**Getting Started - Chapter 4 - Avoiding a Car Crash**

**@@启程-第四章-阻止一场车祸@@**

**Avoiding a Car Crash**

**@@阻止一场车祸@@**

The simplest way of seeing if two meshes are in contact is to use the *intersectsMesh* method, as in

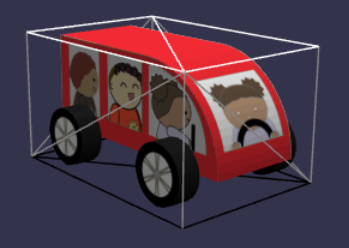
@@最简单的判断两个网格是否相交的方式是使用“*网格相交*”方法，就像这样：@@

mesh1.intersectMesh(mesh2)

which will be true if a box bounding mesh1 would overlap with a box bounding mesh2. Each mesh has a built-in bounding box which lies close to the surface of the mesh that is used in checking the intersection of the meshes.

@@如果网格1的边界盒与网格2的边界盒重合，则这个方法返回true。每个网格都有一个内置的边界盒，这个边界盒紧贴着网格的表面，可以用来检测这个网格与其他网格的相交情况。@@





Since the character's walk and the car's journey are not phased together there will be a time when they are in the same place. However it is not possible to predict when the character, taking its long walk around the village, and the car, on its short journey, might intersect. In order to demonstrate the *intersectsMesh* method we will make the character walk backwards and forwards across the stopping place of the car.

@@因为角色的行走和小车的游弋都不会停止，总会有某个时刻二者处于相同位置。但是我们没有办法预计，这个绕村庄行走的角色和这个短途游览的小车何时会相遇。为了演示“*网格相交*”方法的效果，我们将让角色前后走动穿过小车的停止区。@@

In our case we want the character to stop moving if the car is in a "hit" zone and the character is not. It would, after all, be dangerous for the character to stop if they are both in the danger zone. In our case because of the way the *Dude* is constructed we need to use one of its children to check intersection. Basically *Dude* is just a holder node for the head, torso, legs and arms and the box bounding it is too small to be effective in this case..

@@在我们的计划中，我们设置了一个“碰撞”区域，当车辆进入碰撞区域，而角色没有进入时，让角色停止运动（译者注：并没有直接检测角色网格和车网格的相交，而是根据二者的运动轨迹设计了一个第三方碰撞区！）。毕竟如果车和人都在碰撞区域内时，人停止移动会很危险，所以人处于碰撞区内时仍会移动。在我们的例子中考虑到这个老哥模型的建模方式，我们需要使用它的一个子元素（索引1）进行相交检测。简单来说老哥模型的顶层元素只是模型头部的一个表示模型位置的节点，而躯干、腿、胳膊等子元素的边界盒又比较小，不足以代表整个模型，所以选择索引1的子元素。（译者注：这个模型建模时就设置了父子元素关系，而父元素是一个没有顶点的“空网格”，只负责保有位置、姿态等信息）@@

Basic Collision Detection

基础碰撞检测

https://playground.babylonjs.com/#KBS9I5#83

Now make the hit box invisible.

@@现在让碰撞盒不可见@@

Basic Collision Detection Invisible Box

使用不可见碰撞盒的基础碰撞检测

https://playground.babylonjs.com/#KBS9I5#84

This repeated animation looks a bit out off place. Even if we revert back to the character walking around the village the repetition of the car not only looks a bit silly it is also annoying. Let's improve the environment a little so that the car can appear to be driving through the village. We will place the village in a valley with distant hills created from a height map and put a road in for the car to travel along. While we are doing that let's add a sky and some distant trees.

@@这个重复动画看起来有点不太合适。即使我们恢复到角色绕村庄行走，小车的重复动画看以来还是会有些傻并且令人不适。让我们来改进一下环境，让小车看起来是合理地驶过这个村庄的。我们将把这个村庄放在一个峡谷中，并且远处有通过“高度图”建立的丘陵，接着在丘陵中为小车安排一条移动路线。在做这件事的同时我们还要为场景添加天空和一些远处的树。@@

const createScene = function () {

    const scene = new BABYLON.Scene(engine);

    const camera = new BABYLON.ArcRotateCamera("camera", -Math.PI / 2.2, Math.PI / 2.2, 15, new BABYLON.Vector3(0, 0, 0));

    camera.attachControl(canvas, true);

    const light = new BABYLON.HemisphericLight("light", new BABYLON.Vector3(1, 1, 0));

    const wireMat = new BABYLON.StandardMaterial("wireMat");

wireMat.wireframe = true;

// wireMat.alpha = 0;

    const hitBox = BABYLON.MeshBuilder.CreateBox("carbox", {width: 0.5, height: 0.6, depth: 4.5});

    hitBox.material = wireMat;

    hitBox.position.x = 3.1;

    hitBox.position.y = 0.3;

    hitBox.position.z = -5;

    let carReady = false;

    BABYLON.SceneLoader.ImportMeshAsync("", "https://assets.babylonjs.com/meshes/", "car.glb").then(() => {

        const car = scene.getMeshByName("car");

        carReady = true;

        car.rotation = new BABYLON.Vector3(Math.PI / 2, 0, -Math.PI / 2);

        car.position.y = 0.16;

        car.position.x = -3;

        car.position.z = 8;

        const animCar = new BABYLON.Animation("carAnimation", "position.z", 30, BABYLON.Animation.ANIMATIONTYPE\_FLOAT, BABYLON.Animation.ANIMATIONLOOPMODE\_CYCLE);

        const carKeys = [];

        carKeys.push({

            frame: 0,

            value: 8

        });

        carKeys.push({

            frame: 150,

            value: -7

        });

        carKeys.push({

            frame: 200,

            value: -7

        });

        animCar.setKeys(carKeys);

        car.animations = [];

        car.animations.push(animCar);

        scene.beginAnimation(car, 0, 200, true);

        //wheel animation

        const wheelRB = scene.getMeshByName("wheelRB");

        const wheelRF = scene.getMeshByName("wheelRF");

        const wheelLB = scene.getMeshByName("wheelLB");

        const wheelLF = scene.getMeshByName("wheelLF");

        scene.beginAnimation(wheelRB, 0, 30, true);

        scene.beginAnimation(wheelRF, 0, 30, true);

        scene.beginAnimation(wheelLB, 0, 30, true);

        scene.beginAnimation(wheelLF, 0, 30, true);

    });

    BABYLON.SceneLoader.ImportMeshAsync("", "https://assets.babylonjs.com/meshes/", "village.glb");

    const walk = function (turn, dist) {

        this.turn = turn;

        this.dist = dist;

    }

    const track = [];

    track.push(new walk(180, 2.5));

    track.push(new walk(0, 5));

    // Dude

    BABYLON.SceneLoader.ImportMeshAsync("him", "/scenes/Dude/", "Dude.babylon", scene).then((result) => {

        var dude = result.meshes[0];

        dude.scaling = new BABYLON.Vector3(0.008, 0.008, 0.008);

        dude.position = new BABYLON.Vector3(1.5, 0, -6.9);

        dude.rotate(BABYLON.Axis.Y, BABYLON.Tools.ToRadians(-90), BABYLON.Space.LOCAL);

        const startRotation = dude.rotationQuaternion.clone();

        scene.beginAnimation(result.skeletons[0], 0, 100, true, 1.0);

        let distance = 0;

        let step = 0.015;

        let p = 0;

        scene.onBeforeRenderObservable.add(() => {

            if (carReady) {

                if (!dude.getChildren()[1].intersectsMesh(hitBox) && scene.getMeshByName("car").intersectsMesh(hitBox)) {

                    return;

                }

            }

            dude.movePOV(0, 0, step);

            distance += step;

            if (distance > track[p].dist) {

                dude.rotate(BABYLON.Axis.Y, BABYLON.Tools.ToRadians(track[p].turn), BABYLON.Space.LOCAL);

                p +=1;

                p %= track.length;

                if (p === 0) {

                    distance = 0;

                    dude.position = new BABYLON.Vector3(1.5, 0, -6.9);

                    dude.rotationQuaternion = startRotation.clone();

                }

            }

        })

    });

    return scene;

};