# OpenGl ES Study

  基本概念

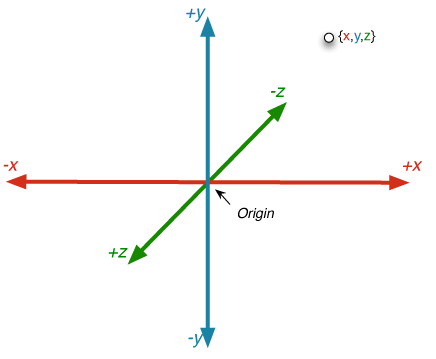
          OpenGL定义了一个跨编程语言的，跨平台的编程接口的规格，是一个性能卓越的三维图形标准。OpenGL是一个专业的图形程序接口，是一个功能强大，调用方便的底层图形库。OpenGL ES是专为嵌入式和移动设备设计的一个2D/3D轻量级图形库。是OpenGL的一个子集。

          3D图像的最小单位成为点（point）或者顶点（vertex）。它们代表三维空间中的一个点并用来建造更复杂的物体。多边形是由一个个点构成，而物体是由多个多边形组成。尽管通常OpenGL支持多边形，但是OpenGL ES 只支持三边形，所以即使我们要绘制一个正方形也要把它拆分成两个三角形绘制。

          先说下坐标系的问题。

         默认情况下，以屏幕中心为坐标轴原点。原点左方X为负值，右边为正值，原点上方Y为正值，原点下方为负值，垂直屏幕向外为Z为正值，向里为负值。

         坐标如图所示：



例如一个正方形的4个顶点可表示为：

float vertices[] = {

-1.0f, 1.0f, 0.0f, //左上

-1.0f, -1.0f, 0.0f, //左下

1.0f, -1.0f, 0.0f, //右下

1.0f, 1.0f, 0.0f, //右上

};

为了提高性能，通常还需要将浮点数组存入一个字节缓冲中。所以有了下面的操作：

ByteBuffer vbb = ByteBuffer.allocateDirect(vertices.length \* 4); //申请内存

vbb.order(ByteOrder.nativeOrder()); //设置字节顺序，其中ByteOrder.nativeOrder()是获取本机字节顺序

FloatBuffer vertexBuffer = vbb.asFloatBuffer(); //转换为float型

vertexBuffer.put(vertices); //添加数据

vertexBuffer.position(0); //设置缓冲区起始位置

其中ByteOrder.nativeOrder()是获取本机字节顺序。OpenGL ES有操作图形渲染管线的函数，在默认情况下这些函数功能的使用状态是处于关闭的。 启用和关闭这些函数可以用glEnableClientState、glDisableClientState来完成。

// 指定需要启用定点数组

gl.glEnableClientState(GL10.GL\_VERTEX\_ARRAY);

// 说明启用数组的类型和字节缓冲，类型为GL\_FLOAT

gl.glVertexPointer(3, GL10.GL\_FLOAT, 0, vertexBuffer);

// 不再需要时，关闭顶点数组

gl.glDisableClientState(GL10.GL\_VERTEX\_ARRAY);

**边**

       边是连接两个点的一条线，是多边形面的边缘。  
 **多边形**

        多边形是由边构成的单闭合环。 OpenGL ES中的多边形必须是凸多边形，即在多边形的内部任意取两点， 如果连接这两个点的线段都在多变的内部，这个多边形就是凸多边形。 绘制多边形时需要指定渲染的方向, 分为顺时针和逆时针。 因为方向决定了多边形的朝向， 即正面和背面。 避免渲染那些被遮挡的部分可以了有效提高程序性能。 函数glFrontFace定义了渲染顶点的方向。

GL\_CCW 表示窗口坐标上投影多边形的顶点顺序为逆时针方向的表面为正面。GL\_CW 表示顶点顺序为顺时针方向的表面为正面。

/ 设置CCW方向为“正面”，CCW即CounterClockWise，逆时针

glFrontFace(GL\_CCW);

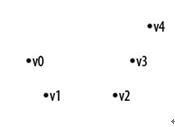
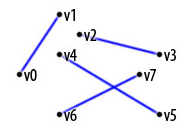
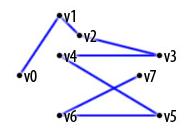
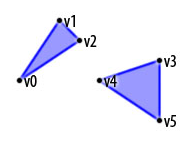
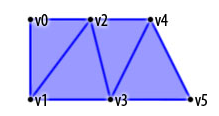
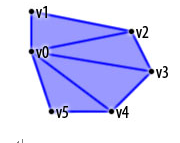
// 设置CW方向为“正面”，CW即ClockWise，顺时针

glFrontFace(GL\_CW);

**渲染**

       有了以上的概念讲解后，现在要进行最主要的工作—渲染。渲染是把物体坐标所指定的图元转化成帧缓冲区中的图像。图像和顶点坐标有着密切的关系。这个关系通过绘制模式给出。常用到得绘制模式有GL\_POINTS、GL\_LINE\_STRIP、GL\_LINE\_LOOP、GL\_LINES、GL\_TRIANGLES、GL\_TRIANGLE\_STRIP、GL\_TRIANGLE\_FAN。下面分别介绍：

GL\_*POINTS*：把每一个顶点作为一个点进行处理，顶点n即定义了点n，共绘制n个点。

  
GL\_LINES：把每一个顶点作为一个独立的线段，顶点2n-1和2n之间共定义了n个线段，总共绘制N/2条线段。，如果N为奇数，则忽略最后一个顶点。  
           
GL\_LINE\_STRIP：绘制从第一个顶点到最后一个顶点依次相连的一组线段，第n和n+1个顶点定义了线段n，总共绘制N-1条线段。           
            
 GL\_TRIANGLES：把每三个顶点作为一个独立的三角形。顶点3n-2，3n-1和3n定义了第n个三角形，总共绘制N/3个三角形。  
            
 GL\_TRIANGLE\_STRIP：绘制一组相连的三角形。对于奇数点n，顶点n，n+1和n+2定义了第n个三角形；对于偶数n，顶点n+1，n和n+2定义了第n个三角形，总共绘制N-2个三角形。       
            
GL\_TRIANGLE\_FAN：绘制一组相连的三角形。三角形是由第一个顶点及其后给定的顶点所确定。顶点1，n+1和n+2定义了第n个三角形，总共绘制N-2个三角形。  
            

绘制函数：

void glDrawArrays(int mode, int first, int count)

void glDrawElements(int mode, int count, int type, Buffer indices)

glDrawArrays创建一个几何图元序列，使用每个被的数组中从first开始，到first + count – 1结束的数组元素， mode为绘制模式。

glDrawElements使用count个元素定义一个图元序列，type是indices(指数)数组中的数据类型，mode为绘制模式，indices数组存储顶点的索引值。（按照指定的indices数组顶点索引，来绘制指定的顶点）

 绘制图形步骤：

1.定义顶点并且转换存储在字节缓冲中;

2.我们使用顶点数组绘制图形，而opengles是默认关闭这个开关的，所以我们要启用它。gl.glEnableClientState(GL10.GL\_VERTEX\_ARRAY);

3.设置绘制的颜色。以下为设置红色

gl.glEnableClientState(GL10.GL\_VERTEX\_ARRAY);

4.由于我们使用顶点数组，我们必须通知OpenGL顶点的数组在什么地方。需使用函数：

gl.glVertexPointer(

3,//每个顶点的坐标的维数，这里为3xyz

GL10.GL\_FIXED,//顶点坐标值的类型为GL\_FIXED

0,//数组中数据的偏移值

mVertexBuffer//顶点坐标数据数组

);

 5.开始绘图gl.glDrawArrays(GL10.GL\_TRIANGLES, 0, 3);

参考文献

http://blog.csdn.net/wangkuifeng0118/article/details/7423109#