动画导入与控制

杨铭 - 5130379022

2016 年 4 月 12 日

目录

1 概述

本次作业主要使用 **Unity**和 **Blender**进行了简单的动画制作和导入控制。由于人物骨架动画的制作较为复杂，我使用了第三方的资源，而物体的动画为自己制作。

2 资源说明

本次主要使用了两个外部导入的模型、动作资源：

- Unity-Chan

来自 Unity 社区 Assets Store 的 3D 模型 + 动画资源Unity-Chan是一个包含模型、动画、动画控制和三个预设场景的资源包。我只使用了它的模型、动作和摄像机控制脚本。

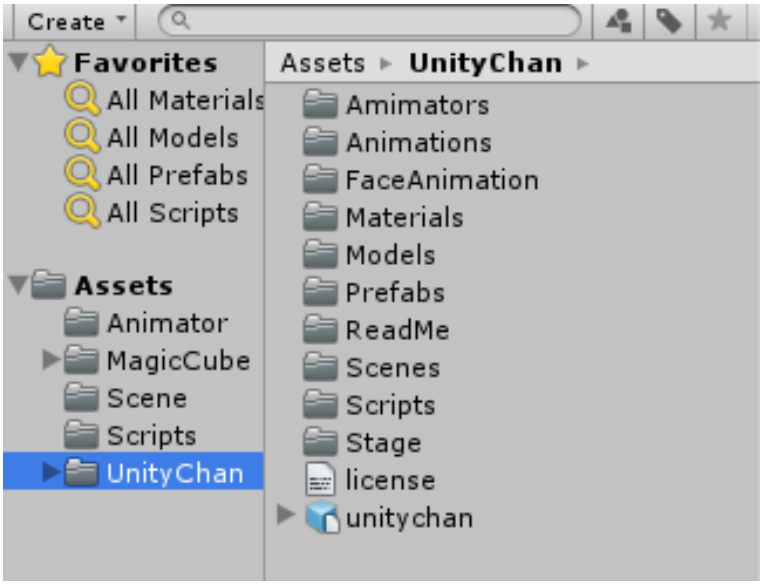

Figure 1: Unity-Chan

- MagicCube

这是一个我自己用 Blender 做的几个立方体运动的动画，通过位移，旋转设置关键帧，最后导出为.FBX 文件

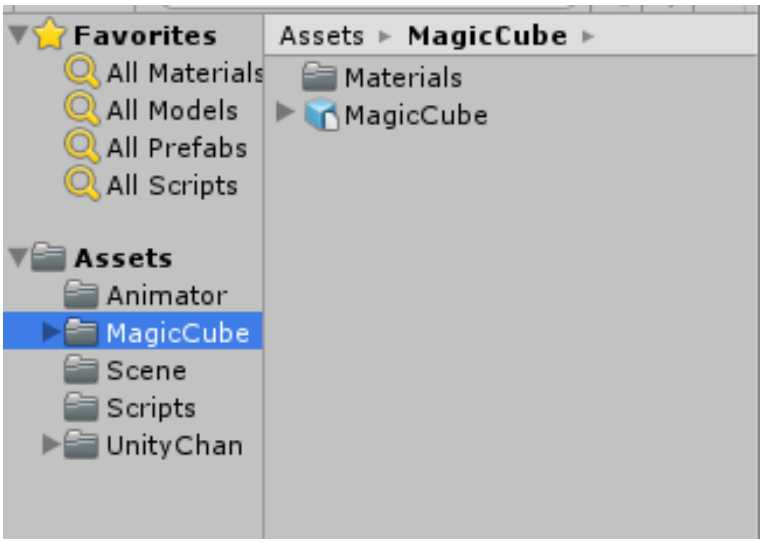

Figure 2: MagicCube1

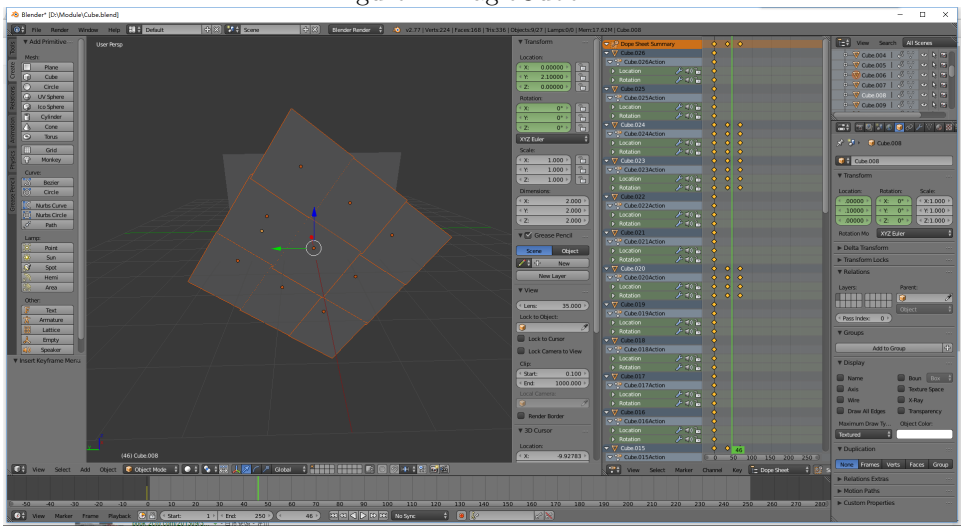

Figure 3: MagicCube2

在 blender 的导出时要注意将所有物体设置为某一个物体的子物体，并在导出设置的 Animation 里勾掉 **all actions**选项，否则将会导出所有物体的所有动作，不方便在 Unity 里的设置。

3 人物动画

3.1 简单动画导入

首先将模型拖入场景，为它添加一个 **Animator Controller**，并将资源中自带的动画拖入 Animator 中

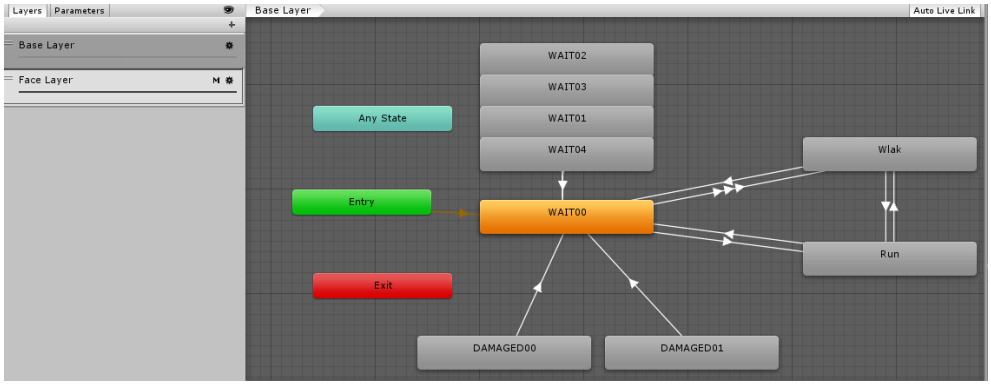

Figure 4: Animator

Unity 的动画可以使用 **Animator**这样一种状态机来控制，状态机可以有多层 -**Layer**，每两个状态之间可以有多种转换 -**Transistion**，每个转换也可以设置约束条件 -**Conditions**

我使用了 Unity-Chan 的五个静止动作 **WAIT00**，**WAIT01**，**WAIT02**，**WAIT03**，**WAIT04**，将 **WAIT00**设置为 **Layer Default State**。这样在这一层中将会默认播放这个动画。

再为其他四个动作每个做一个指向 **WAIT00**的 **Transistion**，不设置约束条件，这样如果播放完了其余四个动作，将自动转换到 **WAIT00**动作。

最后在控制脚本 **PlayerController**中写上用数字键控制播放这几个动作的代码

```
void Update () {
    // Play Cute
    if(Input.GetKeyDown(KeyCode.Keypad1) || Input.GetKeyDown(KeyCode.Alpha1))
    {
        anim.Play("WAIT01", -1, 0f);
    }
    if (Input.GetKeyDown(KeyCode.Keypad2) || Input.GetKeyDown(KeyCode.Alpha2))
    {
        anim.Play("WAIT02", -1, 0f);
    }
    if (Input.GetKeyDown(KeyCode.Keypad3) || Input.GetKeyDown(KeyCode.Alpha3))
    {
        anim.Play("WAIT03", -1, 0f);
    }
    if (Input.GetKeyDown(KeyCode.Keypad4) || Input.GetKeyDown(KeyCode.Alpha4))
    {
        anim.Play("WAIT04", -1, 0f);
    }
```

Figure 5: Code1

到这里，基础的动作导入和控制就完成了！

3.2 混合动画制作

许多模型时自带向前、向后、向左、向右行走的动作，但是对于一个任意的方向，用这些动作来行走将比较奇怪。Unity 中使用 **Blend Tree**的混合动作来实现这一类功能。

在 **Animator**面板中创建一个新的 **Blend Tree**，命名为 **Walk**, 将它的 **Blend Type**设置为 **2D Simple Directional**。这样混合的动作将会在一个二维坐标系里根据参数的比例来混合。

然后将四个方向的行走动作设置好，并新建两个参数作为混合比例的输入。接着添加两个从 **WAIT00**到 **Walk**的 **Transistion**，一个从 **Walk**到 **WAIT00**的 **Transistion**

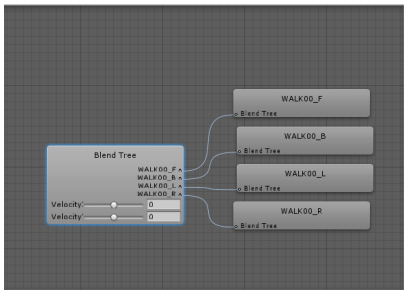

Figure 6:

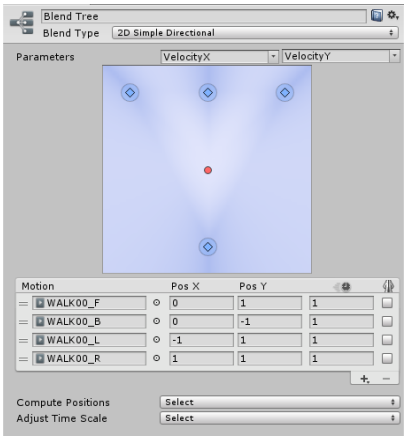

Figure 7:

我们希望用按键输入来控制行走动画的播放，这需要为 **Walk**状态的转换设置条件。将 Y 轴的输入大小作为约束条件：如果它是一个大于 0.1 或者小于 -0.1，进入 **Walk**状态，否则回到 **WAIT00**状态。

最后在代码中写好输入的接受，并用 **Animator**的 **SetFloat**函数设置状态机中的参数

```
InputX = Input.GetAxis("Horizontal");
InputY = Input.GetAxis("Vertical");
anim.SetFloat("VelocityX", InputX);
anim.SetFloat("VelocityY", InputY);
```

Figure 8:

然后如法炮制制作了一个 **Run**的动画，使用 Left-Shift 来控制

3.3 分层动画

要实现两个不同位置动画同时播放而又不互相干扰，用 **Blend Tree**就不太合适。这种情况需要给状态机设置多层状态，并使用 **Mask**来标记动画改变模型的范围

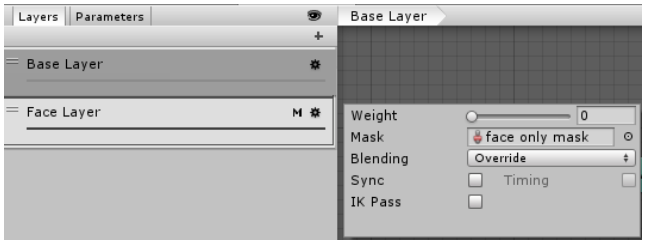

Figure 9:

使用 **Unity-Chan**中制作好的脸部动作和脸部标记，并且使用 **FaceController**脚本来控制面部表情动画

```
public class FaceController : MonoBehaviour {

    public AnimationClip[] animations;

    Animator anim;

    public float delayWeight;

    // Use this for initialization
    void Start () {
        anim = GetComponent<Animator>();
    }

    void OnGUI()
    {
        foreach (var animation in animations)
        {
            if (GUILayout.Button(animation.name))
            {
                anim.CrossFade(animation.name, 0);
            }
        }
    }

    float current = 0;

    // Update is called once per frame
    void Update () {
        if (Input.GetMouseButton(0))
        {
            current = 1;
        }
        else {
            current = Mathf.Lerp(current, 0, delayWeight);
        }
        anim.SetLayerWeight(1, current);
    }
```

Figure 10:

这样即可实现脸部动作和身体动作同时播放而又不互相影响的效果。

Figure 11:

Figure 12:

4 物品动画

4.1 制作

使用 Blender 的动画功能制作简单的物体动画 **MagicCube**。动画的主题是一个 2X2X2 的魔方，首先创建 8 个 Cube 排列成魔方。然后在时间轴上选中帧数，移动、旋转或者改变 Cube 的大小，然后插入关键帧。制作完成后导出.FBX 文件拖入 Unity 里即可使用。

4.2 控制

由于导出时设定了将动画合并成一个 Action，所以模型虽然有 8 个方块，但是控制起来十分方便。

Figure 13:

在 Unity 的 assets 中可以看到导入的动画带有两个 **Clips**，即两个动画切片，可以分开使用。**EmptyAction**是一个空的动作，什么也没做，而**Cube.010Action**就是刚刚制作好的动画。

制作一个简单的状态机 **Animator**控制这个模型即可。

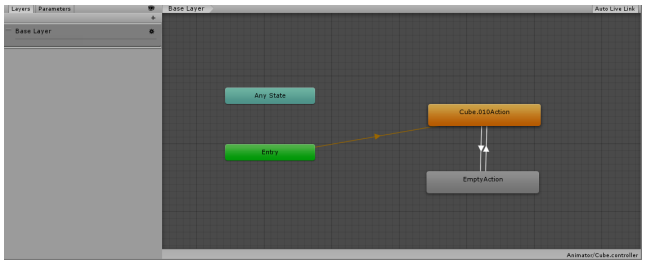

Figure 14:

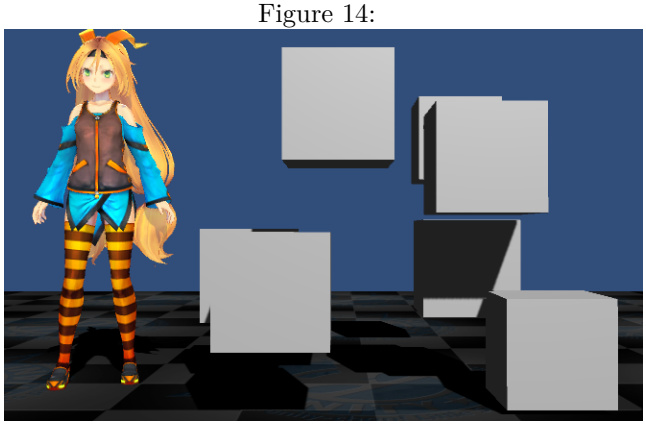

Figure 15:

5 总结

这次作业让我学到了建模软件动画的制作和 Unity 对导入动画的控制，熟悉了动画控制的整个流程，并对 **Blend Tree**、**Mask**等机制有所了解，也稍微熟悉了 Blender 的操作。

实际上最开始我准备自己做一个人物（双足）模型来完成这次作业的，结果建模就用了我 3 天时间，结果做好了骨架没时间做动画，果然建模是个功夫活。希望我的模型能出现在后面的游戏大作业中。

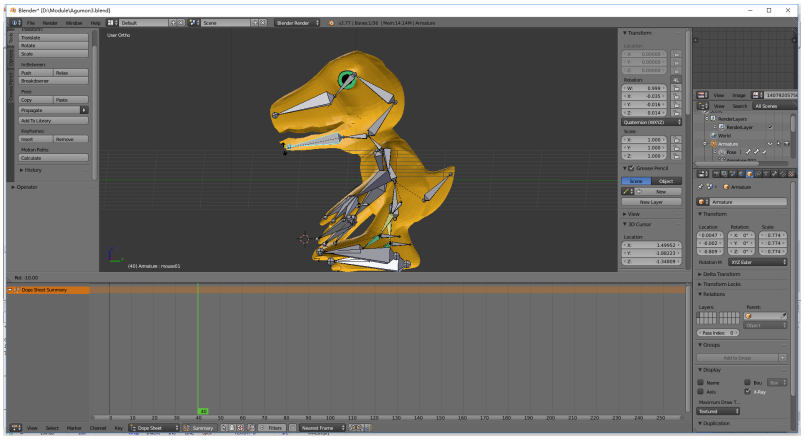

Figure 16: 未完成的亚古兽