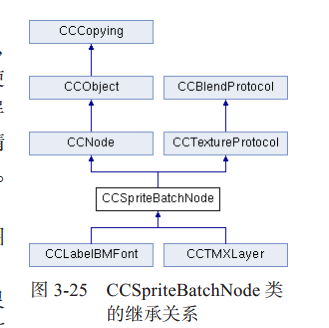
概述

2014-2-14

当你需要显示两个或两个以上相同的精灵时，如果逐个渲染精灵，每一次渲染都会调用OpenGL的函数，因为当系统在屏幕上渲染一张贴图的时候，图形处理硬件必须首先准备渲染，然后渲染图形，最后完成渲染以后的清理工作。以上是每次渲染固定的开销，这样帧率就会下降15%左右或者更多。

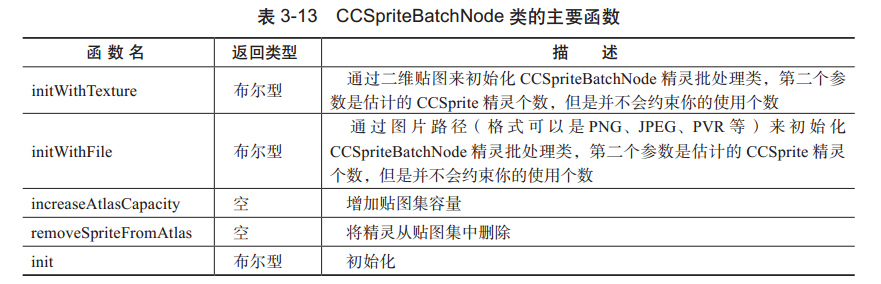
如果将所有需要渲染的同一张贴图只进行一次准备，一次渲染，一次清理就可以解决这个问题了。这时可以使用CCSpriteBatchNode类来批处理这些精灵，比如游戏屏幕中的子弹等就可以这样做。用它作为父层来创建子精灵，并且使用它来管理精灵类，这样可以提高程序的效率。CCSpriteBatchNode类的继承关系如图3-25所示。可以看到，CCSpriteBatchNode类继承于节点类和贴图协议。



这里需要说明的是，加入CCSpriteBatchNode类的精灵类越多，提高效率的效果就越明显。不过也有一些限制，所有属于同一个CCSpriteBatchNode类的精灵类都有相同的深度值，也就是说，如果需要呈现一个子弹在人物前面、另外一个子弹在人物后面的不同遮挡关系，获得每个子精灵并单独设置和重排序它们，尽管使用的是同一张贴图，但可以把它们理解为不在同一“层”（并不是布景层）。

此外，所有属于同一个CCSpriteBatchNode类控制的精灵类必须使用同一张贴图，但是这并不是一个限制，如果想使用不同的图片，可以把它们放在同一张贴图集当中。

另外还有一些限制，就是CCSpriteBatchNode类设置锯齿/抗锯齿效果时，所有子精灵也同时设置了锯齿/抗锯齿效果，不可以单独设置。同样不能单独设置的还有混合函数（blendfunc）。可以把CCSpriteBatchNode类理解为CCLayer类，只不过CCSpriteBatchNode类只接受CCSprite类和它的子类。下面介绍CCSpriteBatchNode精灵批处理类。



创建方法的第一个参数可以是贴图对象，也可以是图片路径。这里主要说明两个创建方法的第二个参数。这个参数是子节点的数量。当然，如果使用第一种方法不显示的定义子节点的数量，系统会使用默认值29，在运行时如果超过空间了，会增加33%的容量。