**charactor controller角色平衡木问题解决办法**

Posted on 2013年06月13日 by U3d / [Unity3D脚本/插件](http://www.unitymanual.com/category/script)/被围观 37 次

charactor controller角色做平衡木的问题，和小球走平衡木不一样，charctor controller是不可以对地面产生压力的，无论角色是否加了刚体，charactor controller都只具有自由下落，而不具有对地面压力。如果换成box collider等碰撞检测，上下楼梯角色运动的时候很容易倒下。针对这个问题，做出了以下解决方案：

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |
| --- | --- |
| 01 | **var** speed = 10.0; |
| 02 | **var** gravity = 10.0; |
| 03 | **var** maxVelocityChange = 10.0; |
| 04 | **var** canJump = **true**; |
| 05 | **var** jumpHeight = 2.0; |
| 06 | **private** **var** grounded = **false**; |
| 07 |  |
| 08 | @script RequireComponent(Rigidbody, CapsuleCollider) |
| 09 |  |
| 10 | function Awake () |
| 11 | { |
| 12 | rigidbody.freezeRotation = **true**; |
| 13 | rigidbody.useGravity = **false**; |
| 14 | } |
| 15 |  |
| 16 | function FixedUpdate () |
| 17 | { |
| 18 | **if** (grounded) |
| 19 | { |
| 20 | *// Calculate how fast we should be moving* |
| 21 | **var** targetVelocity = new Vector3(Input.GetAxis("Horizontal"), 0, Input.GetAxis("Vertical")); |
| 22 | targetVelocity = transform.TransformDirection(targetVelocity); |
| 23 | targetVelocity \*= speed; |
| 24 |  |
| 25 | *// Apply a force that attempts to reach our target velocity* |
| 26 | **var** velocity = rigidbody.velocity; |
| 27 | **var** velocityChange = (targetVelocity - velocity); |
| 28 | velocityChange.x = Mathf.Clamp(velocityChange.x, -maxVelocityChange, maxVelocityChange); |
| 29 | velocityChange.z = Mathf.Clamp(velocityChange.z, -maxVelocityChange, maxVelocityChange); |
| 30 | velocityChange.y = 0; |
| 31 | rigidbody.AddForce(velocityChange, ForceMode.VelocityChange); |
| 32 |  |
| 33 | *// Jump* |
| 34 | **if** (canJump && Input.GetButton("Jump")) |
| 35 | {Unity3D教程手册 |
| 36 | rigidbody.velocity = Vector3(velocity.x, CalculateJumpVerticalSpeed(), velocity.z); |
| 37 | } |
| 38 | } |
| 39 |  |
| 40 | *// We apply gravity manually for more tuning control* |
| 41 | rigidbody.AddForce(Vector3 (0, -gravity \* rigidbody.mass, 0)); |
| 42 |  |
| 43 | grounded = **false**; |
| 44 | } |
| 45 |  |
| 46 | function OnCollisionStay () |
| 47 | { |
| 48 | grounded = **true**; |
| 49 | } |
| 50 |  |
| 51 | function CalculateJumpVerticalSpeed () |
| 52 | { |
| 53 | *// From the jump height and gravity we deduce the upwards speed* |
| 54 | *// for the character to reach at the apex.* |
| 55 | **return** Mathf.Sqrt(2 \* jumpHeight \* gravity); |
| 56 | } |