

doi: 10.3969/j.issn.1006-1576.2009.11.025

三维可视化地形图的设计与实现

李大雨, 刘新文, 汪新兵, 谢方明
(防空兵指挥学院 信息控制系, 河南 郑州 450052)

摘要: 对三维地形可视化地形图开发中的几个关键技术进行了分析和研究。利用 VC++ 和 OpenGL 作为开发工具, 通过建立位图的灰度值与地形高程之间的映射关系, 快速生成三维地形, 然后再经过纹理贴图与种植随机树, 产生具有较强真实感的三维可视化地形图。

关键词: OpenGL; 可视化; 纹理贴图; 随机树

中图分类号: TP391.41 **文献标识码:** A

Design and Realization of 3D Visualization Terrain Map

LI Da-yu, LIU Xin-wen, WANG Xin-bing, XIE Fang-ming
(Dept. of Information Control, Air Defense Forces Command Academy, Zhengzhou 450052, China)

Abstract: Analysis and research several key technologies in development of 3d visualization terrain map. Adopt VC++ and OpenGL as the development tool, establish the mapping relation between gray-number of bitmap and terrain altitude, generation three-dimensional terrain quickly. Then, through texture-pasting and planting random trees, creates three-dimensional visualization terrain map with strong truthfulness.

Keywords: OpenGL; Visualization; Texture-pasting; Random trees

0 引言

地形是一种非常复杂的自然景物, 如何在 PC 机上设计与实现具有动态特性的三维真实感地形图是计算机图形领域的重要课题, 同时也具有重要的经济价值和军事应用价值。三维电子地形图历经了划线地形图、实体地形图和三维真实感地形图 3 个发展阶段, 划线地形图和实体地形图虽然具有一定的立体效果, 但信息量不足, 而三维真实感地形图能够较逼真地反映外部真实世界。故讨论一种在 VC++ 环境下使用 OpenGL 绘制三维地形图的方法, 并给出实现的主要过程。

1 OpenGL 与 VC++ 的接口设置

OpenGL 是美国 SGI 公司开发的可独立于操作系统和硬件环境的三维图形库, 强大的图形操作功能和跨平台的能力使其在许多领域都有应用, 并得到硬件 (显卡) 厂商的广泛支持。同时, OpenGL 是一种过程性而非描述性的图形 API, 不包括任何程序流控制、窗口操作、人机交互之类的命令或函数, 因此需要利用 VC++ 作为开发平台。OpenGL 与 VC++ 的图形接口的实现步骤如下:

1) 首先, 利用 VC++ 的 MFC App Wizard 生成应用程序的工程文件, 即创建应用程序的基本框架, 接着需要设置连接 OpenGL 库文件, 在菜单中选择

Project ->Settings, 最后选择 LINK 选项, 然后在 “Object/Library Modules” 下增加 OpenGL 所需的函数库文件, 具体包括 OpenGL32.lib、Glu32.lib 和 Glaux.lib。

2) 在 StdAfx.h 中添加 OpenGL 函数定义的头文件, 在工程的任何位置调用 OpenGL 的各种函数。

添加代码如下:

```
#include <gl/gl.h>
#include <gl/glu.h>
#include <gl/glaux.h>
#include <gl/glut.h>
```

2 三维可视化地形图的绘制

2.1 生成高程灰度图

在地理科学中, 一般使用等高线的方法来描述山地。如果按一定的横切面将山体切成若干份, 然后把山体在每个横切面上的投影全部集中在一个平面上, 就形成了等高线地图。

在计算机图形处理技术上, 等高线地图提供了还原山势地貌的可行性。可以采用位图的格式来表示等高线地图, 即用黑白色的深浅 (灰度) 来表示山势的高低 (通常定义白色为最高, 黑色为最低), 这样就生成了用来表示高度的高程灰度图。

2.2 获取地形高度

由高程灰度图中读出对应点高度是由地域数组

收稿日期: 2009-05-26; 修回日期: 2009-08-06

作者简介: 李大雨 (1978-), 男, 安徽人, 硕士, 防空兵指挥学院讲师, 从事遥控靶机研究。

生成函数来完成的, 可以分别生成位图中各个位置的 X 分量、Y 分量和 Z 分量。其中, Y 分量的大小代表了各点的高度, 即地面高低起伏的状态。

地域数组用 `g_terrain[MAP_W*MAP_W][3]` 来表示, 以下是地域数组生成函数的主要代码:

```
for (int x = 0; x < MAP_W; x++)// MAP_W 为地块数
{
    Vertex = z * MAP_W + x;
    g_terrain [Vertex][0] = float(x)*MAP_SCALE;
    //地域数组 X 分量
    g_terrain [Vertex][1] = (float)
    (g_imageData[(z*MAP_W+x)*3]/N);
    // 地域数组 Y 分量, 代表了该点的高度。
    g_terrain [Vertex][2] = -float(z)*MAP_SCALE;
    //地域数组 Z 分量
}
```

其中, 数组 `g_imageData[]` 是位图 (高程灰度图) 装载函数的返回值, 代表了像素的颜色深度。通过改变地域数组 Y 分量中整数 N 的大小, 可以生成高低起伏不同效果的地形。

2.3 地面纹理贴图

纹理贴图又叫纹理映射, 是计算机图形学中广泛使用的一项重要技术, 能描述表面的细节, 提高图形的真实性。纹理贴图的实现由函数 `RenderScene()` 实现, 主要完成地面景物的设置。`RenderScene()` 函数的主要代码如下:

```
RenderScene()
{
    glBindTexture(GL_TEXTURE_2D, g_cactus[0]);
    //地面贴图选择
    glTexEnvf (GL_TEXTURE_ENV,
    GL_TEXTURE_ENV_MODE, GL_REPLACE);
    glTexParameterf(GL_TEXTURE_2D,
    GL_TEXTURE_MAG_FILTER, GL_LINEAR);
    glTexParameterf(GL_TEXTURE_2D,
    GL_TEXTURE_MIN_FILTER,
    GL_LINEAR_MIPMAP_NEAREST);
    for (int z = 0; z < MAP_W-1; z++)
    glDrawElements(GL_TRIANGLE_STRIP, MAP_W*2, GL_UNSIGNED_INT, &g_index[z*MAP_W*2]);
}
```

通过选择相应的贴图纹理, 可以生成诸如草原、沙漠、戈壁等不同的地面效果。

2.4 生成随机树

在三维场景中的树应该采用 3D 对象 (3DS 模型) 要好一些, 但有人却设计了一种用平面图形做的树^[4], 显示时不但具有很好的 3D 效果, 而且显示速度极快。与采用 3D 对象相比, 在一个场景中

栽上几千棵树对显示速度几乎没有影响。栽种平面树步骤如下:

1) 调入 16 位色的 BMP 位图作为树的贴图, 实现代码如下:

```
LoadBMP("image/TREE.BMP", g_cactus[10]);
//选择树贴图
```

```
2) 设置随机树的数量、位置、种类及大小
srand(100); //产生树的固定随机数种子
for(int i=0; i<30; i++) //树的数量
{float x= RAND_COORD((MAP_W-1)*MAP_SCALE);
//树的位置 x
float z= RAND_COORD((MAP_W-1)*MAP_SCALE);
//树的位置 z
float size=4.0f+rand()%4; //大小 2~4 随机
float h=-size/10; //深浅
int cactus=10; //树形
}
```

3) 种植平面树

为了在场景变化时, 始终看到的都是平面树的正面, 即始终面向它, 需要对平面树进行特殊处理, 主要代码如下:

```
PlantTrees(float x, float z, float h, float s, int cactus)
//平面树
{
    glEnable(GL_BLEND); //启用混色
    glBlendFunc(GL_SRC_ALPHA,
    GL_ONE_MINUS_SRC_ALPHA); //指定混合属性
    glEnable(GL_ALPHA_TEST); //启用透明
    glAlphaFunc(GL_GREATER, 0);
    //设置透明测试
    float mat[16]; //定义数组保存场景矩阵
    glGetFloatv(GL_MODELVIEW_MATRIX, mat);
    //取得场景矩阵的数据
    vector3_t X(mat[0], mat[4], mat[8]);
    //场景矩阵的 X 方向矢量
    vector3_t Z(mat[1], mat[5], mat[9]);
    //场景矩阵的 Z 方向矢量
    glBindTexture(GL_TEXTURE_2D,
    g_cactus[cactus]); //选择树
    vector3_t pos(x, 0.0, -z);
    //树的位置 X、Z 矩阵变换
    pos.y = GetHeight(x, -z) + h + s;
    //树的位置 Y 分量
    .....
}
```

这样一来, 当场景旋转时, 场景中的树也跟着旋转, 本来由平面构成的树就看不出是平面的了。

按上述步骤, 经选择不同的高低起伏效果、设置不同的纹理贴图, 种植不同的树木, 可生成诸如高山、丘陵、沙漠等不同的三维地形模型, 如图 1。

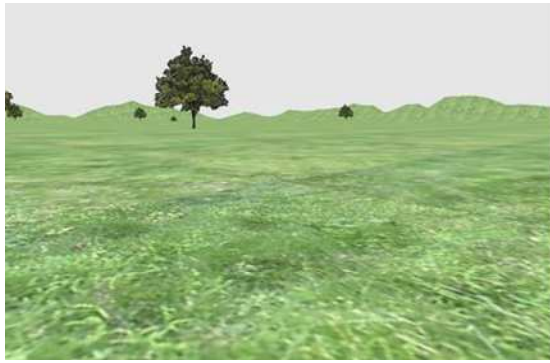


图 1 三维可视化地形图（丘陵、沙漠）

3 结束语

利用 OpenGL 提供的图形函数构造三维地形模型，并通过纹理贴图、种植树木，最终生成的三维可视化地形图具有一定的真实感，但仍缺乏高度仿真、交互控制的功能和特点，将在工作中努力解决。

参考文献：

- [1] Richard J. Wright. OpenGL 超级宝典[M]. 北京：人民邮电出版社, 2005: 9.
- [2] 和平鸽工作室. OpenGL 高级编程与可视化系统开发[M]. 北京：中国水利水电出版社, 2006.
- [3] 向世明. OpenGL 编程与实例[M]. 北京：电子工业出版社, 1999: 9.
- [4] 张筠莉. Visual C++实践与提高—串口通信与工程应用篇[M]. 北京：中国铁道出版社, 2006.
