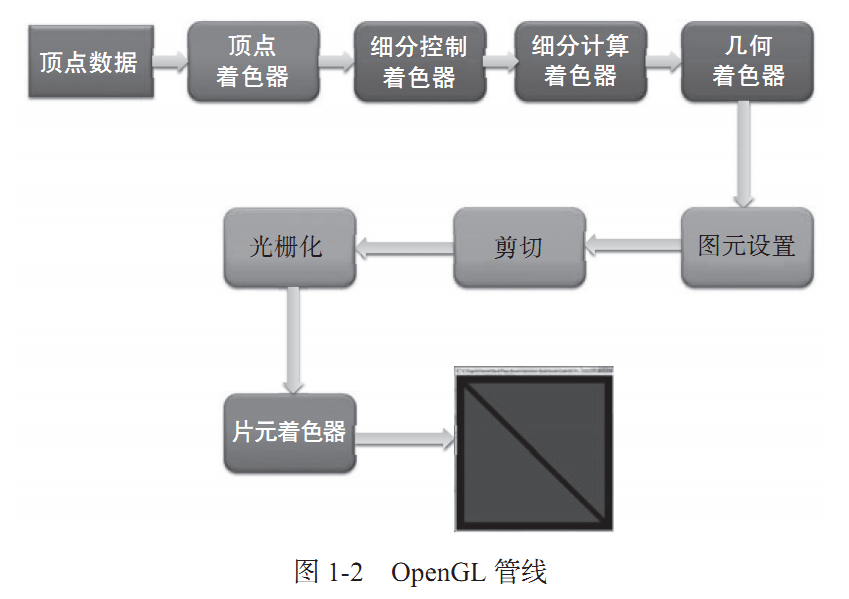
关于OpenGL中的几个坐标系统的理解

Author:lihaiping1603@aliyun.com

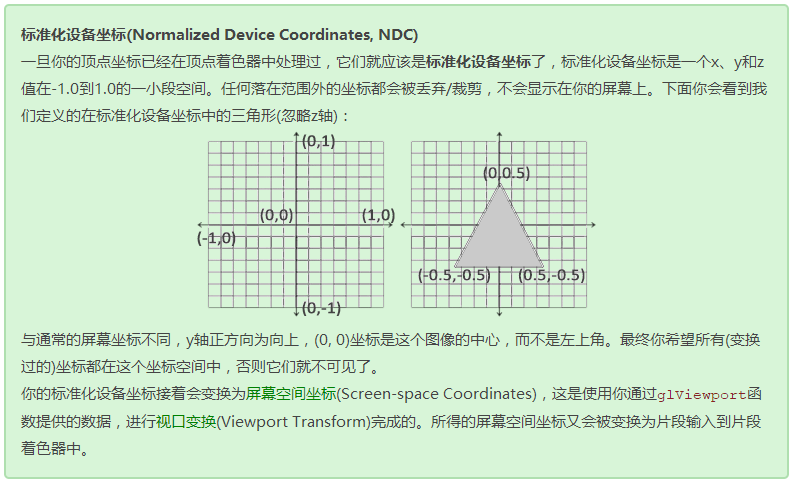
在我们使用opengl做图像处理的过程中，其中必不可少的基本都会用到顶点着色器和片元着色器。

完整的渲染管线图：



那么在这两个着色器程序中，我们需要绘制我们的图像的时候，他们的坐标和位置对应关系是如何的？

这里我们可能会想到坐标系统中介绍的坐标变换，他们涉及了局部空间，世界空间，观察空间，剪切空间，再到屏幕空间。其实这几个坐标系统关联的是对顶点数据的操作，我们可以在CPU中将这几个矩阵的计算应用于顶点数据，然后得到一个结果传递给顶点着色器使用，也可以直接将这几个变换矩阵直接传递进顶点着色器，然后再GPU中进行数据的计算，这都是可以的，然而这整个的计算过程的目的是啥？为的是将处理前的顶点数据处理为我们标准化设备坐标，方便我们映射到屏幕坐标，那么opengl中的顶点数据对应的标准化设备坐标系统是如何的呢？还记得LearnOpenGL中的这个图么？



所以处理过以后的顶点数据，他对应的就是这个坐标系统，值的范围为[-1,1].

所以我们在很多的时候，看到二维图形处理的时候，如果我们在后期不做变换的话，直接定义顶点数据位置，然后传递给顶点着色器中的顶点数据，就是对应的上面这个归一化坐标系统。

例如下面我们将整个视频图像绘制在整个视窗上的话，二维的顶点数据定义如下：

static const float vertex\_position[12] = {

-1.0f, -1.0f,

1.0f, -1.0f,

-1.0f, 1.0f,

-1.0f, 1.0f,

1.0f, -1.0f,

1.0f, 1.0f

};

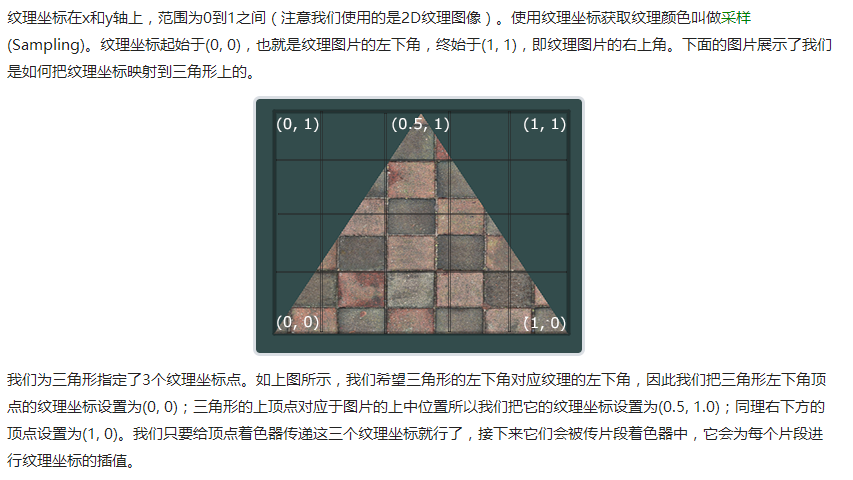
这个定义是我们很常见的。

然后接着再说一下片元着色器中的纹理坐标。我们在片元着色器中，经常性的拿纹理坐标进行贴图，或者进行一些计算，或者啥的其他操作等。

而片元着色器的纹理坐标很多时候会经过这样一个操作，这个是我们常见的，先将纹理坐标数据从CPU传递到顶点着色器中的变量，然后顶点着色器将纹理坐标数据再传递给片元着色器。这样我们贴图的时候，就能将纹理贴到我们所得到的纹理坐标位置了。

那么纹理坐标是否也跟顶点数据一样，有一个归一化的坐标系统呢？

是的，暂且我们就这么喊吧，叫他为纹理归一化坐标系统。同样我们引用learnOpenGL中纹理一节中的图：



纹理坐标值的归一化范围为[0,1]，所以如果我们自己定义上面的这个三角形纹理坐标：

float texCoords[] = {

0.0f, 0.0f, // 左下角

1.0f, 0.0f, // 右下角

0.5f, 1.0f // 上中

};