DOI:10.16182/j.cnki.joss.2006.s1.128

2.5D 游戏中地图的即时生成算法

渠 鑫，周 军，付百文，宁静静
（北京联合大学信息学院，北京 100101）

摘 要：地图的显示在游戏引擎中占有很重要的地位，常用的是一次生成算法。提出了一种效率较高的**即时生成算法**。该算法通过游戏主屏相对于逻辑地图的位移量，对游戏中景物进行**动态裁剪**，达到即时生成的目的。

关键词：即时生成；裁剪；游戏
中图分类号：TP391.9 **文献标识码**：A **文章编号**：1004-731X (2006) S1-0414-03

In-Time Map Generation Method for 2.5D Game

QU Xin, ZHOU Jun, FU Bai-wen, NING Jing-jing
(Information College, Beijing Union University, Beijing 100101, China)

Abstract: The map display in game Jet is important. Once-generation method is general. An *in-time generation method* was given. *Dynamatic clip* of game scene was generated by using relative displacement of main screen to logical map.
Key words: in-time generatin; clip; game

引 言

一般来说游戏中的地图生成有两种算法，一种是事先做好的，也就是先在一张大的画板上将游戏在这个场景中所用到的各种景物事先画好，当游戏的显示屏幕根据键盘或者鼠标进行移动时，将所用到的背景图从做好的大地图中裁剪出来，然后拷贝到显示屏幕上形成屏幕背景的卷轴动画。这样做就必须对每一游戏帧都重新画整个背景大图，即使有些景物在现在的主屏上并不显示出来，这样就会使游戏的运行效率很低，而且必须建一个非常大的画板，由于有这个限制在游戏中一个场景就不可能做得非常大。

另一种方法就是地图画面的即时生成，简单来说就是在游戏的主屏上需要哪一部分就显示那一部分。这样由于只需要在每一个游戏帧当中重画主屏大小的画面，这样游戏的运行效率就会大大提高。而且没有了游戏画面的限制，就可以把一个游戏场景地图做得非常大。但是由于它不是一次性生成的，所以在显示方面比较慢，可现在的计算机硬件技术已经非常的先进，能够弥补这种缺陷。

1 即时生成算法的思想

现在的游戏基本上都是运行在 Windows 操作系统下的。所以一般的游戏开发都是运用 DirectX 技术进行编写的。而我们这里所编写的 2D 斜视角游戏就是运用了 DirectX 中的 DirectDraw 技术进行编写的。

HRESULT BltFast(DWORD xDest, DWORD yDest, LPDIRECTDRAWSURFACE7 lpSurfSrc, RECT* pRtSrc, DWORD dwTrans);

这是一个 IDirectDrawSurface7 接口中的一个函数，这个函数的作用就是将原表面所指定的矩形块传送到目标表面的具体位置。在这个函数中：

xDest: 目标表面的 x 坐标；
yDest: 目标表面的 y 坐标；
lpSurfSrc: 源表面指针；
pRtSrc: 原表面矩形块结构；
dwTrans: 块传递的参数一般用于设置关键色；

这是一个在游戏制作中比较常用的函数，将游戏中用到的图片加载到离屏页，然后根据游戏中场景的需要就用这个函数将所用到的图片拷贝到缓冲区然后进行翻页，就可以将做好的游戏场景显示到屏幕上。

在即时生成算法中，首先应该有一张事先做好的完整地图的位图,以及在地图中应该显示的各种景物。然后将这些图片先在程序中加载进来。由于这是即时生成算法，所以不需要真正建一张与地图大小完全一样的绘制面，但为了方便以后计算，在这里建立一张大小与地图一样的逻辑中的地图。

逻辑地图在游戏中只是我们在头脑中预先想象的一张完整的地图，实际上面并没有任何的景物。但是我们要画的景物在逻辑地图中的位置是事先确定下来的。当用键盘控制游戏中的人物走动时，屏幕就好象一个望远镜似的看到逻辑地图的哪个部分，那么那部分景物就在逻辑地图当中画出来显现在镜片当中。如何协调好逻辑地图、主屏幕、和装满许多景物的位图三者之间的关系来完成地图的即时生成就变得比较复杂，这也是我们要解决的问题。

收稿日期：2006-03-01 **修回日期**：2006-05-31
基金项目：北京市优秀人才培养专项经费资助项目（20042D0502202）
作者简介：**渠鑫**(1983-)，男，北京人，大学本科，研究方向为计算机游戏、软件设计；**周军**(1983-)，男，北京人，研究方向为计算机游戏、软件设计；**付百文**(1968-)，男，山东临清人，副教授，硕士，研究方向为计算机图形学、算法设计。

根据扇面场景的描述，我们应该知道景物在位图中的顶点坐标即：(Src_X,Src_Y),也能知道景物的宽和高即：Src_W 和 Src_H。同样我们看到景物的哪个部分，才将那个部分显现出来，这样我们就必须先知道截得的图相对逻辑地图的坐标（CutPic_X,CutPic_Y）,最后拿着望远镜要在逻辑地图中移动着看，也就是主屏幕相对逻辑地图要运动，所以我们必须知道屏幕相对逻辑地图的坐标（MsPic_X,MsPic_Y）。到此为止能够给我们直接提供的量就这么多了，当然这是不够的，剩下的就需要我们在算法中实现了。抽象出来，我们要完成即时生成地图的工作还需要六个量，也就是我们待求的。它们是截图相对位图的坐标（CutBm_X,CutBm_Y）、要截的图的宽和高即：Cut_W 和 Cut_H，因为最后截得的图要反映在屏幕上，我们还得求出截图相对屏幕的坐标（CutMs_X,CutMs_Y）。

2　算法的描述

为了清楚起见，我们将这些变量结合图形表示出来，如图 1 和图 2。

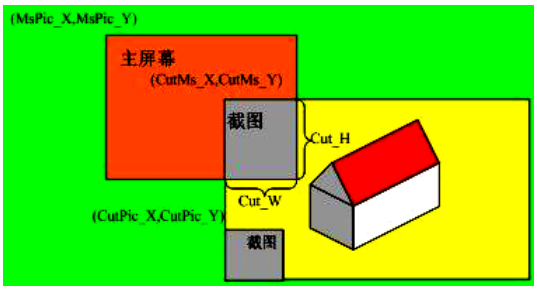

图 1　逻辑地图、主屏幕和位图的关系

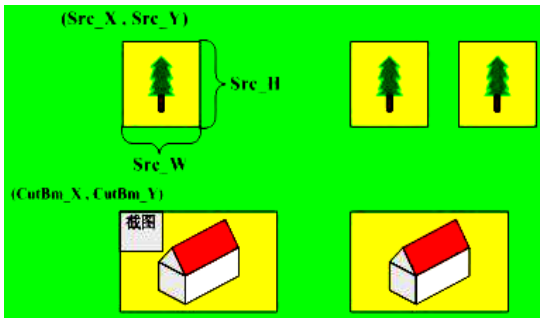

图 2　截图和位图的关系

由于主屏的移动是由玩家所控制的，所以带有不确定的因素，所以必须将所有可能的情况都考虑齐全。在这个算法中将可能出现的情况为九种。

根据主屏的位置不同划归为九种情况，再根据每种情况的不同将主屏上所需要的房子图片的截图正确地显示出来，如图 3 所示。

根据 SrcPic_X 与 MsPic_Y 把整个算法分成 A、B 两个部分：当 SrcPic_X 大于 MsPic_Y 时，执行情况 A，否则执行情况 B。算法的 N-S 流程图如图 4、5 所示。

我们对于这九种情况进行一下分析，对于 CutBm_X、

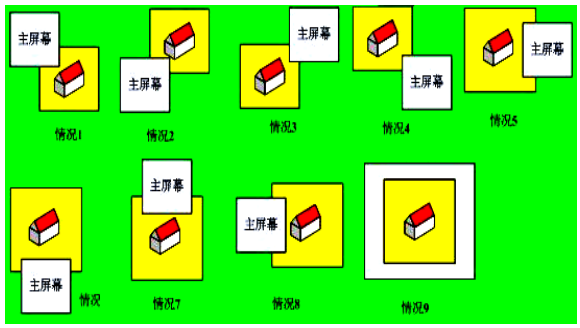

图 3　算法情况讨论

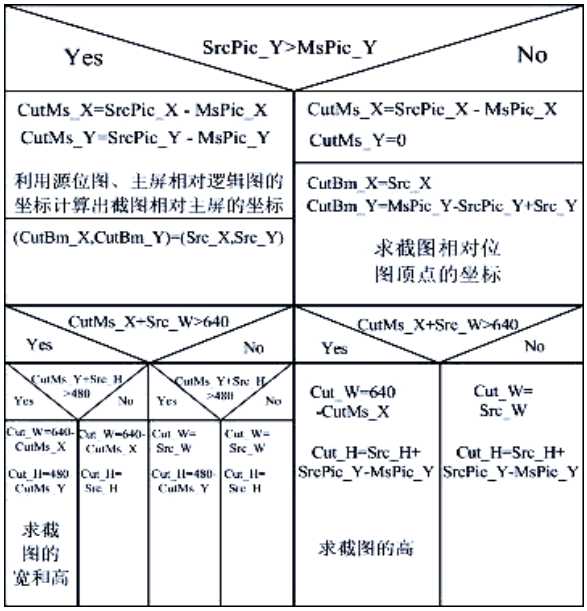

图 4　N-S 即时生成算法的图(情况 A)

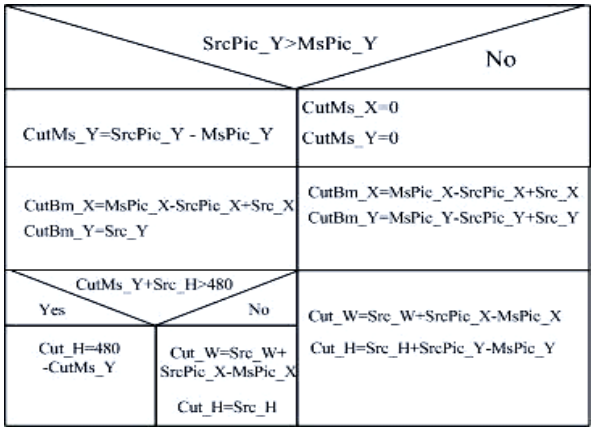

图 5　N-S 即时生成算法的图(情况 B)

CutBm_Y、CutMs_X、CutMs_Y 这四个所要求的值可以分成四类：

第一类：包括 1、5、6、9 情况；

第二类：包括 2、7 情况；

第三类：包括 3、8 情况；

第四类：包括 4 情况；

第一类：

在这一类中包含了 1、5、6、9 这四种情况。其特点是

房子截图相对于逻辑地图坐标的 x、y 值都比主屏相对于逻辑地图坐标的 x、y 值大。利用源位图相对逻辑图的顶点坐标和主屏相对逻辑图的坐标计算出截图相对主屏的坐标。

到这里所要求的六个值当中我们已经求出了四个，下面来求房子截图的宽和高。

由于第一类包含了 1、5、6、9 这四种情况，在这四种情况中每种情况的房子截图宽和高的值都是不一样的，所以我们还要分情况进行讨论。在这里我们假设主屏的分辨率是 640*480。根据截图相对主屏幕的坐标、源位图的宽和高与主屏幕的宽（640）和高（480）进行比较区分各种情况就能计算出截图的宽和高。

到这里为止在第一类情况下所需要的六个值就都已经求出来了。

第二类：

在这一类中包含了 2、7 两种情况。这种的特点是房子截图相对于逻辑地图坐标的 x 值比主屏相对于逻辑地图坐标的 x 值大，但是 y 值小。而且房子截图相对于主屏的 y 坐标为 0。根据这些特性可求出房子截图相对位图的坐标和房子截图相对于主屏的坐标。

下面在求截图的宽和高时也要根据截图相对主屏幕的坐标、源位图的宽和高与主屏幕的宽（640）和高（480）进行比较区分各种情况来计算。这和上面讨论的方法一样，只不过其中只包含了两种情况，所以讨论起来就比较简单。

所需要的六个值就已经讨论完毕。

第三类：

在这一类中包含了 3、8 两种情况。而这一类与第二类是截然相反的。它的特点是房子截图相对于地图逻辑坐标的 x 值比主屏相对于地图逻辑坐标的 x 值小，而 y 值大。这样房子截图相对于主屏的坐标也截然相反是 x 坐标为 0。根据以上特点就可以求出房子截图相对位图的坐标和房子截图相对于主屏的坐标。

还是利用与一、二类情况相类似的讨论方法来求截图的宽和高。

第四类：

这一类比较特殊只包含了一种情况，即情况 4。它的特点是与第一类截然相反。房子截图相对于逻辑地图坐标的 x、y 值比主屏相对于逻辑地图坐标的 x、y 值都小。而且房子截图相对于主屏的坐标 x、y 值都为零，所以相比来说这种情况下的六个值比较好求。

到此为止所有情况下的六个所需要的值就都已经求出来，只要将这些值合理的安排起来，当主屏发生移动时将主屏相对于地图的值不断地进行计算就能够即时地算出房子等景物要截得的图的顶点坐标，以及截图的矩形区域，还有要在主屏上显示的坐标。有了这六个参数就能够准确的将房子的截图显示出来。达到了即时生成的目的，再与其它程序配合，就能够完整作出游戏背景卷轴动画。

3 算法分析

在这种算法中只要事先将景物等在地图中的实际坐标定下来，就可以直接计算出所需要的各种参数。比之用地图数组解决的算法，由于本算法没有循环等复杂控制结构，所以时间复杂度为一个常数，大大提高了效率。

经过实验，在地图比较大时，用本算法没有出现画面显示缓慢的问题，而用一次生成算法时会出现明显的缓慢情况。

4 结论

游戏引擎中画面显示的是否流畅是大家比较关注的问题。本算法利用动态裁剪实现地图的即时生成，解决了一次生成算法和用地图数组方法解决地图显示中的效率问题以及场景限制问题，大大提高了游戏的美观以及效率。

参考文献：

[1] 刘勇奎, 高云, 黄有群. 一个有效的多边形裁剪算法 [J]. 软件学报, 2003, 14(4): 845-856.
[2] 荣钦科技. Visual C++ 游戏设计 [M]. 北京: 北京科海电子出版社, 2003.
[3] 坂本千寻. Visual C++ 角色扮演游戏程序设计 [M]. 北京: 清华大学出版社, 2004.
[4] Julian Gold. 面向对象的游戏开发 [M]. 北京: 电子工业出版社, 2005.
[5] 四维科技, 丁展. Visual C++ 游戏开发技术与实例 [M]. 北京: 机械工业出版社, 2005.
[6] 李雪, 石广田. 一种新的任意四边形窗口线裁剪算法 [J]. 兰州交通大学学报, 2005, 24(6): 90-92.