[**Flash与3D编程探秘（一）- Flash与3D空间**](http://www.cnblogs.com/yangzhou1030/archive/2008/10/30/1322592.html)

日期：2008年10月

#### ****Flash 和 3D空间****

第一件事情我想你知道的是，在Flash里，并不存在真正的3D，或者我应该说，Flash CS3并不支持3D绘制。我们所做的是运用Flash里的2D绘制方法去模拟3D绘制，Flash并不知道3D是什么也不知道如何去处理3D对象。但是好 消息是所有的3D处理和3D计算都是建立在数学计算的基础上的，加上Flash知道如何的处理数学计算。太好了，这些工具足以使我们创造出自己的动画了。 这并不是说Flash里的3D编程要简单，与其相反，如果你打算深入去探索的话你会发现你会进行很大一部分的Low level Programming。不管怎样，即使你没有3D编程经验，只要有读文章和多多动手联系，我相信你也能快速的学会这些看似深奥的东西。另外，虽然文中数 学占很大一部分比例，但是文章中我会用最大的努力把数学部分变得简单易懂，如果实在有问题的话，你可以找一本数学资料参考。

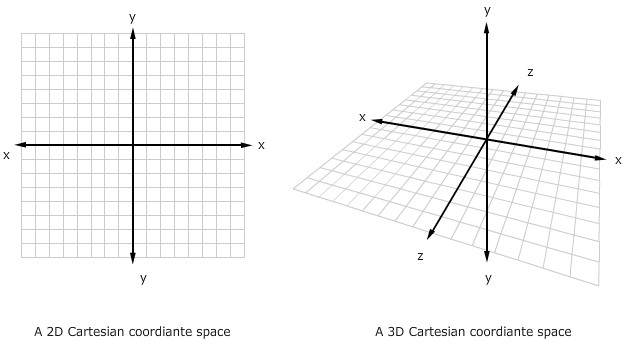
在Flash里，有两种3D处理方式，一种是提前处理好3D模型，另外一种是在程序运行时通过数学计算处理3D图形。运用第一种方法，可以提前处理好一系 列的图形，然后通过对祯或者是时间操作进行动画播放，以达到3D效果，这一种方法对美工的要求比较高。第二种方法是通过大量的数学计算对物体进行操作，实 时计算出物体的顶点的位置，方向，旋转角度等等变量，并且绘制出图形，这也是在这篇文章里所关心的。

#### ****3D绘制工具****

如今你在互联网上搜索3D Flash你会找到很多的工具和程序，比如Swift，利用这些程序你可以绘制一些3D图形并导出你需要的格式，甚至完全使用在你的项目里。基本上你可以 不必操心如何绘制3D图形。但是，这些程序给你提供的都是提前做好的模版，不能够作太多的runtime处理，有一定的局限性。于是，我们发现自己也有大 脑，也可以制作3D的运动动画，还可以制作复杂的3D绘制引擎，最重要的是能够探索其中的奥秘，学习为什么在OpenGL里使用glVertex3f函数 就可以定义一个空间点。这并不像我们在使用OpenGL时那么得心应手，你要做的是去实现glVertex3f这个函数而不是去利用它。一切3D的物体都 要自己动脑动手加上数学运算进行绘制，肯定会增加一些数学上的挑战。不过我认为，这对你来说应该不是问题，其实使用Flash作3D图形常常会给你带来你 意想不到的乐趣，那么开始吧！

#### ****3D空间坐标系****

从技术角度而言，Flash中并不存在3D，也就是说z轴并不存在，所以所谓的z轴是由你来制作的，利用缩放物体让大家产生3D的错觉。那么也就是说，对于一个3D虚拟空间，z轴与你看进显示器的方向重合，x轴和y轴分别为横向和纵向。



**2D与3D坐标系**

#### ****对于z轴的解释****

在现实中，当一个物体离你远去，那么对你的眼睛来说，它所看到的是物体越来越小。当然并不只有物体的大小在改变，物体离你的距离也在增大。可以假定，在3D空间里，离得人眼越远的物体，它的大小就越小，那么它在x和y轴上的移动就越缓慢。很简单对吧？很好。

#### ****注意：****

如果你用过OpenGL，那么你应该知道y轴的正方向是指向上方，z轴正方向指向屏幕外，然而文章中的Flash 3D空间的y轴和z轴是相反的。

#### ****原点****

Flash中3D空间是围绕坐标系原点的，原点的坐标我们用Point(x, y, z), (0, 0, 0) 来表示。在Flash 2D中，原点存在于程序的左上角Point(0, 0)，那么对于3D来说，原点也自然存在于程序的左上角，也许你会发现，如果原点在左上角的话，那么你所在的位置肯定是成一定角度（仰看）来观察程序中 3D空间中的物体的。当然可以把3D空间的原点向右再向下移动，因为围绕原点绘制物体的时候会发现方便很多。

叙述了这么多，到底用什么方法来表达3D空间呢？

#### ****缩放物体****

离人眼越近，那么物体就越大，反之物体就越小。物体缩放的比率以及移动速率与物体z的大小成反比。现在我用一个实例给说明如何制造3D动画效果，在这个例 子中，我手工绘制了几个小球，作为虚拟的3D的物体，让它们沿着z轴在舞台上来回移动，以制造3D效果。虽然很基本，不过别担心，把这些简单的东西掌握好 是深入探索的基础。注意：从第一到第六篇文章中都不涉及3D物体的概念，例子中使用的小球（小P）是2D矢量图，这样做的目的是让你在不关心物体的情况下 锻炼空间感。

**小球与3D空间（无层次）**

#### ****动画制作步骤****

1. 第一步，用Flash画出一个你喜欢的物体，任何物体都可以。在这个例子中我画了一个蓝色的小球。当然可以导入你喜欢的图片，不过不要忘记在 Library里创建的物体上点击右键，选择Linkage，然后在Export For Actionscript上打勾。

2. 下一步，详细解说一下代码。当然一开始要设置一些变量，原点和焦距（摄像机）。Focal length（焦距）确定了摄像机（在本例子中为人眼的）的凸透镜的焦距，值越大，那么物体的扭曲就会越小。把它设为400，这是一个在本例子中适中的数值。

var origin = new Object();  
origin.x = stage.stageWidth/2;  
origin.y = stage.stageHeight/2-80;

[复制代码](javascript:void(0);)

var focal\_length = 400;

3. 创建一个舞台，并且把它的x和y设置为原点，这样在在舞台上创建物体时，就会默认原点在程序窗口的中央了。

var scene = new Sprite();  
this.addChild(scene);  
scene.x = origin.x;  
scene.y = origin.y;

[复制代码](javascript:void(0);)

4. 然后要在舞台上添加一些我们绘制好的小球。在这个例子中我们绘制3个，分别在左中右。把它们的x\_3d, y\_3d, z\_3d，也就是它们的3D空间的x，y，z的值设为相应的数值，我把它们排为一排。每一个都添加一个direction属性，1代表向屏幕方向移 动，-1 代表向我们的方向移动。然后设置它们的移动速度，并且添加到舞台上。这时你如果编译的话，你会看到有3个球在舞台上，那么下一步就让小球运动起来。

for (var i = 0; i < 3; i++)  
{  
    var sphere = new Sphere();  
    sphere.x\_3d = -190+i\*160;  
    sphere.y\_3d = 80;  
    sphere.z\_3d = i\*100;  
    sphere.direction = 1;  
    sphere.speed = 6;  
    scene.addChild(sphere);  
}

[复制代码](javascript:void(0);)

5. 下面这个函数，在每一次执行，都会把小球移动到相应的位置，并且对小球进行缩放。当小球的z大于600时，让它向相反的方向移动。当小球的z\_3d值变化 后，计算小球当前的大小和位置，把小球移动到相应的位置然后对其进行缩放，这样在一连串的函数执行后，就会得到动画效果。代码里scale代表物体应该缩 放的比率，因为当物体沿z轴移动的时候，物体的大小以及x和y值都会改变。所以要计算出这个比率，那么我们才能把物体缩放到合适的大小，并且把物体移动到 相应的2D空间位置。

function run(e:Event)  
{  
    for (var i = 0; i < scene.numChildren; i++)  
    {  
        scene.getChildAt(i).z\_3d += scene.getChildAt(i).speed\*scene.getChildAt(i).direction;  
        if (scene.getChildAt(i).z\_3d > 600)  
        {  
            scene.getChildAt(i).z\_3d = 600;  
            scene.getChildAt(i).direction = -1;  
        }  
        else if (scene.getChildAt(i).z\_3d < 0)  
        {  
            scene.getChildAt(i).z\_3d = 0;  
            scene.getChildAt(i).direction = 1;  
        }  
          
        var scale = focal\_length/(focal\_length+scene.getChildAt(i).z\_3d);  
        scene.getChildAt(i).x = scene.getChildAt(i).x\_3d\*scale;  
        scene.getChildAt(i).y = scene.getChildAt(i).y\_3d\*scale;  
  
        scene.getChildAt(i).scaleX = scene.getChildAt(i).scaleY = scale;  
    }  
}

[复制代码](javascript:void(0);)

6. 最后，在舞台上添加一个函数循环响应时间。让第5步写的函数循环执行。编译看一下，现在小球在3D舞台上移动了。

Hooray！你的第一个Flash3D程序完成了。总结一下，其实并没有使用任何高深的技巧，只不过是利用了变化物体的x和y以及小球的缩放来制造3D效果。对你来说太简单？好，那我们继续。

#### ****层叠****

在Flash中表现3D空间，仅有缩放是不够的，还需要另外一个技巧，层叠。它的基本概念是，离人眼较近的物体会在离人眼较远的物体之上显示。

在上一个例子里面，我们缩放小球，以达到3D效果。可是你会发现，3个小球之间的x距离都很大，那么你也许会想如果3个小球离得很近的话，会出现什么现象呢？

#### ****注意****

尝试把上面例子中小球之间的x距离变小，看看有什么变化？这时会发现，不管小球离我们多远，右边的小球始终在最上面。即使中间的小球应该盖过右边小 球的时候，右边的小球还会在上面。这是因为在把小球添加到舞台上的时候，已经给了小球层次，也就是说最后添加的小球（右边的小球）就在最上面。

**小球3D空间（无层次）位置不对了！**

看起来应该设计一种方法实现小球的层次感，当小球离我们远的时候，那么它的层次就比较靠后，以此类推。换句话说，需要利用小球的z值给小球们分开层次，这也是即将要做的。在下面这个例子中，我们使用7个小球的运动来说明是如何实现层次的。

**小球3D空间（有层次）好多了！**

#### ****动画制作步骤****

1. 和上次的例子一样，重复6个步骤。不同的是初始化7个小球，并且把它们的x距离缩短。

for (var i = 0; i < 7; i++)  
{  
    var sphere = new Sphere();  
    sphere.x\_3d = -150+i\*40;  
    sphere.y\_3d = 80;  
    sphere.z\_3d = Math.random()\*(0-600)+600;  
    sphere.direction = 1;  
    sphere.speed = Math.random()\*(5-12)+12;      
  
    scene.addChild(sphere);  
}

[复制代码](javascript:void(0);)

2. 利用Bubble Sort算法，在每一次对小球x，y和大小设置后，对所有的小球在舞台上的层次进行操作。这里使用的理论是最小的z值的小球，它所在层次就应该在最上面。 不要小看我写的这短短几行代码哦！它可以把所有的小球分配到相应的层次！只要你的CPU够块，不管多少小球都可以。

function swap\_depth(container:Sprite)  
{  
    for (var i = 0; i < container.numChildren - 1; i++)  
    {  
        for (var j = container.numChildren - 1; j > 0; j--)  
        {  
            if (Object(container.getChildAt(j-1)).z\_3d < Object(container.getChildAt(j)).z\_3d)  
            {  
                container.swapChildren(container.getChildAt(j-1), container.getChildAt(j));  
            }  
        }  
    }  
}

[复制代码](javascript:void(0);)

3. 然后在循环函数里的最后加上

swap\_depth(scene);

本文章旨在讨论如何在Flash中实现3D动画，对于一些算法，比如这一节中使用到的冒泡排序算法，当然你可以选择使用Merge Sort等O(nlogn)算法以提高效率。你可以完全拷贝我写的代码去使用，但是请务必注明出处。如果是排序的话，google或者百度一下，我相信你 会找到更详细的说明文字来解释冒泡排序。

#### ****一个简单Particle System例子****

这是一个使用缩放的基本概念的例子，可以使用键盘上下左右键移动你的观看位置，按下W键加速前进，按下S键减速。由于涉及到摄像机，在这篇文章中就不再讲 解。源文件在本文的下载中可以找到，你可以尝试改变星星的数量，星星很多的话，移动摄像机，应该会看到一个很长的星带（题外话：在我的电脑里运行2000 个 星星就有些慢了，Dual Core 2.0GHZ，如果你有更强大的CPU的话，那你看到星空要比我的漂亮的多）。

**3D星空，简单的Particle System，WASD移动观看位置，K键加速，L键减速**

OK，你已经学会了如果使用缩放和一些简单的设置产生3D效果的动画。在后面的几篇中，我们会逐渐深入探讨更有趣的内容。加油！

http://www.cnblogs.com/yangzhou1030/archive/2008/10/30/1322592.html