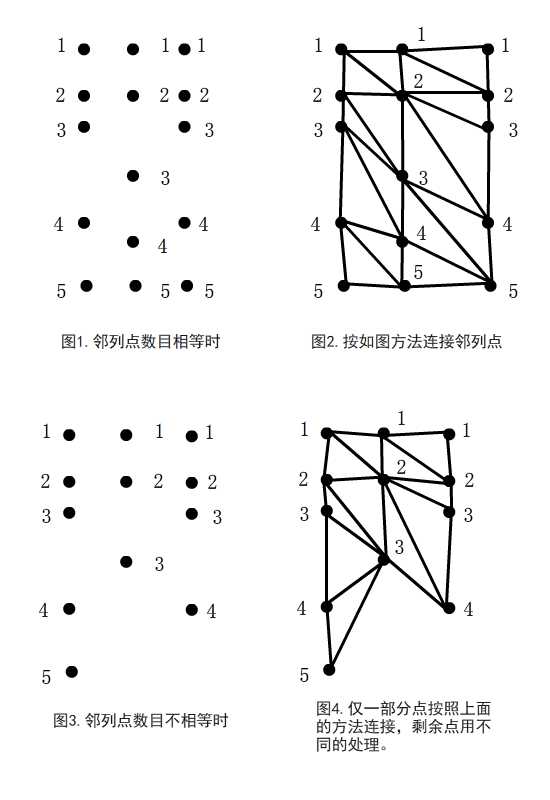
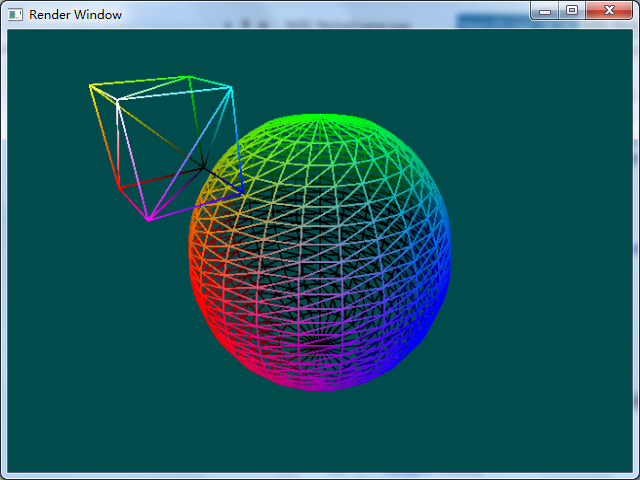
* **NoiseMesh：**

1. (2015.6.29)这是最基本的模型类了，所有Vertex和Index创建后都不需要改变了。
2. (2015.6.29)它自己也管理着一些属性，如Position,Rotation。当然顺带管理的就是World Matrix。它有自己的VB/IB，这样的结构比较清晰，当然以后要进行优化，因为有太多零散的小VB时，在执行Draw Call的时候代价较大，GPU需要等待CPU的传递命令时间。
3. (2015.6.29)Mesh要渲染的时候可以执行Add To Render List。
4. (2015.6.29)Mesh自带几种创建几何体方法。下面会提到具体。
5. (2015.7.7)现在会在类内管理一个Material结构体。
6. (2015.7.22)继承了NoiseFileManager，用于加载文件
7. (2015.8.20)准备实现多纹理(texture)和多材质(material)，这样的话要引入Mesh Subset(子集)的概念。我的定义是：具有同样纹理和材质的图元（三角形）集合，每个子集在shader里用同一个设置来Draw。因为Texture和Material都是用int来确定的，所以我们要先对三角形列表的TexID和MatID属性进行排序（应该要用到<algorithm>），然后再搞一个列表记录subset的区间端点（都是UINT）。
8. (2015.8.20)如果Material/Texture 的 ID不合法，会用NOISE\_MACRO\_INVALID\_XXX\_ID来表示，值为-1。
9. (2015.8.22)每个mesh必须有material。在mesh初始化时设一次user-set 的material（设为默认），用户也可以自己手动设置mesh的material ID（有一个专门的UINT来储存，就是防止用户在CreateXXX之前就设了material），在CreateXXX里面也会设置这种user-set 的 material。每次给mesh设定material都会更新mesh的图元信息列表和子集列表（subset）
10. (2015.8.22)没有无效的Material，如果设置了无效的mat ID就要把它的mat ID设为0（default）。（贴图可以不要，但是材质必须要有）

**————2015.6.29————**

**NoiseMesh::CreateSphere**

(2015.6.29)用的基本思路是当年做的3D扫描仪时圆柱体模型，用一张包着球体大网来生成球体网格。

(2015.6.29)如上左图解释，图1画出了3列点。当我们只考察第一与第二列点时，设函数L1(x)的返回值为第一列的第x个点，L2(x)的返回值为第二列的第x个点，T(p1,p2,p3)得到一个以p1,p2,p3为顶点的三角形，则假设每列点都有N个点，则设三角形集：

三角形的集合**E**为，

（呵呵论文上复制下来的，我的天这还是高二写的）

(2015.6.29)所以其实通俗的讲就是根据iRingCount的数量来确定横切片的数量，用iColumnCount来确定纵切片的数量，先生成侧面的网格。注意由于以后要加上纹理坐标，故第一列要复制多一份（因为接缝处纹理坐标不一样）。还要注意iColumnCount是不算重复的那一列的。最后再单独讨论TOP/BOTTOM的封顶问题（TOP顶点和BOTTOM是单独生成的）。反正就是这样了，代码那里可以慢慢纠结，这里给个思路。

(2015.7.16)生成顶点的时候，会有当前处理的列ID和环ID，把这两个ID分别映射到【0，1】，生成纹理坐标，就当是纹理的球形包裹方法。

(2015.8.23)生成y坐标时的步数(step)从y坐标换成了空间角，生成更均匀分布的顶点。

**————2015.6.29————**

**NoiseMesh::CreateCylinder**

(2015.6.29)其实跟CreateSphere没多大区别....Index都是同一种方法生成的的，只是多了个参数Height，Generate Vertex和其他稍作修改便可。

(2015.7.5)在顶点属性增加了Normal之后，一切都变得坑爹。然后就增加了TOP/BOTTOM两个重复的Ring，用于生成不同的法线。

(2015.7.16)这个柱形纹理坐标的生成会有点特别吧，texCoordX还是照旧和球体类似，但是texCoordY是把 ( (半径x2) + 高)的这段长度映射到【0，1】，以确保两个顶面和侧面都不会有太大的纹理拉伸。