|  |  |
| --- | --- |
| 计算机图形学project 3  WebGL编程基础 | 时间节点：  发布日期：2018-5-28  DeadLine：2018-6-24  负责TA  朱天乐  王思杰  Email：  [17212010053@fudan.edu.cn](mailto:17212010053@fudan.edu.cn)  [17212010036@fudan.edu.cn](mailto:17212010036@fudan.edu.cn) |
|  |  |

Project3 三维场景漫游

## 项目目的

理解WebGL绘制3D图形的原理，使用WebGL绘制简单的3D场景与动画。

## 项目知识点

着色器Shader的原理与代码编写，三维相机模型，三维空间坐标与物体，透视投影与矩阵变换，二维纹理，光照模型，从文件中载入和显示3D模型

## 使用的语言

Html5, Javascript, WebGL

## 开发与测试环境

**硬件：**带有支持OpenGL ES2.0显卡的计算机

**软件：**支持HTML5, JavaScript和WebGL的浏览器

本地Web服务器如Apache等，也可使用WAMP等集成环境安装包

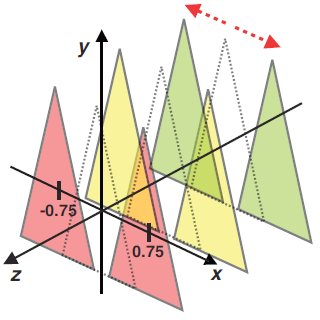
WebGL三维场景漫游

# 功能描述与知识点介绍

## 3D场景

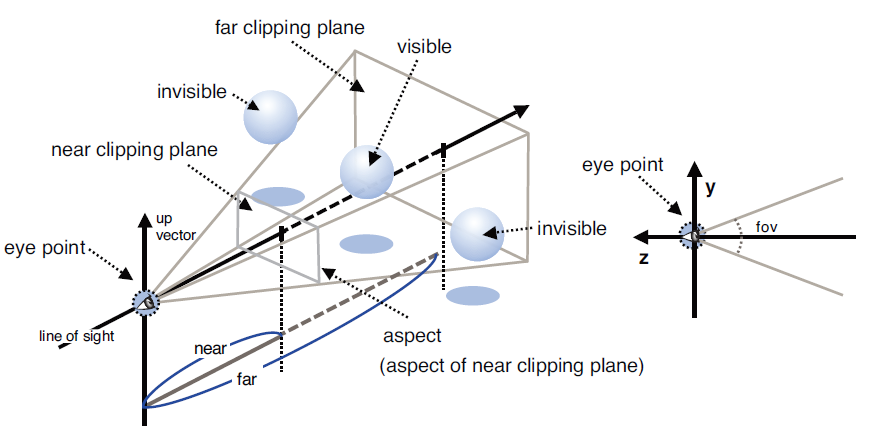
### 三维立体坐标系

本次Project所绘制的物体都是3维立体空间中的，因此，我们所使用的坐标系也是空间中的三维坐标系，有三个相互垂直的坐标轴，分别是X轴，Y轴和Z轴。相应的，模型上的每一个顶点都有三个坐标分量（x, y, z）。如下图所示：



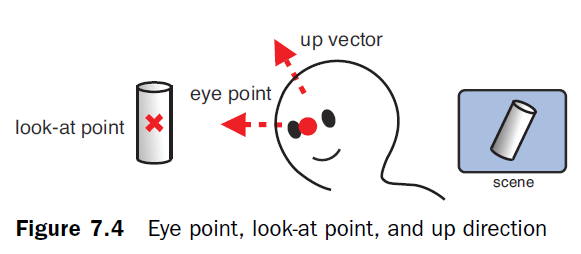
### 3D相机模型 - 透视投影

无论是用人眼看物体，还是摄影，写实油画等，其展现的画面中的物体都是近大远小的。在计算机三维立体绘制中，这样的成像方式我们称之为透视投影（与之相对的是正交投影，比较简单，详见教材图7.12），它需要一种特定的相机模型，如下图所示。我们需要四个参数来描述一个透视投影相机远/近成像距离，画面宽高比，垂直张角。



### 3D相机模型 – 相机位置与姿态

对于3D场景的展现，我们都是在场景中的某一位置，对着某个方向，使用透视投影的方式绘制一幅2D的画面。因此，我们还需要对摄像机的位置，朝向和姿态等信息进行描述。这里我们也需要一个参数化模型，如下图所示：



### 从2D到3D

在Project 2当中，大家已经能够在2D平面上绘制图形了，而且能制作简单的动画。在制作动画的时候，我们用到了矩阵变换来改变物体的坐标。在3D场景中绘制物体，其本质同样也是坐标变换，只不过其过程较2D变换而言复杂一些。我们需要如下几步：

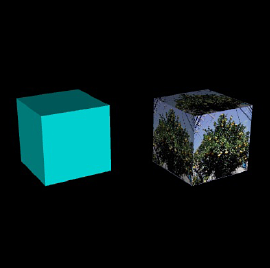
1. 将物体按照其位置，旋转，大小等描述，将其自身坐标，转换到世界坐标系中
2. 将三维世界中的物体坐标，转换到相机坐标系中
3. 将相机坐标系中的物体坐标，转换到相机成像平面中
4. 将相机成像平面上投影出的画面，映射到最终显示的画布当中

虽然上述步骤比较多，但是所有的这些都能够使用cuon-Matrix.js提供的API生成相应的矩阵。之后将物体顶点坐标与这些矩阵依次相乘，绘制，即可得到最终的画面。具体编程方法，可以参考教材第七章。

本次Project就是要求大家根据配置文件（配置文件scene.js在ProjectSample目录下，具体含义详见文件内的注释），绘制出一个3D的场景，如封面图所示，同时还要求大家能够通过键盘控制来在场景中漫游（前后左右移动，左右转身等）

## 纹理（贴图）

在室内装修的时候，很多人会觉得纯色的墙面过于单调，因此会选择在墙上贴一些墙纸来进行修饰。同样，对于3D场景中的物体，我们需要使其具有更逼真的细节，和更丰富的色彩。因此，我们需要使用纹理来赋予物体更多的表现力，如下图所示。



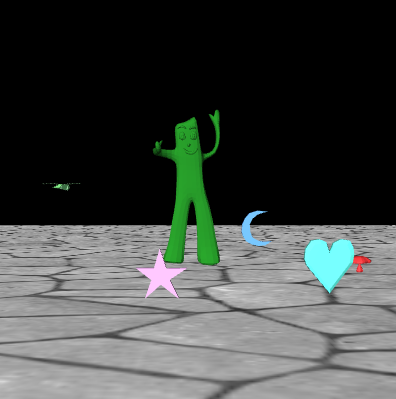
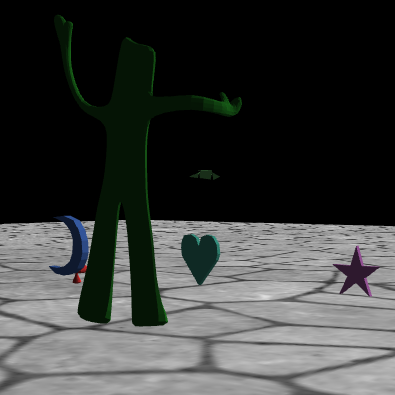
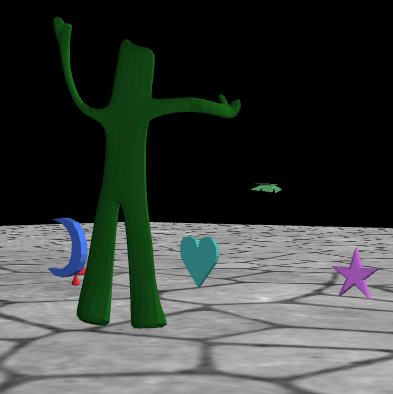
封面图中的地面和箱子也都使用了纹理，对于纹理的使用，请大家参考教材第5章Pasting an Image onto a Rectangle一节。

## 从obj文件中读取复杂的物体模型

在实际应用（比如游戏）的3D场景中，我们不能只绘制一些简单的物体诸如矩形平面，正方体，球体等，而是需要一些更加复杂的立体模型。这些立体模型通常由艺术设计人员使用诸如Maya，3DMax等三维建模软件手工制作的，并以一些特定的格式存储。对于这些比较复杂的物体，我们通常需要从文件中读取。本次Project当中就具有这样的物体，比如封面图中的月亮，心形，海宝等。从文件中载入三维模型也有简单的API供大家调用，参考教材第10章，Load and Display 3DModels一节。

## 光照模型

为了模拟真实世界中的效果，我们还需要在场景中添加光源，来使物体产生明暗等效果。如下图所示：

场景中添加了由海宝正面射向海宝背面的**平行光**线。所以，从正面看物体（左图）是明亮且色彩鲜艳的，而从背面看（中图），物体非常暗淡。另一方面，从背面看物体，并非完全漆黑，而是略微有一点色彩，这是**环境光**的影响。同时我们还需要添加一个跟随相机移动的**点光源**来随时照亮前方的物体，这个光源可以通过键盘上的F键来开启，开启后效果如图（右图）。关于光照的原理和添加方法，参考教材第8章。

## 动画

同样要求大家实现一个简单的动画。Sample代码中的动画为 bird模型 环绕 海宝模型 旋转飞行，高度按照旋转角的正弦值不断变化。对于具体的动画内容本次Project不做要求，同学们可以自由发挥，并在文档中描述清楚。

## 其它（也很重要）

### 优雅的Shader切换

本次Project要求同学们在绘制地面和箱子的时候，使用纹理来为物体着色，而在绘制其它物体的时候，使用光照模型。因此在一轮绘制当中，需要在两个Shader Program之间进行切换，先用一个shader绘制地面和箱子，再用另一个Shader根据光照绘制物体。之前为了简单起见，允许同学们多次调用initVertexShader函数来进行切换。但这种方法会影响系统性能，且为编程带来不便，本次Project要求大家使用教材第十章Switching Shader一节中的方法来进行切换。

### 场景漫游的实现方法

场景漫游，本质上是通过不断的移动摄像机的位置和朝向来实现的。具体要求是，使用键盘上的w/s来控制摄像机前后移动，使用a/d来控制摄像机左右移动（移动的速度为30个单位/秒），使用j/k来使摄像机以z轴为轴左右旋转（旋转速度为60角度/秒）。关于动画和速度控制，方法可以和Project 2相同。场景漫游的控制方法建议，通过键盘事件改变某个（些）全局变量g的值（表示状态的时候用布尔值或者0/1），同时通过包含了requestAnimationFrame的tick函数（Project2中有提到）来不断重新绘制场景，在绘制函数中读取全局变量g，并根据g的值（布尔）来决定是否改变摄像机的坐标。

# 实现步骤（建议）

1. 下载并安装支持html5和webgl的浏览器（如ie11，最新版本的Firefox，chrome和Safari等）
2. 使用安装好的浏览器，打开（运行）testBrowser.html确认浏览器支持html5以及是否已经开启了对webgl的支持，如果没有，请在浏览器设置选项中开启对webgl的支持。
3. 由于浏览器的安全策略，不允许js跨域请求，因此需要大家使用Web服务器来支持本次Project。具体为下载并安装WAMP集成环境，在其WWW目录中放置本次Project的html，css，js和其它各种需要用到的库和文件。在本地浏览器中通过localhost来访问Web页面。（也可以尝试修改浏览器安全策略，请根据浏览器种类和版本自行查找方法）
4. 详细了解项目目标与基本实现方法，项目目录下sample文件夹中有本次项目的结果范例（不可抄袭这里的代码）。
5. 学习WebGL基本编程方法。

以下内容为TA在学习WebGL和编写本次Sample code之后提出的建议过程，仅供同学们参考。

**第一部分 3D场景与贴图**

1. 阅读教材第七章内容，学习绘制三维立体图形的原理和基本方法，同时学会如何Drawing the Object with Indices and Vertices Coordinates。**建议参考代码为HelloCube和DepthBuffer。**
2. 阅读教材第五章，Pasting an Image onto a Rectangle一节，学习如何载入一张图片，并且将其映射到一个矩形上。**参考代码为TexturedQuad。纹理信息从配置文件中读取，纹理图片在ProjectSample中的image目录下。**
3. 结合6,7中的内容，根据配置文件设置摄像机的初始参数，并绘制出一个贴了图的正方体（木箱）。
4. 深入理解纹理绑定的每一行代码的含义，思考如何预先载入两张纹理，并在循环绘制中进行绑定。
5. 思考如何预先为两个物体生成各自的ElementArrayBuffer，并且在绘制循环中绑定各自的纹理。这里强烈建议大家参考教材第十章SwitchingShader一节的内容及相应的代码，里面的initArrayBufferForLaterUse，initElementArrayBufferForLaterUse和绘制过程中的initAttributeVariable三个函数应该能够给你足够多的启发。**参考代码为ProgramObject。**
6. 根据配置文件，绘制出箱子和地面。

**第二部分 3D场景漫游**

1. 根据要求实现漫游，也就是摄像机的平移和旋转，方法见1.5.2。**参考代码为LookAtTrianglesWithKeys\_ViewVolume。在ProjectSample目录下的MyVector3.js为大家提供了用于三维向量相关计算的工具函数，其操作的三维向量原型为cuon-Martix中提供的Vector3**

**第三部分 复杂模型的载入**

1. 阅读教材第十章中Load and Display 3D Models一节的内容，学习如何从obj文件中载入3D模型并且绘制。**参考代码为OBJViewer。在ProjectSample目录下的objReader.js为大家提供了用于读取3D模型（obj文件）的类，这些函数均提取自OBJViewer。模型文件在ProjectSample的model目录中**
2. 思考如何载入多个3D模型以及显示。
3. 根据配置文件，载入全部3D模型，进行相应的放缩旋转，并放置在3D场景中。

**第四部分 光照**

1. 阅读教材第八章的内容及相应的代码，学会如何向场景中添加不同种类的光源（环境光，方向光，点光源）。**建议参考代码为LightedCube\_ambient和PointLightedCube。**
2. 根据配置文件，向第三部分的代码中添加光源，实现光照效果。

**第五部分 动画**

1. 在第二三四部分的基础上，制作一个小动画，动画内容自由发挥。

**第六部分 多种Shader的综合使用**

1. 阅读教材第十章SwitchingShader一节的内容及相应的代码，学习及思考如何在同一场景的绘制中，使用不同的shader来绘制场景中不同的部分。将第一部分的代码，和第二三四五部分的代码融合起来，使其能够同时显示地面，箱子，复杂模型，漫游，光照，动画等所有效果。**参考代码为ProgramObject。**

物体

动画

光照

复杂模型的载入与显示

漫游

纹理贴图

3D场景的绘制

复杂模型场景漫游 – 地面可暂用纯色

地面和箱子

地面箱子场景漫游

拿着灯笼（点光源）漫游带光照的复杂场景

光源复杂场景

+会飞的小鸟

最终成品

1. 尽量在每完成一个部分之后，重构你的代码，比如调整代码结构，合并冗余代码，修改该变量名称使其更加合理等。这会极大的方便你之后的代码的编写。
2. 备份你的代码，然后自己做实验，尝试不同的光照效果，动画模型。
3. 学有余力的同学可以去尝试其他高级话题，比如纹理和光照的结合，phong shading，雾，影子等。这一部分会有最高20的额外加分（每项5分）。
4. 测试并调优你的代码。
5. 编写你的项目文档，内容需包括但不限于：你的项目目录及文件说明，开发及运行环境，运行及使用方法，你的项目中的亮点，开发过程中遇到的问题（以及你的解决办法），项目仍然或者可能存在的缺陷（以及你的思考），你对本课程project的意见及建议等。
6. 完成后将你的代码和文档放置在同一文件夹中，文件夹名称为”[学号]\_[姓名]”，使用zip格式压缩后上传至ftp。

# 项目要求及注意事项

项目要求如下：

1. 根据配置文件绘制出3D场景
   1. 绘制出地面和箱子，并对他们进行贴图
   2. 从obj文件中读取复杂的3D模型，并根据配置文件对3D模型进行旋转缩放，并移动到合适的位置
   3. 根据配置文件添加光照
2. 通过键盘使相机能够在3D场景中漫游
3. 为一些模型添加动画效果

注意事项：

1. 代码具有可读性：
   1. 所有代码有合理的缩进
   2. 类，函数，变量需合理命名
   3. 所有的函数，重要代码，算法中的关键步骤添加注释
2. WebGL默认后绘制的图形会遮挡住先绘制的图形，3D场景绘制会使用深度缓冲等实现近处物体遮挡远处物体
3. 在引入js库的时候，注意修改你的html文件，以及使用正确的相对路径
4. Javascript是弱类型的编程语言，所以在进行数据运算时务必注意。