基于斜视角可变的游戏引擎设计

陈 松

(重庆交通大学计算机学院 重庆 400030)

摘 要 游戏引擎是游戏的心脏,图形引擎模块是游戏引擎中最复杂,也是最能体现游戏效果的模块,对图形引擎模块的研究设计有利于提高二维游戏的画面质量。基于对二维游戏引擎中的图形处理模块研究,剖析了二维引擎实现引擎的算法,设计出了图形引擎函数库,用该函数库实现了一个地图编辑器。地图编辑器的实现使游戏引擎和具体游戏内容分离,提高了游戏软件的可重用性。

关键词 游戏引擎,图形模块,二维,游戏

Design of Graphic Engine Based on Changeable Angle

CHEN Song

(School of Computer ,Chongqing Jiaotong University, Chongqing 400074)

Abstract The engine is the kernel part of a game. And the graphics engine module is the most complicated one in it, also the one that can embody game's effect forcefully. The paper is about the primary research of graphics engine module in planar engine of the game. It first explores the planar game engine system and graphic processing technology. Using some pictures of pop game it shows the expression of the graphics engine module in game engine. It studies graphic processing key technology later. Finally the paper discusses the application of the graphic engine in planar engine of the game, and realizes a map editor designed by authors.

Keywords Game engine,Graphic module,Planar,Game

1 引言

随着社会进步,网络游戏逐步成为一种重要的休闲娱乐和教育手段。随着电子技术发展,游戏产业内容不断得到丰富。目前,游戏产业已经成为包含计算机软硬件技术、网络技术以及无线技术等最新科技和各种文化艺术的新型娱乐产业。

游戏引擎是游戏的心脏,它在后台控制着游戏中各个模块同时有序地工作,是一个由多个子系统构成的复杂系统,是游戏引擎的核心模块。它独立于特定的游戏,是抽象地处理游戏图片的模块。玩家所看到的画面,令人产生深刻印象的就是游戏引擎里面的图形引擎驱动的。以下对引擎设计做些探讨。

2 斜视角图形引擎的分析

一个最基本的游戏画面需要实现一些特效,如天气变化情况,地图的滚动,以及一定的动态视觉让游戏画面看起来更真实等,如图 1 是截取网易公司的大话西游里的一个场景分析,可以剖析其中的关键技术,当实现所有这些技术之后,一个斜视角的图形引擎就完成了。

(1)地表。许多 2D 游戏中都使用数组来描述游戏中的地图[1],在每个地图坐标处的地面情况称为地表,常见的地表有草地、沙漠、水、石板等等。由于每一块地表都拥有同样的大小,因此地表通常也称为 tile (瓦块,瓷砖)。一个完整的地图就是由各种各样的 tile 组成的。

(2)物体。游戏中物体最明显的特征就是它在地图上位置是不会发生改变的[2],诸如树、房屋、巨石等都属于物体的范畴。

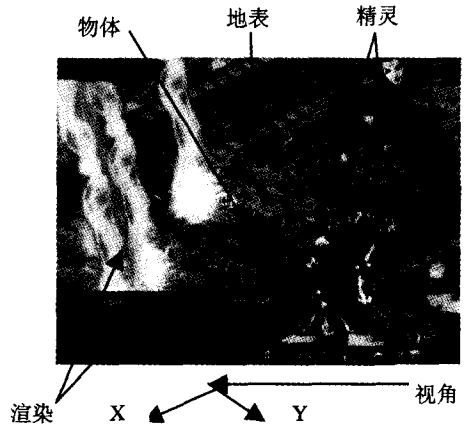

图 1 游戏画面技术总体

物体的位置不会发生改变并不就是说物体是完全静态的,在许多游戏中经常可以看到树和草等物体随风摆动等情景,不过这些只是表示物体的图片在不断改变而已,其实物体的位置并没有改变。

(3)精灵。一般的 RPG 游戏或即时战略游戏等都存在代表玩家的某一特定人物或事件等,称之为精灵。与物体最大的区别在于精灵会在地图上移动,精灵使用的是逻辑坐标而不是地图坐标。在实际游戏中精灵的控制要比物体复杂得多,这不仅因为要根据精灵不同的状态使用不同的图片,而且因为在游戏中通常为精灵融入了思维,也就是通常所说的人

陈 松 副教授,主要研究方向:图形图像、网络、数据库。

工智能。因此精灵拥有更多的属性，需要对所处的环境进行更多的判断。

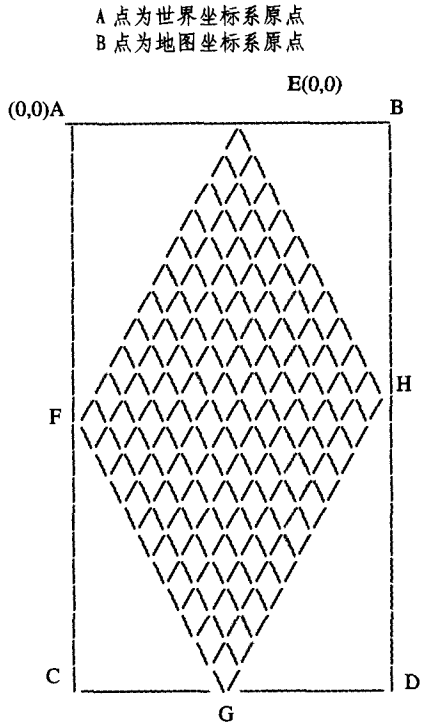

图 2　地图坐标系的表示方法

(4)渲染。一个游戏当中华丽的场面会给玩家留下很深刻的印象，而这些华丽的场面需要游戏引擎的渲染，要达到像现实生活中那样细腻、光亮，需要图形引擎的渲染，这些渲染也正是游戏的瓶颈所在。因为渲染往往要花费大量的 CPU 时间来计算，所以，要提高渲染的速度，关键在于算法。

在游戏中透明的特效用得非常的多，无论是用于游戏 UI 方面，还是用于游戏的特效，图片透明的处理都用得极多，它是由图片的 Alpha 渲染算法实现的。

(5)视角。在斜视角游戏中，每个 tile 并不是正方形的，而是有点扁的菱形。正是这种有点扁的菱形使地面出现了立体的感觉，就好像人们站在空中斜着向下看一样，斜视角游戏也因此而得名。该菱形到底有多扁是由游戏使用的视角大小来决定的。

(6)地图坐标系。在游戏中需要确定物体和精灵的位置，这样才能确定这些物体或精灵图片的输出位置，如图 2 所示[3]。

3　斜视角图形引擎的实现

游戏引擎的库函数主要分为三个模块，一个是显示模块(DisplayLib)，一个是地图驱动模块(MapEngine)，一个是脚本模块。引擎所有的功能都是基于这三个模块实现的，如图 3 所示。这三个模块之间相互独立但又是相互联系在一起的。从图可看出，显示库和脚本驱动是独立的模块，显示库是高度的抽象，所有游戏的显示功能都是与此库差不多的，但是地图驱动却是可以变化的，它相当于一个插件，改进或改变地图坐标不需要改动另外两个模块。

显示模块：该模块中有 4 个类，Surface 类，SurfaceManager 类，AlphaHelper 类，DDHelper 类 。类是功能类，后面 3 个都是行为类。该模块的类是图形引擎最底层的实现类，是直接跟 DirectX 和显卡打交道的类，这些类构成了图形引擎的实现。

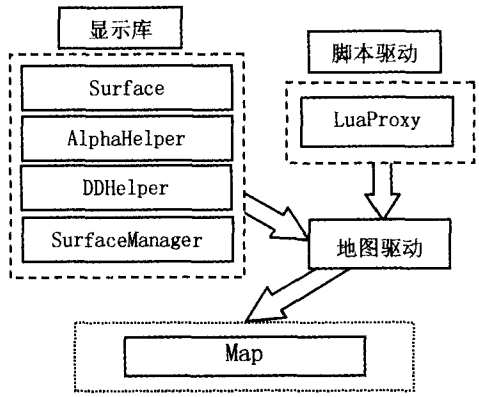

图 3　图形引擎类构

(1)Surface(页面)类。Surface 类是一属性类，是一个很重要的类，屏幕上所显示的都是这个类的对象。Surface，顾名思义，就是表面的意思，就是一切需要显示到屏幕上的所有的材质都是通过 Surface 类渲染的，就相当于贴图。游戏中的所有一切都是图片形成的，这个类就是组织这些图片，让它们各自有自己的属性，使某个行为知道应该把它们放到相应的位置。

(2)SurfaceManager 类。Surface 类是一属性类，而 SurfaceManager 类是一行为类，也是一个很重要的类，屏幕上所显示的都是经过这个类进行操作的。不错，它跟 Surface 类是相互相通的，它是 Surface 类的友元类，它主要是用图片进行操作。这个类被设计成单态类，这样对所有的页面都可以通过这个类的唯一实例进行操作了。这个类包括了所有对图片进行处理的操作，但它不是直接通过图片，而是都以 Surface 类来设计，这样就能很好地对图片原来既定的属性进行保护了，当用户调用这个引擎函数库的时候，他们操作的都是以页面为单位，达到面向对象的设计。此类是页面管理器，可以显示页面，就是把构造的页面显示到屏幕上，截取屏幕上一个区域作为一个页面，页面之间可进行一系列非常方便的互替换，拷贝，对页面进行 Alpha 处理，刷新页面，校正页面，把文字输入到页面等。

(3)DDHelper 类。DDHelper 类是一行为类，它是对 DirectX 进行封装，此类没有面向用户，用户不必知道怎么利用 DirectX，用户只知道页面也是页面管理器的使用，这个类是提供给 Surface 类和 SurfaceManager 类用的，对 DirectX 的操作都经过这个类。

(4)AlphaHelper 类。AlphaHelper 类是一行为类，它是对游戏特效算法的集合类。这是一个非常大型的类，由于特效算法涉及到游戏速度问题，游戏速度快不快大半部分都取决于此，就是因为这些特效所以游戏也更加绚丽多彩。由于追求速度，因此该类所有算法的实现都是经过汇编来实现的。

地图驱动模块：该模块跟显示模块一样是非常重要的核心模块。模块中有一个核心类：Map 类[4]，Map 类的功能非常强大，整个地图的驱动都在于这个类。除了这个核心类外，这个模块还有一些非常重要的结构[5]，这些结构构成了地图上信息的存贮，读取等。

脚本模块：人们在玩 RPG 游戏的时候，当走过某个地图块的时候就经常蹦出某几个怪物，它就像是地雷一样，只有当踩上去的时候它才会触发，而这就是脚本引擎的最基本的应用。可是脚本引擎的应用远不止这些，脚本，就像它的含义一

样，是这个游戏的剧本，玩家所走的，应该过哪几个关卡才能到达下一场景，这些都是通过脚本引擎来实现的。这里用的脚本引擎是一个开源的脚本引擎：LUA。Lua 是一种为支持有数据描述机制的一般过程式编程语言而设计的扩展编程语言。它同样可以对面向对象语言、函数式程序设计（Functional Programming，如 Lisp）以及数据驱动编程（data-driven programming）提供很好的支持。它的目标是被用作一种强大的、轻型的配置语言。Lua 目前已经被实现为一个扩展库，是用 clean C（ANSI C/C++ 的一个通用子集）编写的。作为一个扩展语言，Lua 没有"Main"函数的概念：它仅仅是嵌入一个宿主程序进行工作，可以称之为嵌入式编程或者简单地说是宿主编程。这个宿主程序可以调用函数以执行 Lua 的代码片断，可以设置和读取 Lua 的变量，可以注册 C 函数让 Lua 代码调用。Lua 的能力可以扩展到更大范围，在不同的领域内，在同样的语法框架下创建了自定义的编程语言。可以通过 LuaProxy 类来引用这个开源的脚本引擎达到控制游戏的作用[6]。

4 斜视角图形引擎的应用——地图编辑器

基于以上设计的图形引擎的地图编辑器的架构如图 4 所示。

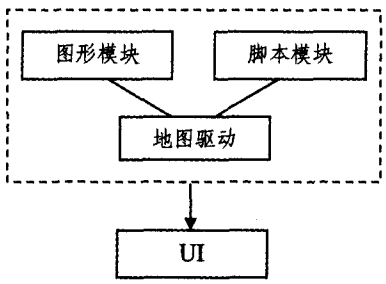

图 4 地图编辑器的架构

现代软件技术和分工的精细要求所有的软件设计应该面向模块化，讲求重用性，因此，引擎是跟具体的游戏分开的。而游戏是一个系统的工程，它由画面、音乐等组成，而对于具体的游戏来说，这些基本上是不会变的，但是随着现代个性化要求愈来愈强烈，大多数游戏都附带有地图编辑器，其目的就是给玩家自己定义自己想要的地图，而地图编辑器是最能体现出模块化设计的结果。

从图 3 可以看出，地图编辑器是直接挂在图形引擎上面的，也就是说，地图编辑器直接用图形引擎来实现，外面多加一层 UI 来操作，因此最能体现出图形引擎应用的就是地图编辑器。

结束语 在其后应用斜视角可变的游戏引擎实现的地图编辑器，证实了该引擎在 2D 游戏的应用中可以有效地提高动态视角的扩展，增加游戏玩家对游戏的动态控制的灵活性。不仅如此，在 3D 游戏中视角可变的应用也是非常常见的，但都没有取出作为一个独立的引擎来实现。在提取出这一功能模块实现引擎后，可以简单快速地应用于游戏，开发节约成本、缩短周期和降低风险等。如今越来越多的开发者倾向于使用第三方的现成引擎制作自己的游戏，一个庞大的引擎授权市场已经形成。因此，设计一个通用标准图形引擎比做一个具体的游戏更有可造性，文中的引擎机制已实际用于游戏开发中，使用效果良好。

参 考 文 献

1 Eberly D H. 3D Game Engine Design : A Practical Approach to Real-Time Computer Graphics[M]. 2005

2 Harrison L T. Introduction to 3d Game Engine Design Using Directx 9 and C#[M]. August 2003

3 荣钦科技. Visual C++游戏设计[M]. 北京：科海电子版社，2003

4 Knuth D. The Art of Computer Programming[M]. Addison-Wesley，2000

5 Franson D. 2D Artwork and 3D Modeling for Game Artists[M]. Muska & Lipman/Premier-Trade. November , 2002

6 Adams J. Role-playing Games With DirectX . Muska & Lipman/Premier-Trade[M]. January 2002

（上接第 232 页）

7 Cohen F S, Huang Z, Yang Z. Invariant matching and identification of curves using B-splines curve representation. IEEE Transactions on Image Processing，1995, 4：1～10

8 Liu H，Srinath M. Partial shape classification using contour matching in distance transformation. IEEE Transactions on PAMI, 1990，12：1072～1079

9 Young I, Walker J，Bowie J. An analysis technique for biological shape. Computer Graphics and Image Processing, 1974，25：357～370

10 Hu M K. Visual pattern recognition by moment invariants. IRE Transactions on Information Theory，1962, 8：179～187

11 Teague M R. Image analysis via the general theory of moments. Journal of the Optical Society of America，1980, 70：920～930

12 Khotanzad A, Hong Yaw Hha. Invariant Image Recognition by Zernike Moments. IEEE Trans. on PAMI，1990, 12(5)

13 Kan Chao, Srinath M D. Invariant character recognition with Zernike and orthogonal Fourier-Mellin moments. Pattern Recognition, 2002，35：143～154

14 Loncaeic S. A survey of shape analysis techniques. Pattern Recognition, 1998，31(8)：983～1001

15 Blum H. A transformation for extracting new descriptors of shape. In：Whaten-Dunn. editor. Models for the Perception of Speech and Visual Forms, 1967. 362～380

16 Leymarie F, Levine M D. Simulating the grassre transform using an active contour model. IEEE Transactions on PAMI, 1992，14：56～75

17 Han C C, Fan K C. Skeleton generation of engineering drawings via contour matching. Pattern Recognition，1994, 27：261～276

18 Haralick R, Sternberg S, Zhuang X. Image analysis using mathematical morphology. IEEE Transactions on PAMI，1987, 9：532～550

19 Serra J，editor. Image Analysis and Mathematical Morphology. Vol2：Theoretical Advances. Academic Press，1988

20 Shapiro L，Haralick R，Decomposition of two-dimensional shapes by graph-theoretic clustering. IEEE Transactions on PAMI, 1979, 1：10～20

21 Zadeh L. Outline of a new approach to the analysis of complex systems and decision processes. IEEE Transactions on SMC, 1973，3：28～44

22 di Baja G S, Thiel E. (3，4)-weighted skeleton decomposition for pattern representation and description. Pattern Recognition, 1994，27：1039～1049

23 Biederman I. Recognition-by-components：a theory of human image understanding, Psychological Review, 94：115～147

24 Sebastian T B, Klein P N, Kimia B B. Recognition of shapes by editing shock graphs. In：Proceedings of the Eighth International Conference on Computer Vision, Vancouver, Canada, July 9-12 2001. IEEE Computer Society Press，2001. 755～762