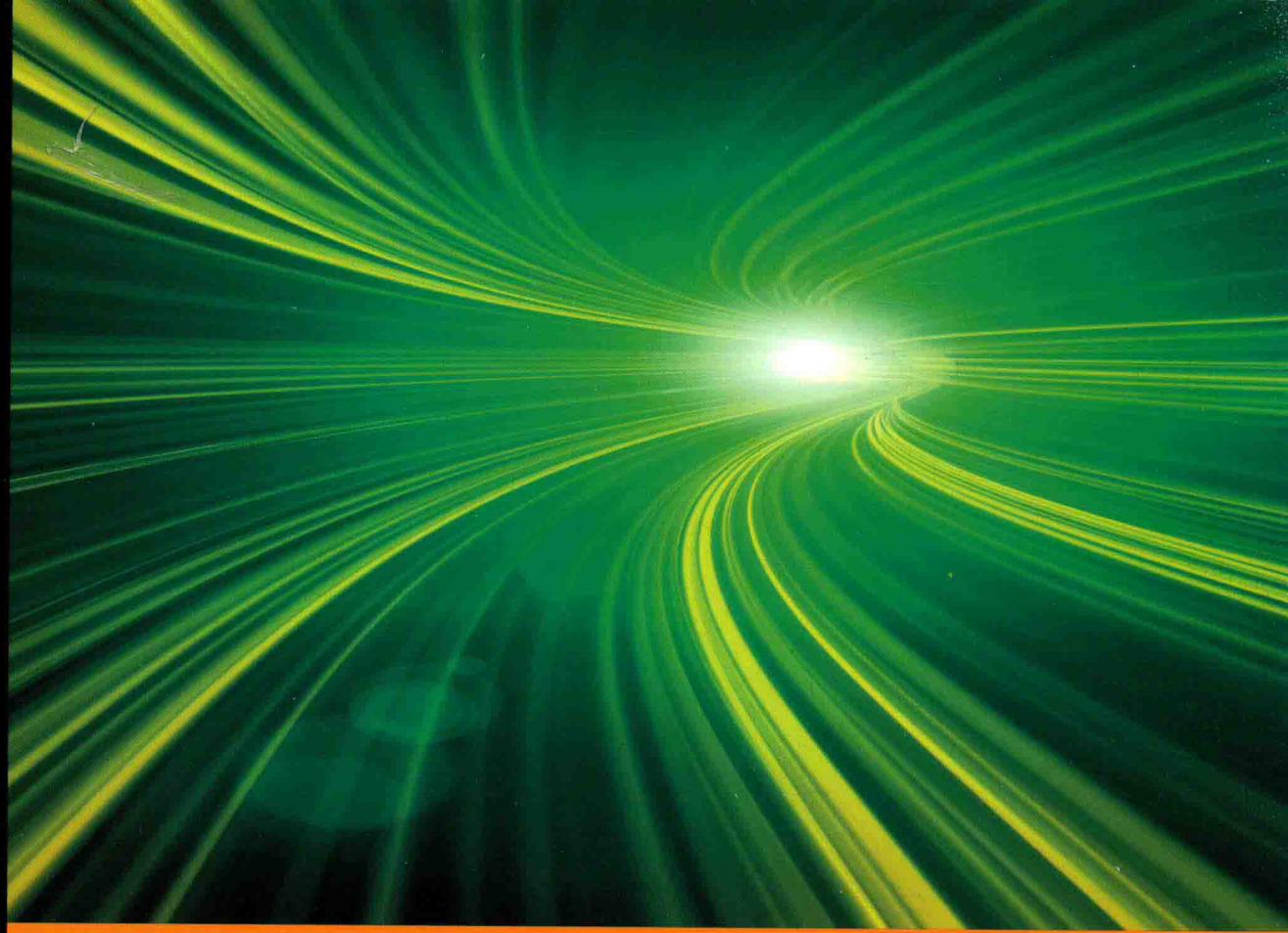

[美]Alan Thorn 著 李强 译

Unity游戏动画设计

Unity Animation Essentials

清华大学出版社

Unity 游戏动画设计

[美] Alan Thorn 著

李　强　译

清华大学出版社

北　京

内 容 简 介

本书详细阐述了与 Unity 游戏动画相关的基本解决方案，主要包括动画基础知识、精灵对象动画、本地动画、基于 Mecanim 的非人物角色动画、游戏角色动画的基础知识、高级角色动画、形状混合、IK 和电影纹理等内容。此外，本书还提供了相应的示例、代码，以帮助读者进一步理解相关方案的实现过程。

本书适合作为高等院校计算机及相关专业的教材和教学参考书，也可作为相关开发人员的自学教材和参考手册。

北京市版权局著作权合同登记号 图字：010-2016-5198

图书在版编目（CIP）数据

Unity 游戏动画设计/（美）艾伦·索恩（Alan Thorn）著；李强译. —北京：清华大学出版社，2016
书名原文：Unity Animation Essentials
ISBN 978-7-302-44266-0

I. ①U… II. ①艾… ②李… III. ①游戏程序-程序设计 IV. ①TP311.5

中国版本图书馆 CIP 数据核字（2016）第 153181 号

责任编辑：贾小红
封面设计：刘　超
版式设计：魏　远
责任校对：王　云
责任印制：王静怡

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：三河市君旺印务有限公司
装 订 者：三河市新茂装订有限公司
经　　销：全国新华书店
开　　本：185mm×230mm　　**印　　张**：9.25　　**字　　数**：191 千字
版　　次：2016 年 10 月第 1 版　　**印　　次**：2016 年 10 月第1 次印刷
印　　数：1～3000
定　　价：39.00 元

产品编号：068951-01

译 者 序

计算机图形技术以及计算机动画在视觉效果和动画制作方面曾引发了重大的技术变革，然而，这一前进步伐仍未停止，数字技术进一步拓展了广大观众群体。由于娱乐业内人士对视觉效果的不断追求，各家公司均对其作品中的图形效果提出了更高的要求。当前，建模、渲染以及动画技术日趋成熟，硬件运算速度不断提升而其价格却不断下降，高质量的计算机图形效果在电影制作过程中，计算机图形方案则使得外星生物看上去更加栩栩如生。

Unity 是近几年非常流行的一个 3D 游戏开发引擎（特别是移动平台），它的特点是跨平台能力强，支持 PC、Mac、Linux、网页、iOS、Android 等几乎所有的平台，移植便捷，3D 图形性能出众，为众多游戏开发者所喜爱。在手机平台，Unity 几乎成为 3D 游戏开发的标准工具。

本书则在此基础上将二者进行有机的结合，并讨论较为高级的开发技术和解决方案。

在本书的翻译过程中，除李强外，朱利平、张欣欣、姚楷峰、王晓晓、王巍、王梅、王典、孙年果、史云龙等人也参与了部分翻译工作，在此一并表示感谢。

限于译者的水平，译文中难免有错误和不妥之处，恳请广大读者批评指正。

译　　者

前　言

动画在游戏中扮演了重要的角色，例如，简单方案中处于运动状态的飞船，以及复杂方案中的面部动画。如果缺乏动画的支持，则难以建立完整的游戏和真正意义上的交互式体验。针对实时动画的构建，本书将逐步探讨 Unity 提供的特征集。同时，本书假设读者并不具备与动画相关的背景知识，但要求读者对 Unity 引擎有所了解，例如，关卡设计、基本的界面应用，以及基础的 C#编码知识。作为起点，本书将逐步对动画特征所涉及的内容进行梳理，并讨论真实场景中的具体应用，进而高效地实现最终的动画效果。

本书内容

第 1 章主要讨论动画的核心概念，其中涉及多项内容以及 Unity 中的动画实例，特别是脚本机制，例如 deltaTime、动画曲线和动画 Mesh UV。整体上讲，本章涵盖了动画入门知识，以及需要了解的各项特征。

第 2 章介绍 2D 动画场景、用于创建 2D 游戏的精灵对象特征集合以及平面动画。其中包含了 Unity Sprite 编辑器、动画帧、帧速率以及常见问题的解决方案。

第 3 章针对 2D 和 3D 场景，从广义角度上探讨了 Unity 的动画操作，并使用了 Animation 窗口和粒子系统。其中涉及两个较为实用的项目，即关卡中的漫游相机和灰尘/萤火虫粒子系统，这在某些幻想系列游戏中十分常见。

第 4 章阐述 Unity 的核心特性 Mecanim。Mecanim 表示为一类特征集，并可生成更为高级的平滑动画，对于游戏角色而言尤其如此。本章将考察相对特殊的 Mecanim 应用，即门对象的开启/闭合动画，以及关卡中的其他交互元素。

第 5 章将分析骨骼化角色动画，并考察骨骼以及角色间的动画方式。除此之外，本章还将介绍骨骼化角色的导入和最优配置方式，以供后续动画操作使用。

第 6 章从逻辑上讲与第 5 章关系紧密，使用了导入、配置完毕的 Unity 游戏角色，并于随后作为玩家控制的角色对其实现动画操作。对应角色支持基于行走和跑动的自然态和前向运动，以及基于左、右运动的旋转动画。

第 7 章包含用于创建面部动画以及其他变形运动的 BlendShapes，用于角色脚部和手部实时定位的逆向动力学知识，以及作为投影至几何体的、用于播放电影文件的电影纹理。

背景知识

本书涵盖了真实的动画开发项目，以及可供读者使用的配套资源文件。读者需要获取一份最新版本的 Unity，在本书编写时，最新版本为 Unity 5。读者可从 Unity 网站上进行下载，免费获取 Unity 5 个人版，对应网址为 https://unity3d.com/。除了 Unity 之外，还需要创建游戏角色模型和其他动画 3D 资源，因而需要使用 3D 建模和动画软件，例如 3ds Max、Maya 或 Blender。读者可免费下载 Blender，对应网址为 http://www.blender.org/。本书主要讨论基于 Unity 的动画设计。

适用读者

如果读者已对 Unity 有基本了解并希望进一步扩充自己的知识，即学习与实时动画相关的内容，本书将是不错的选择。本书假设读者曾使用 Unity 构建过简单的关卡，并可编写较为基础的脚本文件，同时希望进一步了解高级动画概念。例如，读者可能希望了解基于 Mecanim 的、简单高效的动画实现方式。

本书约定

本书涵盖了多种文本风格，进而对不同类型的信息加以区分。下列内容展示了对应示例及其具体含义。

文本中的代码、数据库表名称、文件名称、文件名、文件扩展名、路径名、伪 URL、用户输入以及 Twitter 用户名采用如下方式表示：

"By default, all new scripts are created with two functions, Start and Update."

代码块则通过下列方式设置：

```
using UnityEngine;
using System.Collections;
public class MoviePlay : MonoBehaviour
{
  //Reference to movie to play
  public MovieTexture Movie = null;
  // Use this for initialization
  void Start ()
  {
    //Get Mesh Renderer Component
    MeshRenderer MeshR = GetComponent<MeshRenderer>();
```

```
    //Assign movie texture
    MeshR.material.mainTexture = Movie;
    GetComponent<AudioSource>().clip = Movie.audioClip;
    GetComponent<AudioSource>().spatialBlend=0;
    Movie.Play();
    GetComponent<AudioSource>().Play();
  }
}
```

如果代码中的部分内容希望引起读者的注意，则相关代码行采用黑体显示：

```
using UnityEngine;
using System.Collections;
public class Mover : MonoBehaviour
{
  // Use this for initialization
  void Start () {
  }
  // Update is called once per frame
  void Update ()
  {
    //Transform component on this object
    Transform ThisTransform = GetComponent<Transform>();
    //Add 1 to x axis position
    ThisTransform.position += new Vector3(1f,0f,0f);
  }
}
```

注意：表示较为重要的概念，而**提示**：则表示相关操作技巧。

读者反馈和客户支持

欢迎读者将对本书的建议或意见予以反馈，以进一步了解读者的阅读喜好。反馈意见对于我们来说十分重要，以便改进我们日后的工作。

读者可向 feedback@packtpub.com 发送邮件，并以书名作为邮件标题。

若读者针对某项技术具有专家级的见解，抑或计划撰写书籍或完善某部著作的出版工作，则可阅读 www.packtpub.com/authors 中的 author guide 一栏。

我们将对每位读者提供竭诚的服务。

资源下载

读者可访问 http://www.packtpub.com 下载本书中的示例代码文件，或者访问 http://www.

packtpub.com/support，经注册后可直接通过邮件方式获取相关文件。

另外，我们还以 PDF 文件方式提供了本书中图片及图表的彩色图像，以帮助读者进一步理解输出结果中的变化，读者可访问 https://www.packtpub.com/sites/default/files/downloads/4813OT_ColoredImages.pdf 下载该 PDF 文件。

勘误表

尽管我们在最大程度上做到尽善尽美，但错误依然在所难免。如果读者发现谬误之处，无论是文字错误抑或是代码错误，还望不吝赐教。这对于其他读者以及本书的再版工作具有十分重要的意义。读者可访问 http://www.packtpub.com/submit-errata，选取对应书籍，单击 ErrataSubmissionForm 超链接，并输入相关问题的详细内容。经确认后，填写内容将被提交至网站，或添加至现有勘误表中（位于该书籍的 Errata 部分）。

另外，读者还可访问 http://www.packtpub.com/books/content/support 查看之前的勘误表。在搜索框中输入书名后，所需信息将显示于 Errata 项中。

版权须知

一直以来，互联网上的版权问题从未间断，Packt 出版社对此类问题异常重视。若读者在互联网上发现本书任意形式的副本，请告知网络地址或网站名称，我们将对此予以处理。

关于盗版问题，读者可发送邮件至 copyright@packtpub.com。

对于作者的爱护，我们表示衷心的感谢，并于日后向读者呈现更为精彩的作品。

问题解答

若读者对本书有任何疑问，均可发送邮件至 questions@packtpub.com，我们将竭诚为您服务。

目　录

第 1 章　动画基础知识

本章主要讨论 Unity 中的动画特征集，欢迎读者开启这一愉快的路程。动画的重要性不言而喻，如果缺少动画的支持，游戏中的某些元素将处于静止状态，因而缺乏生命力而看上去有些枯燥。例如，在游戏世界中，门通常可开启，游戏角色应可运动，树叶可随风摇摆，爆炸后的花火和粒子往往闪耀发光，如此等等。因此，作为开发人员，动画技能不可或缺。作为一类主题体验，游戏中的大多数领域均涵盖动画效果。因此，这对开发团队的全体成员均提出了新的要求，例如，设计师和动画师、程序员、音频设计人员以及关卡设计人员。本书对大多数开发人员仅具有较大的参考价值，并采用高效、简洁的方式引入相关基本概念，将实时游戏中的动画效果付诸于实践，特别是 Unity 动画。通过阅读本书，相信读者可具备扎实的基础知识和熟练的技能，并通过动画方式展示艺术视觉效果，以在后续工作中不断提升自己的素质。对此，本章首先讨论动画的基本概念，即理解动画的基础框架。

1.1　理解动画概念

在最基本的层面上，动画表示为两个特定且独立的特征之间的关系，即变化和时间。从技术上讲，动画定义了一段时间内的变化。也就是说，特征在时间范围内的调整和变化，例如，车辆在一段时间内的位置变化，交通灯在一段时间内的颜色变化（从红色到绿色）。因此，动画出现于全部时间长度内（持续时间）。在其生命周期内，对象特征将以特定的运行方式变化（帧），对应范围涉及动画的开始和结束点。

作为技术术语，上述定义稍显枯燥，但却具有重要的意义。然而，这一概念却并未谈及动画的审美和艺术特征。通过动画以及一段时间内的特征变化，可有效地表达情绪、氛围、场景以及理念等特征。动画中的感情和艺术感染力最终可解释为底层的时变关系。在这一时变框架的基础上，可进一步确定某些关键术语，对于计算机动画尤其如此。某些读者可能已对此有所了解，下面将采用正规方式对其加以定义。

1.1.1　帧

在动画表达中，时间需要划分为变化过程中独立且离散的单位，这一类单位称作帧。

就目前来讲，时间（例如秒）通常可划分为更小的单位（例如毫秒），但实际上，时间往往呈连续状态且不可打破。从理论上讲，该细分过程是可实现无限等分的，并产生越来越小的时间片段。相比较而言，时刻或事件则表示为人为的、离散且自包含实体。作为离散事物，人们可实时感受，并产生更为明确的场景体验。此类事物可发生于某一时刻或位于某一帧内。对于门的开启行为、角色的运动行为以及颜色的变化过程，动画帧提供了属性变化的机会——门的开启、游戏角色的移动行为、颜色的改变等。特别地，在视频游戏中，每秒须维持或包含特定数量的帧。根据硬件能力、所安装的软件以及其他因素，每秒内的帧数会随着计算机设备的不同而变化。相应地，每秒的帧数量称作 FPS（即每秒帧数），通常用作游戏性能的测算标准，较低的帧速率往往体现了较差的性能。图 1.1 显示了帧数除以时间后的结果。

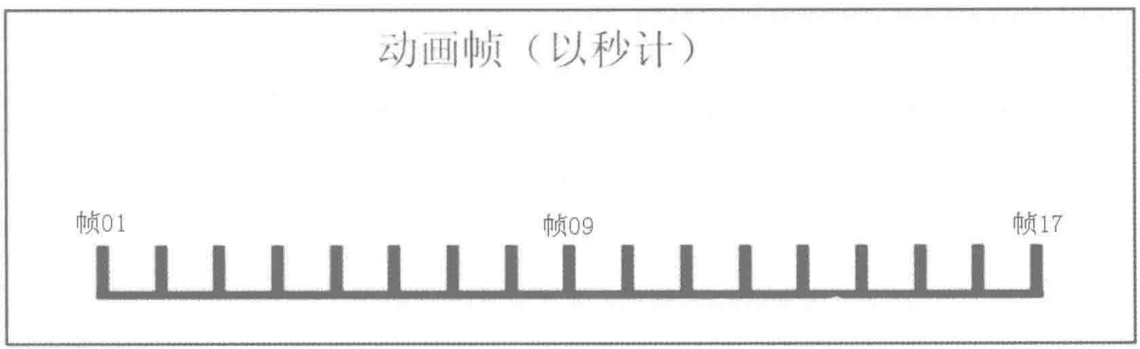

图 1.1　动画帧

1.1.2　关键帧

虽然帧提供了一种变化机会，但实际上并不会总是产生改变。也就是说，每秒内传递了多个帧，但并不是每帧都需要改变。而且，即使某帧需要进行适当调整，动画设计师定义其各个动作帧将是一个冗长且枯燥的过程。与手工或早期动画技术相比，计算机动画的优势是使操作过程变得越发简单。相应地，动画设计师可在动画序列中定义关键帧，也称作重要帧，随后计算机可自动生成中间帧。下面来看一类简单的动画，其中房门通过合页旋转 90°。开始时，门处于闭合状态，而最终会处于开启状态。此处，可对门定义两个关键状态，即开启状态和闭合状态，并定义了动画序列的起始点和结束点。由于定义了动画中的关键时刻，因而此类状态称作关键帧。在关键帧的基础上，Unity 自动生成中间帧（即补间动画，稍后将会看到），并在起始帧和结束帧平滑地旋转门，中间帧的数学处理过程称作插值计算。图 1.2 显示了关键帧之间的帧生成方式。

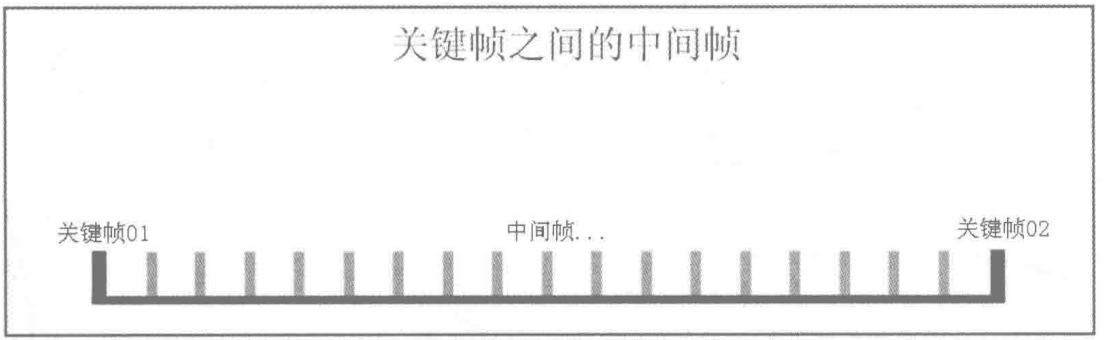

图 1.2　关键帧之间的中间帧

1.2　动 画 类 型

在 1.1 节定义了基础动画的核心概念。特别地，其中涉及了变化、时间、关键帧、中间帧和插值计算。在此基础上，可从技术视角区分视频游戏中的不同动画类型，这一点不同于艺术视角。不同的动画类型，其实现方式也有所不同。由于本质上的差别，故需要采用不同的处理和工作方式，包括特定的工作流以及所涉及的技术（后续章节将对此加以深入讨论），因而动画类型对于 Unity 而言具有重要的意义。1.2 节将介绍与此对应的动画类型。

1.2.1　刚体动画

刚体动画用于创建预置动画序列，如图 1.3 所示，可移动和改变对象的属性，并将此类对象视为整体实体，这与具有少量可运动部分的对象不同。此类动画类型包括奔驰在赛道上的车辆，合页固定下的开启的门，沿特定轨道飞行的飞行器，以及倒向墙体的钢琴。尽管上述示例间有所差异，但均包含了某些共性。虽然对象在关键帧之间有所变化，但依然作为独立的整体进行操作。换言之，尽管门可沿合页从闭合状态旋转至开启状态，但动画操作结束后其内部结构和组成未发生任何改变，该对象并未变形为老虎或狮子，未呈现爆炸效果或转换为果冻一类的胶状物，也未融合为雨滴。在动画过程中，门始终保持其物理结构，仅位置、旋转和缩放产生变化。因此，在刚体动画中，关键帧间的改变仅作用于整体对象及其高层属性中，并未渗透至其子属性或内部组成中，也就是说，对象的本质内部形式保持不变。这一类动画可在 Unity 的动画编辑器或 3D 动画软件（例如 Maya、Max 或 Blender）中直接加以定义，并于随后通过网格文件导入至 Unity 中。第 3 章将对刚体动画加以具体介绍。

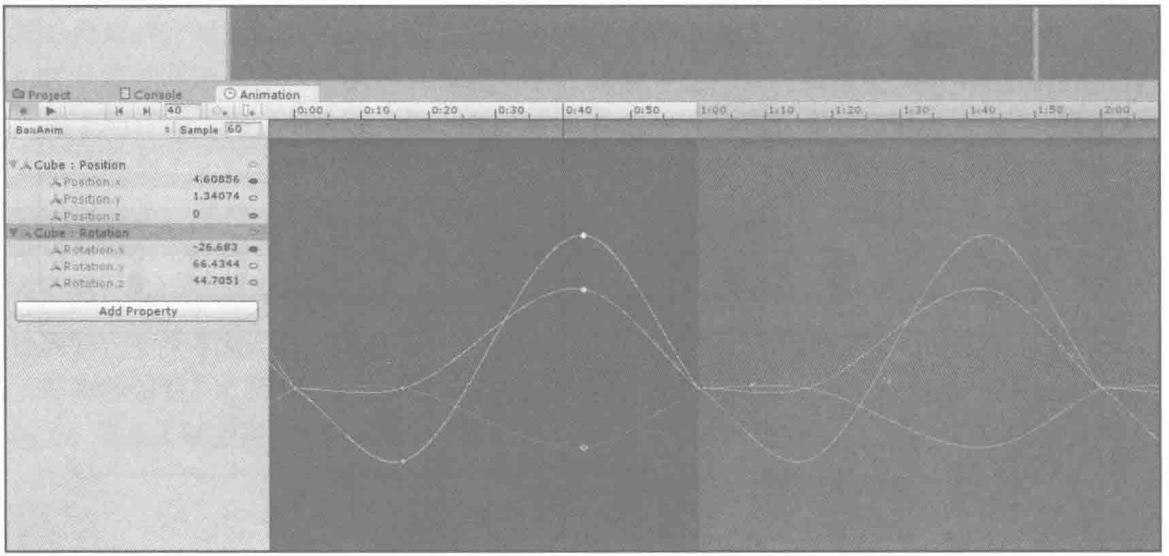

图 1.3　刚体关键帧动画

1.2.2　骨骼动画

如果需要呈现人物角色、动物、沾满黏液的食肉动物、爆炸或变形对象的动画效果，刚体动画难以满足这一要求。对此，需要使用到骨骼动画（也称作搭建动画）。这一类动画并不改变对象的位置、旋转和缩放内容，但会在帧间对其内部组成进行移动或变形。例如，动画设计师创建特定骨骼对象的网状物，并与底层网格骨骼予以近似，进而对周围和内部几何体实现方便、独立的操作方式。对于手臂、腿部、头部转动、嘴部运动以及舞动的树叶等，这一方式十分有效。通常情况下，各个动画在 3D 建模软件中作为完整的动画序列予以创建，并于随后通过网格文件导入至 Unity 中，如图 1.4 所示。同时，还可通过 Unity 的 Mecanim 动画系统加以处理和访问，第 5～7 章将对此予以讨论。

1.2.3　精灵动画

对于 2D 动画、图形用户界面以及各类 3D 特效（例如水面纹理），某些时候需要使用到包含动画纹理的、标准的四边形和平面网格。此处，对象不会像刚体动画那样产生运动，其内部结构也不会产生任何变化；相反，纹理自身处于动画状态。这一动画类型称作精灵动画。精灵动画利用图像或帧序列并以特定的帧速率播放，进而呈现一致的动画外观。例如，2D 横向卷轴游戏中，某一角色的行走动作循环过程，如图 1.5 所示。关于精灵动画的更多信息，读者可参考第 2 章。

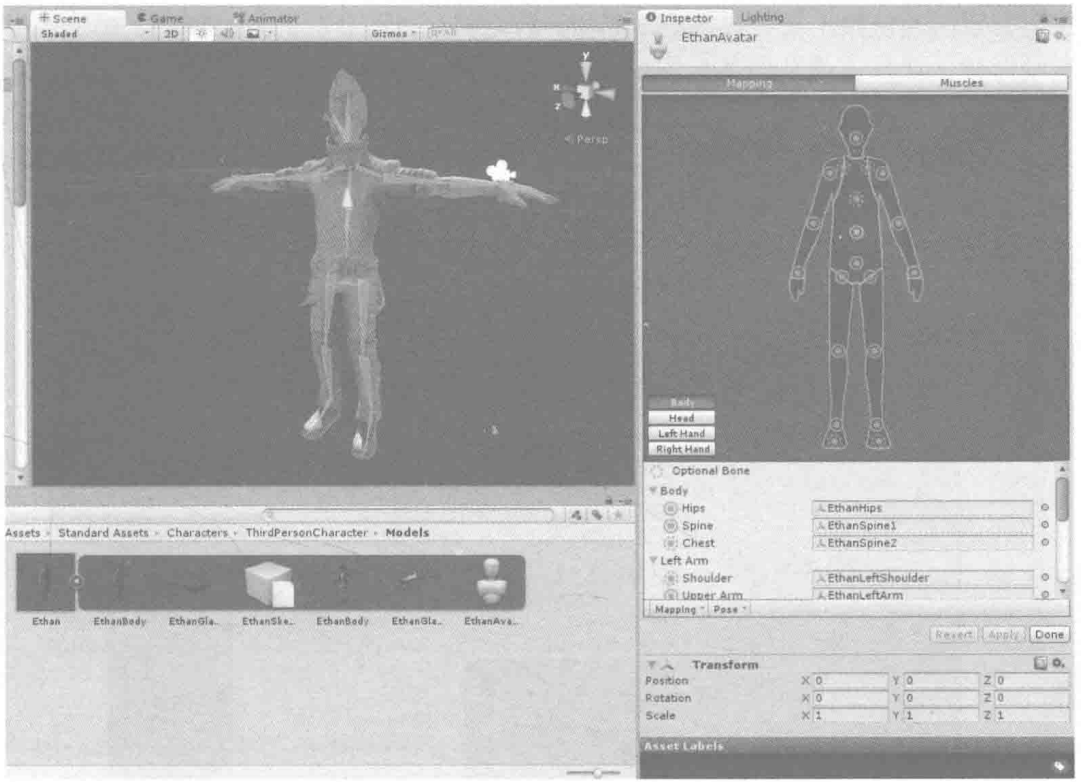

图 1.4　骨骼动画对于人物角色网格十分有效

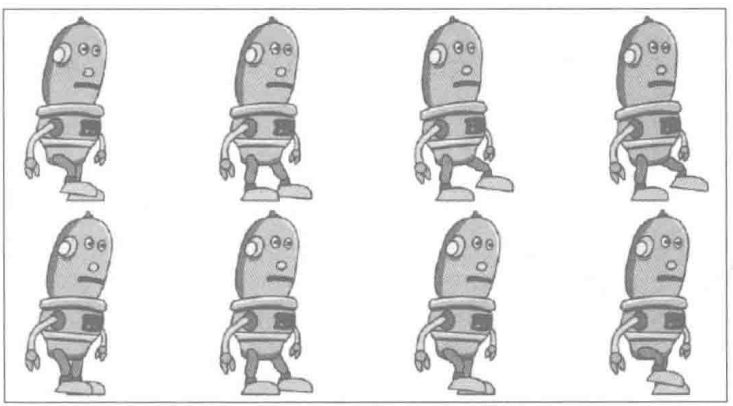

图 1.5　精灵动画

1.2.4　物理动画

多数时候，用户可对动画进行预定义。也就是说，可整体规划并创建对象动画序列，

并在运行期内通过预定义方式播放，例如，行走循环过程、门的开启序列、爆炸效果等。但某些时候，在玩家或其他各种场景因素无法预知的情况下，动画需要呈现真实效果并采用动态方式与场景反馈。相应地，存在多种方法可对此加以处理，其中一种方式是采用 Unity 的物理系统，如图 1.6 所示，包括绑定至对象上的组件和其他数据，进而实现真实的行为方式。例如，因重力作用下坠落至地面的物体，以及因风吹而褶皱的布料。

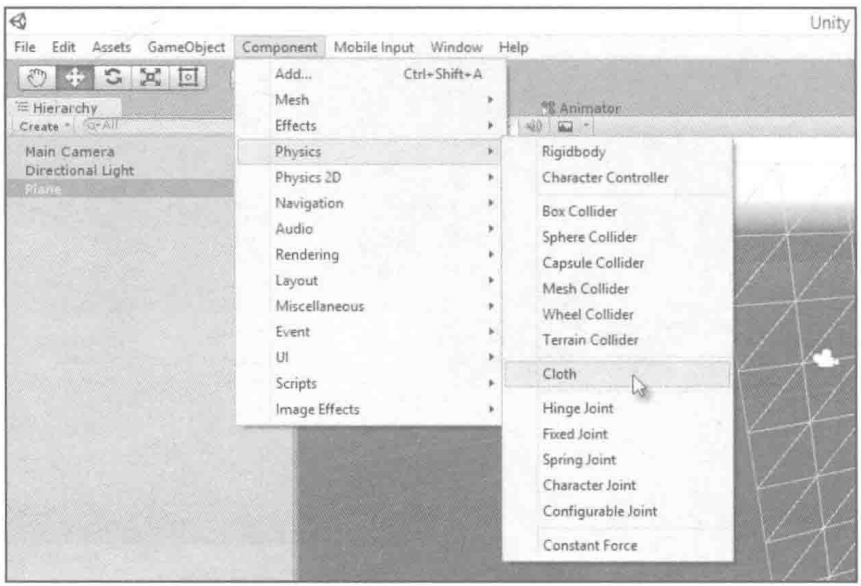

图 1.6　物理动画

提示：如果读者从正规渠道购买了 Packt Publishing 出版的书籍，则可在个人账户下下载示例代码文件。另外，读者还可登录 http://www.packtpub.com/support，注册后可通过邮件方式获得示例代码文件。

1.2.5　变形动画

有些情况下，前述动画方法（即刚体动画、精灵动画、骨骼动画或物理动画）无法满足需求，或许，用户需要呈现不同物体间的变形效果，例如，从人变成狼人，丑陋的蟾蜍变成美丽的公主，或者从巧克力棒变为一座城堡。类似这些情况，需要将网格状态混合或通过平滑方式从一帧合并至另一帧的不同状态中去，这一过程称作变形动画，或形状混合。实际上，该方法依赖于动画关键帧之间网格的对应顶点，以及中间帧不同状态间的混合结果。该方案的计算量较大，并对性能产生一定的影响，但可生成精美且逼真的外观。第 7 章将对形状混合加以讨论，图 1.7 显示了变形动画的起始状态。