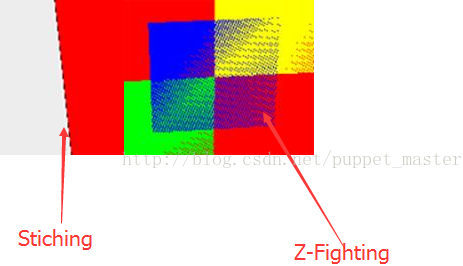
Offset指令

写到这里强行插一波基础知识。上面的描边效果，我们用了一个Offset指令，很好地解决了穿插的问题。其实Offset就是解决Stiching和Z-Fighting的最佳途径之一。当然，也可以用模板测试，但是Offset操作更快一点。关于Stiching和Z-Fighting，引用一下这篇文章：

在OpenGL中，如果想绘制一个多边形同时绘制其边界，可是先使用多边形模式GL\_FILL绘制物体，然后使用多边形模式GL\_LINE和不同的颜色再次绘制这个多边形。但是由于直线和多边形的光栅化方式不同，导致位于同一位置的多边形和直线的深度值并不相同，进而导致直线有时在多边形的里面，有时在多边形的外面，这种现象就是"Stiching"。而Z-fighting主要是指当两个面共面时，二者的深度值一样，深度缓冲就不能清楚的将它们两者分离开来，位于后面的图元上的一些像素就会被渲染到前面的图元上，最终导致图象在帧与帧之间产生微弱的闪光。



比如我们要绘制两个面完全共面时，两者深度值完全相同，那么我们在进行深度测试的时候，就不能分辨到底哪个在前，哪个在后了。类似我们上面的例子，当我们需要渲染背面时，通过背面进行外拓的Pass渲染的结果就和正常的Pass有穿插了。那么，要解决这个问题，很明显，我们就可以强行设置某个pass的深度偏移，推测这个offset的偏移值是针对ZTest阶段，在进行深度测试的时候，将当前pass的深度用offset进行调整再与深度缓冲区中的值进行比对。附上一张官方文档中关于Offset的部分：

Offset指令有两个参数，一个是Factor，主要影响我们绘制多边形的深度斜率slope的最大值；另一个是Units，主要影响的是能产生在窗口坐标系的深度值中可变分辨率差异的最小值r，这个r一般是OpenGL平台给定的常量。最终的Offset = slope \* Factor + r \* Units。Units我们一般在有使用Offset指令的地方给一个统一的值就可以了，主要起作用的就是Factor。Offset操作的层面是像素级别的，多边形光栅化之后对应的每个Fragment都有一个偏移值，我们调整Factor，其实相当于沿着当前多边形的斜率深度前进或者后退了一段距离，默认的深度方向是向Z正方向，如果我们给一个大于0的Factor，那么偏移值就会指向Z正方向，深度测试的时候相当于更远了一点；而如果给了个小于0的Factor，相当于原理Z正方向，深度测试时就更近了一点。

总结一句话就是：Offset大于0，Pass对应的模型离摄像机更远；Offset小于0，Pass对应的模型离摄像机更近。