cocos2d之Box2D详解 鼠标关节实现

DionysosLai 2014-5-7

我们经常要移动物理世界中的某个物体，比方说石头、木块等。如果我们直接改变这些物体的位置，让这些物体跟随我们手指移动，即使这样是可行的，却违反了物理世界的基本规则。这个世界没有“上帝之手”。

换个思路，如果我们要移动物体，那是否说，就是我们要在物体上施加一个某个方向的无穷大的力量。恩，没错，就是这样。在Box2D中，有一个比较特殊的关节类型：鼠标关节（Mouse Joint），之所以特殊，就是因为它并不是物理世界中原生的物体，是来自于用户的操作。鼠标关节，可以试图将物体拖向当前鼠标光标的位置，同时在选择方向上没有限制。

使用鼠标关节一般有三个步骤：

1. 创建（在touchBegan中）
2. 改变鼠标关节位置（在touchMove中）；
3. 销毁鼠标关节（在touchEnd中）

按照上面步骤：我们一步步创建鼠标关节：

在头文件中创建一个鼠标关节：

b2MouseJoint\* m\_mouseJoint;

然后在cpp文件中初始鼠标关节为NULL:

m\_mouseJoint = NULL;

下面就是创建鼠标关节：

b2Vec2 vec(m\_pTouchPoint.x/PTM\_RATIO,m\_pTouchPoint.y/PTM\_RATIO);

// b2Vec2 vec = b2Vec2(touchPoint.x,touchPoint.y);

if(m\_mouseJoint != NULL)

{

return false;

}

// Make a small box.

b2AABB aabb;

b2Vec2 d;

d.Set(0.001f, 0.001f);

aabb.lowerBound = vec - d;

aabb.upperBound = vec + d;

b2BodyDef bodyDef;

b2Body \*m\_groundBody = m\_world->CreateBody(&bodyDef);

// Query the world for overlapping shapes.

QueryCallback callback(vec);

m\_world->QueryAABB(&callback, aabb);

if (callback.m\_fixture)

{

b2Body\* body = callback.m\_fixture->GetBody();

b2MouseJointDef md;

md.bodyA = m\_groundBody;//一般为世界边界

md.bodyB = body;//需要拖动的物体

md.target = vec;

md.maxForce = 1000.0f \* body->GetMass();

m\_mouseJoint = (b2MouseJoint\*)m\_world->CreateJoint(&md);

body->SetAwake(true);

CCLog("touch bengin \n");

return true;

}

return false;

在这里，调用了一个回调函数，因此必须在头文件中，新建一个类：

class QueryCallback : public b2QueryCallback

{

public:

QueryCallback(const b2Vec2& point)

{

m\_point = point;

m\_fixture = NULL;

}

bool ReportFixture(b2Fixture\* fixture)

{

b2Body\* body = fixture->GetBody();

if (body->GetType() == b2\_dynamicBody)

{

bool inside = fixture->TestPoint(m\_point);

if (inside)

{

m\_fixture = fixture;

// We are done, terminate the query.

return false;

}

}

// Continue the query.

return true;

}

b2Vec2 m\_point;

b2Fixture\* m\_fixture;

};

这里要注意一个问题，就是设置鼠标关节边界时：

md.bodyA = m\_groundBody;//一般为世界边界

md.bodyB = body;//需要拖动的物体

md.bodyA是我们的世界

下面就是在touchMove中改变鼠标关节的属性，代码如下：

m\_iTouchType = TOUCH\_MOVE;

CCPoint point = pTouch->getLocation();

m\_pTouchPoint = point;

if(m\_mouseJoint == NULL )

return;

b2Vec2 vecMouse;

vecMouse.Set((m\_pTouchPoint.x)/PTM\_RATIO, (m\_pTouchPoint.y)/PTM\_RATIO);

//改变关节位置.

m\_mouseJoint->SetTarget(vecMouse);

最后一个当我们手指离开屏幕时，我们要销毁我们所创建的鼠标关节，在ccTouchEnded加入如下代码：

m\_iTouchType = TOUCH\_END;

CCPoint point = pTouch->getLocation();

m\_pTouchPoint = point;

CCLOG("%f, %f", point.x, point.y);

//销毁关节.

if(m\_mouseJoint != NULL)

{

m\_world->DestroyJoint(m\_mouseJoint);

m\_mouseJoint =NULL;

}

好，到目前一切就ok了。