## [Flash与3D编程探秘（三）- 摄像机（Camera）](http://www.cnblogs.com/yangzhou1030/archive/2008/11/01/1324242.html)

日期：2008年10月

在前面的两节中，你，作为观察者，所在空间的位置是一成不变的，物体在来回移动，让你产生了3D错觉。但是随着对3D的深入，会发现只让物体运动并 不足够。当讨论3D空间的时候，摄像机理论上代表3D中的一个点，我们从这个点去观看这个空间。为什么要使用摄像机呢？因为观察者希望能够透过一个镜头看 到其他所有舞台上的物体，对于他来说，它只需转动眼球就可以看到大千世界的另一面。（其实程序里摄像机只不过是一组数值来表示你的摄像机在3D空间的参 数，比如位置）想象一下你在广阔的撒哈拉沙漠里越野，又或者你的朋友小P走向在你的身旁。你的朋友小P慢慢走向你并最终到达离你很近的位置，几下来你们可 以交谈了。但是，如果小P站在原地不动，而是你走向他的身旁，那么对于地面来说，小P是不动的，而你（摄像机）是移动的。在这里你的眼睛就充当了摄像机。 看看下面的两个动画文件，再对比一下。左边是小P走向你，右边是你走向小P。

**移动物体和移动摄像机**

你会发现对于你的眼睛来说，你可以并不走动，只要把小P和周围一切的物体都移动到你的面前，也会达到同样的效果。但是哪一种可行呢？下面的两个动画说明了如何在3D空间中使用摄像机。动画效果如下，左面的是移动小P，右边是移动摄像机。

**对比移动小P和移动你的摄像机**

#### ****制作步骤：****

1. 和以前一样，定义原点，设置焦距，创建舞台。

// origin is the center of the view point in 3d space  
// everything scale around this point  
// these lines of code will shift 3d space origin to the center  
var origin = new Object();  
origin.x = stage.stageWidth/2;  
origin.y = stage.stageHeight/2-70;  
  
// focal length of viewer's camera  
var focal\_length = 400;

[复制代码](javascript:void(0);)

2. 下面定义一个摄像机物体，它具有3D空间的x，y，z，并且给它一个移动方向和初始的在z方向的移动速度。

// setup camera  
var camera = new Object();  
camera.x = 0;  
camera.y = 0;  
camera.z = 0;  
camera.direction = 1;  
camera.speed\_z = 6;

[复制代码](javascript:void(0);)

3. 创建一个小球在舞台上。

// create a sphere  
// go to library, right click on Sphere, choose linkage  
// and check Export for Actionscript  
for (var i = 0; i < 1; i++)  
{  
    var sphere = new Sphere();  
    sphere.x\_3d = -30;  
    sphere.y\_3d = 80;  
    sphere.z\_3d = 600;  
  
    // add all the spherees to the scene object  
    scene.addChild(sphere);  
}

[复制代码](javascript:void(0);)

4. 接下来开始写运动的循环函数。每一次执行函数一开始我们要把摄像机的位置在z方向移动一定的量。如果摄像机移动的离小球很近了的话，让摄像机向反方向移动。

// move the spherees back and forth  
function run(e:Event)  
{  
    // here we offset the camera position by its moving speed times the direction  
    camera.z += camera.speed\_z\*camera.direction;  
      
    if (camera.z > 600)                    // if the camera is too close to the ball  
    {  
        camera.z = 600;  
        camera.direction = -1;           // move camera backward  
    }  
    else if (camera.z < 0)                // if the camera is too close to the screen  
    {  
        camera.z = 0;  
        camera.direction = 1;  
    }  
    // loop through all the objects on the scene  
    for (var i = 0; i < scene.numChildren; i++)  
    {  
        // calculate the scale what the object should be  
        var scale = focal\_length/(focal\_length+scene.getChildAt(i).z\_3d-camera.z);  
        scene.getChildAt(i).x = (scene.getChildAt(i).x\_3d-camera.x)\*scale;  
        scene.getChildAt(i).y = (scene.getChildAt(i).y\_3d-camera.x)\*scale;  
        // properly scale the object to look 3d  
        scene.getChildAt(i).scaleX = scene.getChildAt(i).scaleY = scale;  
    }  
}

[复制代码](javascript:void(0);)

5. 计算出小球离摄像机的x距离，y距离和z距离，然后得出小球的缩放比率。最后把小球缩放并移动到相应的位置。That's it! 不要忘记添加循环函数执行事件。

this.addEventListener(Event.ENTER\_FRAME, run);

#### ****注意:****

你需要考虑让摄像机的z的指不能小于-1乘焦距，如果z小于这个值，那么公式scale = focal\_length/(focal\_length+z)得出的缩放比率会是负数，那么物体就会开始向后运动。

#### ****一个简单的赛车小游戏制作****

下面运用摄像机的概念来制作一个简单的赛车小游戏，游戏里你可以使用WASD键控制赛车，COOL！那么开始。

**简单赛车游戏，键盘WASD控制**

1. 定义原点，设置焦距，创建舞台。

// origin is the center of the view point in 3d space  
// everything scale around this point  
// these lines of code will shift 3d space origin to the center  
var origin = new Object();  
origin.x = stage.stageWidth/2;  
origin.y = stage.stageHeight/2;  
  
// now create a scene object to hold all the spheres  
var scene = new Sprite();  
this.addChild(scene);  
scene.x = origin.x;  
scene.y = origin.y;  
  
// focal length of viewer's camera  
var focal\_length = 400;

[复制代码](javascript:void(0);)

2. 下面定义一个摄像机物体，它具有3D空间的x，y，z，并且给它初始的在z方向的移动速度。

// setup camera  
var camera = new Object();  
camera.x = 0;  
camera.y = -40;                   // make the camera off the ground a little bit  
camera.z = 0;  
camera.speed\_z = 0;            // your driving speed

[复制代码](javascript:void(0);)

3. 创建两个个场景，一个用来盛放所有的赛车，另外一个盛放放有的路边轮胎。然后把它们添加到舞台上。

var tires = new Sprite();                // this sprite holds all the tires  
var cars = new Sprite();                // this sprite holds all the cars  
var txt\_speed = new TextField();    // your dashboard  
txt\_speed.x = 20;  
txt\_speed.y = 20;  
// now add them to the screen  
scene.addChild(tires);  
scene.addChild(cars);  
this.addChild(txt\_speed);

[复制代码](javascript:void(0);)

4. 定义一些赛车的运动状态的变量。

// now here are the variables determine the car's moving state  
var move\_left = false;  
var move\_right = false;  
var speed\_up = false;  
var brake = false;

[复制代码](javascript:void(0);)

5. 那么接下来创建40个轮胎并且把前20个放在路的左边，后20个放在路的右边，给赛道画出一个轮廓。

// now create 40 tires, 20 on the left and 10 on the right  
for (var i = 0; i < 40; i++)  
{  
    var tire = new Tire();  
    if (i < 20)  
    {  
        tire.x\_3d = -400;  
        tire.z\_3d = i\*500;  
    }  
    else  
    {  
        tire.x\_3d = 400;  
        tire.z\_3d = (i-20)\*500;  
    }  
    tire.y\_3d = 40;  
    tires.addChild(tire);  
}

[复制代码](javascript:void(0);)

6. 创建8个赛车，给它们相应的xyz位置（注意要赛车放在赛道上，设置它们的x范围在-230到230之间）不同的起始z位置和速度，最后添加到舞台上。

// create 8 opponent cars  
for (var j = 0; j < 8; j++)  
{  
    var car = new Car();  
    car.x\_3d = Math.random()\*(-230-230)+230;        // give them random x position  
    car.y\_3d = -30;  
    car.z\_3d = -800+(8-j)\*400;                              // give them speed  
    car.speed\_z = (8-j)\*15;  
    cars.addChild(car);  
}

[复制代码](javascript:void(0);)

7. 接下来要写一个函数，每一次执行这个函数，首先把赛车在z方向移动一定量（赛车相对地面是运动的），然后计算比率，把赛车移动到相应的位置并且缩放。我把 它命名updateCar，还是运用摄像机的理移动的基本知识，在每一个摄像机移动后，分别计算出摄像机与小车的xyz距离，然后把小车缩放和移动。注意 小车如果离摄像机太远或者被甩到摄像机的后面的话，让它不在屏幕上显示。

// for each of the running cycle, these two functions are called  
function updateCar(car)  
{  
    var x = car.x\_3d-camera.x;             // calculate the x distance between your camera and car  
    var y = car.y\_3d-camera.y;            // same we can y distance  
    var z = car.z\_3d-camera.z;            // and z distance  
      
    if (z < 0 || z > 10000)                   // if car is too far or left behind  
        car.visible = false;                    // then do not draw it  
    else  
        car.visible = true;  
          
    car.z\_3d += car.speed\_z;             // move the car  
    z = car.z\_3d-camera.z;                // recaculate the z distance  
      
    var scale = focal\_length/(focal\_length+z);     // caculate the scale what the car should be  
    car.x = x\*scale;  
    car.y = y\*scale;  
    car.scaleX = car.scaleY = scale;    // scale it to a proper size  
}

[复制代码](javascript:void(0);)

8. 轮胎的更新函数updateTire和赛车的更新函数类似，不同的是，轮胎相对地面是静止的，所以这里不改变它们的xyz值。如果轮胎已经到了摄像机后面（轮胎的z小于摄像机的z），把这个轮胎重新定位到摄像机前非常远的地方。

function updateTire(tire)  
{  
    var x = tire.x\_3d-camera.x;  
    var y = tire.y\_3d-camera.y;  
    var z = tire.z\_3d-camera.z;  
      
    if (z < 0)  
    {  
        tire.z\_3d += 10000;               // if the tire is left behind, then offset it   
        z = tire.z\_3d-camera.z;              
    }  
      
    var scale = focal\_length/(focal\_length+z);  
    tire.x = x\*scale;  
    tire.y = y\*scale;  
    tire.scaleX = tire.scaleY = scale;  
}

[复制代码](javascript:void(0);)

9. 下一步，run函数执行首先把摄像头沿z方向移动，然后调用前面写的updateCar和updateTire函数，刷新所有赛车和轮胎的位置和大小。

// here is the function loop  
function run(e:Event)  
{  
    camera.z += camera.speed\_z;                            // first move your camera  
      
    for (var i = 0; i < cars.numChildren; i++)              // update all the cars  
    {  
        updateCar(cars.getChildAt(i));  
    }  
    for (var j = 0; j < tires.numChildren; j++)             // and the tires on the side  
    {  
        updateTire(tires.getChildAt(j));  
    }  
      
    txt\_speed.text = int(camera.speed\_z) + " MPH";   // show your speed  
    txt\_speed.setTextFormat(new TextFormat("Verdana", 16, 0x444444, true));  
}

[复制代码](javascript:void(0);)

10. 下面是键盘响应事件函数，我写了一些的注释在程序里，不过我相信你应该很快就能看懂，就不打算详细解说了。实现的功能是当按下左键你的赛车左移；按下上键，赛车开始加速（当然极速是需要你来定义的了）等等。

// keyboard functions  
function key\_down(e:KeyboardEvent):void  
{  
    if (e.keyCode == 37)            // left key  
        move\_left = true;  
    if (e.keyCode == 39)            // right key  
        move\_right = true;  
    if (e.keyCode == 38)            // up key  
        speed\_up = true;  
    if (e.keyCode == 40)            // down key  
        brake = true;  
}  
function key\_up(e:KeyboardEvent):void  
{  
    if (e.keyCode == 37)  
        move\_left = false;  
    if (e.keyCode == 39)  
        move\_right = false;  
    if (e.keyCode == 38)  
        speed\_up = false;  
    if (e.keyCode == 40)  
        brake = false;  
}  
function keyboard\_response(e:Event):void  
{  
    if (move\_left)              
    {  
        // move the camera to the left, remember here the fast you go, the fast your steer  
        camera.x -= camera.speed\_z/6;  
        if (camera.x < -300) camera.x = -300;     // limit your car so it won't go off the road  
    }  
    if (move\_right)  
    {  
        camera.x += camera.speed\_z/6;  
        if (camera.x > 300) camera.x = 300;        // limit your car so it won't go off the road  
    }  
    if (speed\_up)  
    {  
        camera.speed\_z += .2;                         // accelerate  
        // limit the car speed in a range  
        if (camera.speed\_z < 0) camera.speed\_z = 0;              
        else if (camera.speed\_z > 120) camera.speed\_z = 120;  
    }  
    else  
    {  
        camera.speed\_z \*= .99;                      // if you don't hit the throttle, it will stop soon  
    }  
    if (brake)  
    {  
        camera.speed\_z -= .3;                       // slow down  
        if (camera.speed\_z < 0) camera.speed\_z = 0;  
    }  
}

[复制代码](javascript:void(0);)

11. 最后，添加循环函数执行和键盘响应事件。如果没问题的话，现在发布运行。成功了！

// now initialize all the necessary event listeners and we are done  
this.addEventListener(Event.ENTER\_FRAME, run);  
this.addEventListener(Event.ENTER\_FRAME, keyboard\_response);  
stage.addEventListener(KeyboardEvent.KEY\_DOWN, key\_down);  
stage.addEventListener(KeyboardEvent.KEY\_UP, key\_up);  
stage.stageFocusRect = false;  
stage.focus = scene;

[复制代码](javascript:void(0);)

#### ****注意：****

当你在循环一个数组中所有对象时，你可能会遇到想要删除一个对象的情况（可能你需要把这个对象从这个数组删除，然后添加到另外一个数组中），那么在 这个删除的过程中你要非常的小心，因为数组在你执行删除操作后的长度会改变，那么你如果循环使用数组长度作为循环次数的话，会造成跳过删除某个对象的现 象。

一种解决办法就是在循环之前定义一个变量然后再执行for循环。

var length = objects.numChildren;

目前为止，一直在讨论3D场景的设置，不要担心，从第七篇开始会讲到如何使用代码实现3D物体的绘制，不过我想在那之前我们还是多看几个3D场景的例子加深你的印象。那么，请不要失去耐心，最终你所关心的内容这里一定会有介绍的。

#### ****关于摄像机的焦距：****

程序中对摄像机初始化时，使用了一个变量focal\_length来对摄像机的镜头进行设置。这个焦距你可以简单理解为物体扭曲的比率，现实中，摄像时镜头焦距越大，那么拍出来的物体的空间扭曲就越小，反而物体在3D空间里的扭曲就越大，程序中也是一样。

 在这篇或者接下来的文章，我只会使用简单的一个变量focal\_length来代表摄像机镜头的设置（当然现实中摄像机的镜头操作要复杂的多，文 章中不再涉及，你可以自己添加镜头设置变量及操作） 。下面的动画里，你可以调节摄像机的镜头的焦距来观看物体空间扭曲的程度。

**调节摄像机镜头焦距**

http://www.cnblogs.com/yangzhou1030/archive/2008/11/01/1324242.html