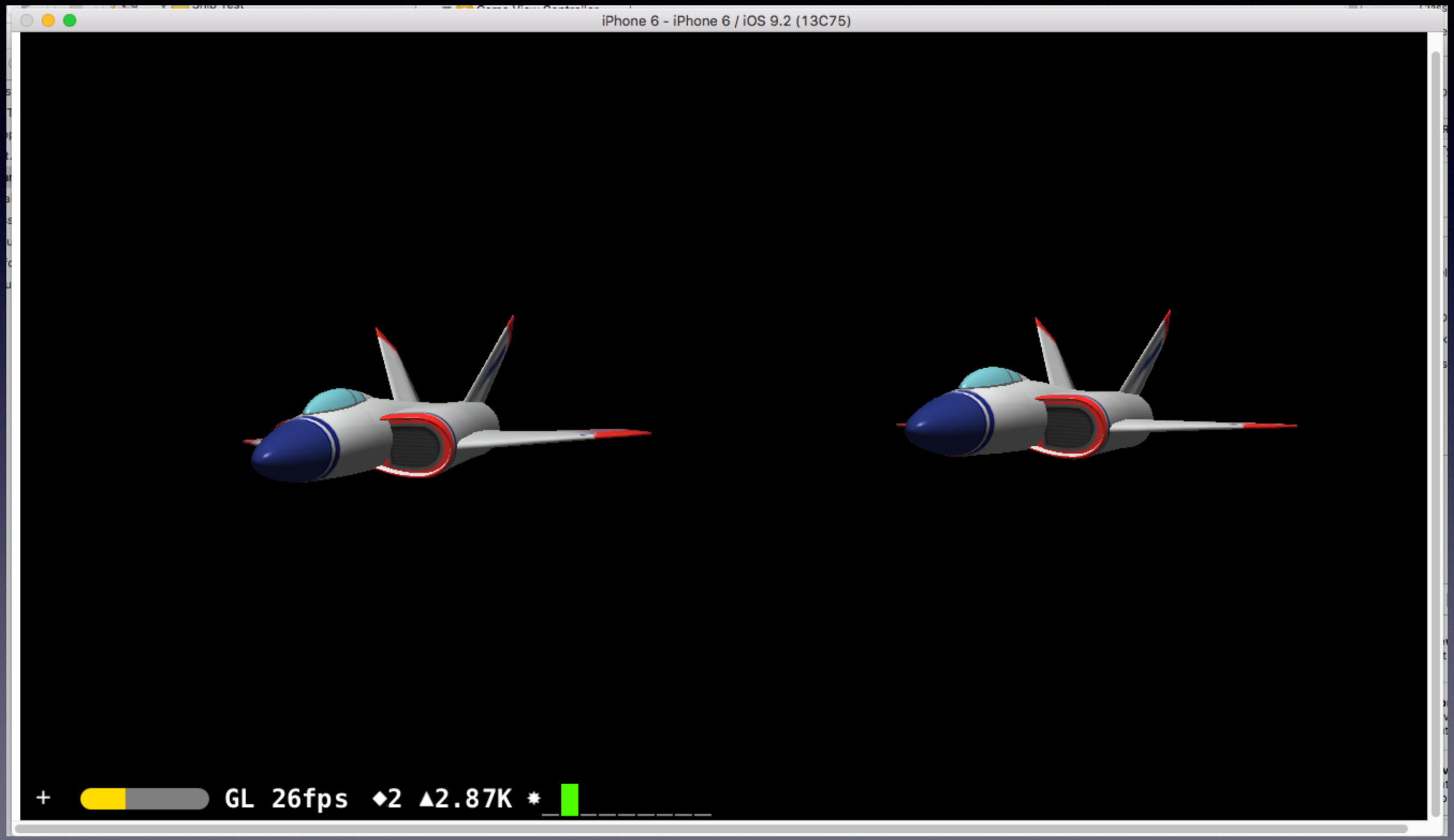
Configure Motion and Tap Capture

```
// Respond to user head movement
motionManager = CMMotionManager()
motionManager?.deviceMotionUpdateInterval = 1.0 / 60.0
        motionManager?.startDeviceMotionUpdatesUsingReferenceFrame(
            CMAttitudeReferenceFrame.XArbitraryZVertical)

// Testing...
leftSceneView!.allowsCameraControl = true

// Respond to taps...
leftSceneView?.delegate = self

// add a tap gesture recognizer
let tapGesture = UITapGestureRecognizer(target: self, action: "handleTap:")
leftSceneView!.addGestureRecognizer(tapGesture)

// Start...

//leftSceneView!.allowsCameraControl = true
leftSceneView!.showsStatistics = true

leftSceneView?.playing = true
rightSceneView?.playing = true
```

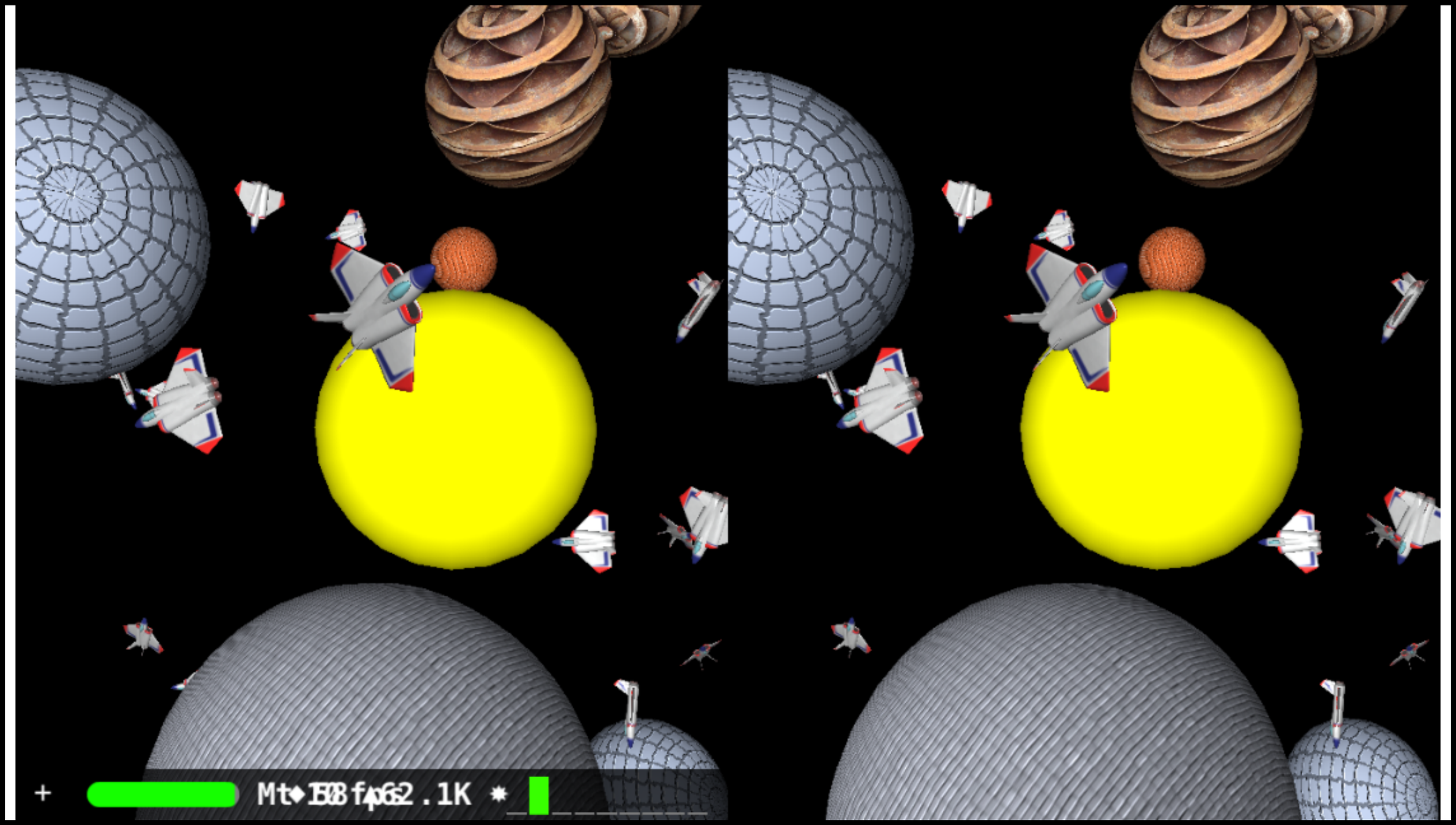
Handle Motion

```
func renderer(aRenderer: SCNSceneRenderer, updateAtTime time: NSTimeInterval)
{
    if let mm = motionManager, let motion = mm.deviceMotion {
        let currentAttitude = motion.attitude
        cameraRollNode!.eulerAngles.x = Float(currentAttitude.roll)
        cameraPitchNode!.eulerAngles.z = Float(currentAttitude.pitch)
        cameraYawNode!.eulerAngles.y = Float(currentAttitude.yaw)
    }
}
```

VR Camera



VR Planets



And Metal!

john@computation.com

@ComputationCom