基于 OpenGL 的 DEM 地形可视化和虚拟漫游系统①

Development of DEM Terrain Visualization and Virtual Wandering System based on OpenGL

赵庆展 （石河子大学信息科学与技术学院 832003）
张 清 （北京大学地球与空间科学学院遥感所 100871）
宁 川 （石河子大学信息科学与技术学院 832003）

摘要：本文介绍了如何使用 OpenGL 进行 DEM 地形可视化，说明了地形虚拟漫游系统的建模原理和实现过程，给出了纹理映射、第一人称漫游和飞行漫游的一般方法，讨论了在 Windows 环境下使用 OpenGL 进行三维编程的基本步骤，并给出了一个虚拟场景生成的实例。

关键词：OpenGL DEM 可视化 虚拟漫游

1 引言

虚拟现实（Virtual Reality，简称 VR）是涉及到计算机图形学、人机交互、数字图像处理、人工智能等多个学科的综合技术，其目的是用计算机来生成逼真三维空间世界，给用户如同真实感的体验。

三维地形模型是三维地理信息系统（3D－GIS）最基本的地形模型，地学分析图形中的三维地形立体图，通常是通过一组投影变换的剖面线或网格线构造的，真实感和立体感较差，纹理映射由于受到具体条件的限制很难通过真实的数字图像拍摄完成。数字高程模型（Digital Elevation Model）从数据上讲是表示区域 D 上的三维向量有限序列，是一种表示三维空间连续起伏状态的数学模型[2]，在测绘、地质、地理、军事等领域用途广泛，利用 DEM 数据生成可视化地形，可以更好的描述特定区域的地形特征，配合区域内自然场景的建模技术，可生成较为逼真的真实场景。

2 数字高程模型介绍

2.1 DEM 数据

数字高程模型 DEM 是由美国 MIT 的 Miller 教授于 1956 年提出来的，经过几十年的发展，现在一般认为 DEM 的目的使用摄影测量或其他技术手段获得的地形，再满足一定精度的条件下，用离散的形式在计算机中进行表示，并用数字计算的方式进行各种分析。DEM 是对地形地貌的数字描述和模拟，包括平面和高程两种数据，是表示区域 D 上三维向量的有限序列，其函数可表示为：

$V_i = (X_i, Y_i, Z_i)$

其中：$i = 1,2,\cdots,n$，X_i, Y_i 是平面地理坐标，Z_i 则表示 (X_i, Y_i) 处所对应的高程。

DEM 用数字形式 X，Y，Z 坐标表示区域内的地貌形态，根据它可以自动生成等高线图，确定地面的通视情况图、坡度图、地形断面和透视等。

2.2 DEM 数据转换

DEM 数据大多是在 Arc/Info、Arc View 等专业 GIS 软件中生成由离散的数据点或等高线直接生成，所以从建模角度出发，在 ArcInfo 中，可通过等高线内插建立不规则三角网（TIN），或将 TIN 进一步转化为规格网格（GRID）。最后这些 DEM 数据转化成为 ASCII 文件，由于 OpenGL 不能直接对其进行读取，所以需要把 DEM 用 OpenGL 原语描述出来，通过程序对数据进行读取，转化成 OpenGL 可以识别的图形函数。

① 基金项目：石河子大学新办专业资助项目（ZX200332）石河子大学骨干教师资助基金（XX01011）

3　OpenGL 的 DEM 数据读入方法

3.1　OpenGL 简介

OpenGL 是一个开放的图形硬件的软件接口，是 SGI 公司于 1992 年发布的三维图形编程接口，也是事实上的工业标准，目前已经成为国际上通用的开放式三维图形标准。OpenGL 集成了图形变换、光照、材质、纹理、像素操作、融合、反走样、雾等复杂的图形算法，并支持各类开发平台，开发人员可以轻易的在各种开发环境中绑定 OpenGL 函数。OpenGL 要求把所有的几何图形单元都使用顶点进行描述，这样运算器和逐个顶点计算操作都可以针对每个顶点完成，然后进行光栅化操作。图像数据包括像素集、影像集、位图集等，像素处理的方式与几何定点处理的方式不同，但都进行光栅化的处理，最后经过逐个顶点的操作，把结果放入帧缓冲器中[3]。

3.2　OpenGL 图元描述和数据读入

在 OpenGL 中，提供了点、线、多边形等基本语句，利用这些语句可以通过程序将 DEM 转换为 ASCII 文件中的平面和高程数据快速读入，由于三角形是最基本的三维图形图元，基于三角形面片的各种算法简单高效，所以对于不规则三角网（TIN）数据存贮方式的 DEM，在 VC 环境下可使用如下的方法从文件中读入数据到指定的数组中：

```
for (int triloop = 0; triloop < numtriangles; triloop ++)// 遍历所有的三角形
{// 遍历三角形的每个顶点
  for (int vertloop = 0; vertloop < 3; vertloop ++)// 遍历所有的顶点
  {
  readstr(filein,oneline);// 读入一行数据
  sscanf(oneline, "%f %f %f", &x, &y, &z); // 读入各自的顶点数据
  // 将顶点数据存入各自的顶点
  // 区段 1 第 triloop 个三角形，第 vertloop 个顶点，值 x = x; y = y; z = z
  sector1.triangle[triloop].vertex[vertloop].x = x;
  sector1.triangle[triloop].vertex[vertloop].y = y;
  sector1.triangle[triloop].vertex[vertloop].z = z;
  }
```

3.3　基于 OpenGL 的场景模型读入及合并流程

地形数据一旦读入到程序中来，虚拟场景中的其它建筑物和配景均可依照真实情况进行场景的搭建，大多数情况是建模坐标的变换，图 1 给出了其基本流程。

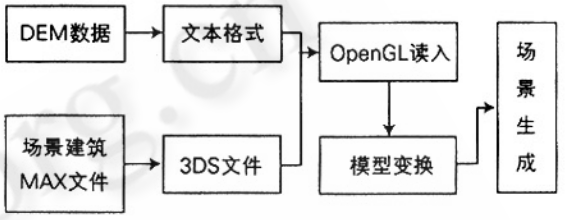

图 1　基于 OpenGL 的场景模型读入及合并流程

为减轻编程工作量和提高代码的执行效率，一般情况下，建筑物模型可转化为 3DS 或 OBJ 文件格式[5]，在 OpenGL 中使用显示列表来存贮。接下来，为使得真实感更强，需要给已经建成的场景添加纹理并指定贴图坐标。

4　OpenGL 纹理影射和天空盒

4.1　OpenGL 纹理载入

地形的纹理使用数码相机拍摄数字图像生成，或通过图像处理软件模拟生成，一般为数字图像。在 OpenGL 中，任意一块纹理可以映射到平面或曲面上，程序控制尚需要注意其纹理控制、映射方式和纹理坐标。可以使用下面的程序段载入纹理并给出纹理的映射方式和纹理控制方式：

```
int LoadGLTextures(){
int Status = FALSE;
    AUX_RGBImageRec * TextureImage[1];
    memset(TextureImage,0,sizeof(void *) *1);
    if (TextureImage[0] = LoadBMP("Data/DX_BD_015.bmp"))
    {
      Status = TRUE;
      glGenTextures(1, &texture[0]);
      glBindTexture(GL_TEXTURE_2D, texture[0]);
      glTexImage2D(GL_TEXTURE_2D, 0, 3, Texture-
```

```
Image[0] - > sizeX, TextureImage[0] - > sizeY, 0,
GL_RGB, GL_UNSIGNED_BYTE, TextureImage[0] - >
data); glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_
MIN_FILTER,GL_LINEAR); glTexParameteri(GL_TEXTURE_
2D,GL_TEXTURE_MAG_FILTER,GL_LINEAR);
    }
    if (TextureImage[0])
    {if (TextureImage[0] - > data) {free(TextureIm-
age[0] - > data); }
      free(TextureImage[0]);
    }
    return Status;
}
```

在执行绘图的代码段时，可使用 glBindTexture(GL_TEXTURE_2D, texture[i]) 来选者我们需要的第 i 个纹理，对于三角形面片的纹理贴图坐标，可通过区段的细分方法完成。

4.2 OpenGL 天空盒

在地形场景绘制完成后，一般情况下我们要给虚拟场景添加一个天空盒以便使得场景看起来更为逼真，天空盒在制作时可使用一个正方体（六面体）或半球体来完成，一般情况下，由于正方体天空盒制作简单，贴图容易控制，被广泛采用。图 2 为天空盒的展开图，可依据其进行贴图坐标和天空纹理选择，图 3 为加入天空盒的效果图。

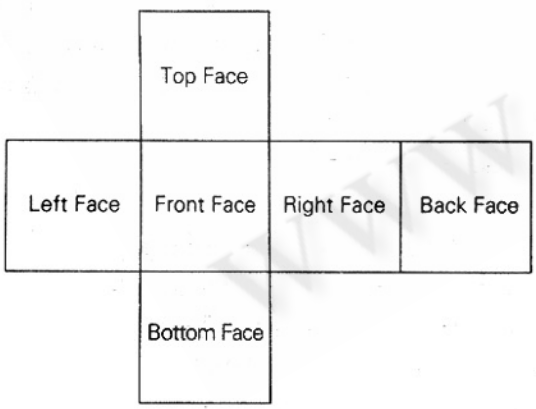

图 2　一般六面体天空盒的展开图

下面给出一个天空盒的贴图代码：

```
GLvoid DrawSkyBox()
{
    float boxSize = 3000.0f;
    glBindTexture(GL_TEXTURE_2D, texture[3]);
    glBegin(GL_QUADS);
      glNormal3f(0.0f, 1.0f, 0.0f);
      glTexCoord2f(0.0f, 1.0f); glVertex3f(-box-
Size, boxSize, -boxSize);
      glTexCoord2f(0.0f, 0.0f); glVertex3f(-box-
Size, boxSize, boxSize);
      glTexCoord2f(1.0f, 0.0f); glVertex3f(box-
Size, boxSize, boxSize);
      glTexCoord2f(1.0f, 1.0f); glVertex3f(boxSize,
boxSize, -boxSize);
    glEnd();
…//绘制其它五个表面
}
```

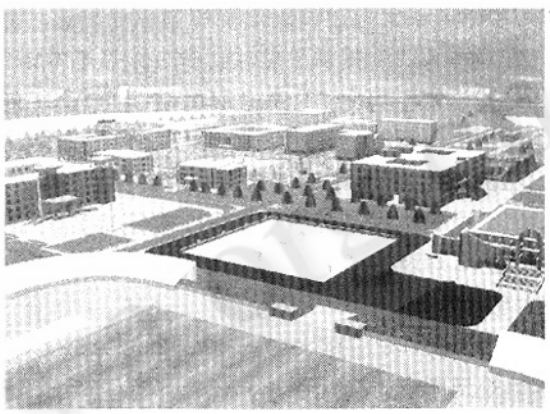

图 3　加入天空和雾效后的场景

5　虚拟场景漫游交互控制

5.1 地形漫游的第一人称视角控制

我们得到 DEM 输入后的数据后，其网格的 Z 坐标可直接进行使用，在 XY 平面的上漫游可在第一人称视角上可使用 gluLookAt() 函数来完成，在旋转方向上和前后方向上可使用如下的程序段，图 4 给出了算法原理。

```
void orientMe(float ang) {
    lx = sin(ang); ly = cos(ang);
    glLoadIdentity();
```

```
    gluLookAt(x, y, z, x + lx,y + ly,z + lz, 0.0f,
1.0f,0.0f);
}
void moveMeFlat(int i) {
    x = x + i*(lx)*movespeed;
    z = z + i*(ly)*movespeed;
    glLoadIdentity();
    gluLookAt(x, y, z, x + lx,y + ly,z + lz, 0.0f,
1.0f,0.0f);
}
```

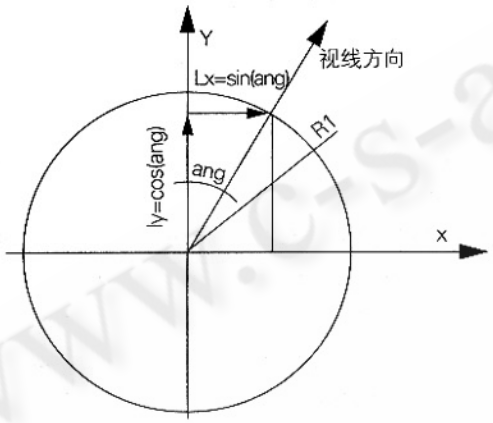

图 4　漫游视线方向的改变引起的 x,y 增量值

5.2 飞行地形漫游视图控制

对于飞行漫游控制而言，一般设置横倾角，纵倾角和航向角，可使用下面的程序段完成视图变换：

```
Void pilot (GLdouble planex, GLdouble planey,
GLdouble planez, GLdouble roll, GLdouble pitch, GL-
double heading)
    {glRotatef(roll, 0.0, 0.0, 1.0);
     glRotatef(pitch, 0.0, 0.0, 1.0);
     glRotatef(heading, 0.0, 0.0, 1.0);
     glTranslatef(-planex, -planey, -planez);
    }
```

6 实例

利用 DEM 描述的地域特征在地形相关的各个领域都有广泛的应用价值和广阔的应用前景，将其可视化后，可加入真实场景中的相应建筑或配景模型，我们通过上述方法实现了石河子大学中区的虚拟漫游系统，图 5、6 分别是来自两个视角的效果图。

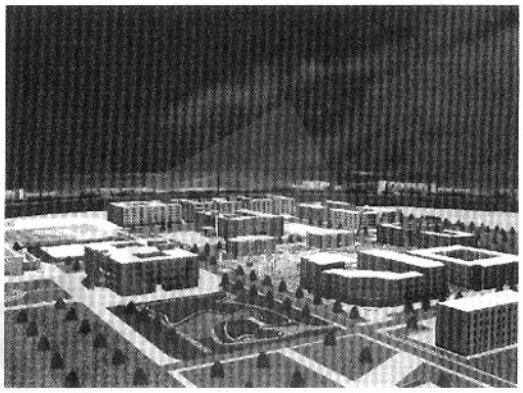

图 5　石河子大学中区飞行漫游效果图(东南向)

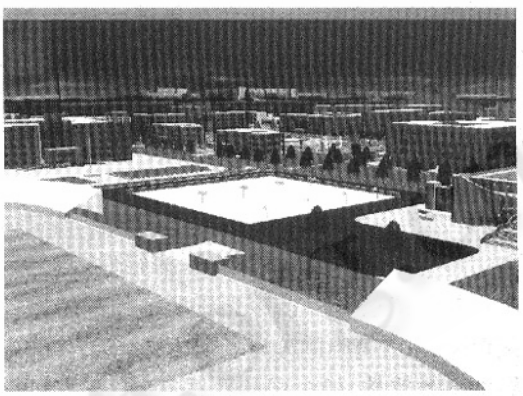

图 6　石河子大学中区飞行漫游效果图(西南向)

参考文献

1 Burdea G, Coiffed P. Virtual Teality Technology [M]. New York: John Wiley and Sons, Inc., 1994.

2 吕恒、江南，基于 OpenGL 和地形图支撑下的地形三维显示[J]，计算机工程，2004(4):174－177。

3 Mason Woo 等，OpenGL 权威编程指南(第三版)/吴斌等译[M]，中国电力出版社，2001。

4 Angel. E. 交互式计算机图形学－自顶向下方法与 OpenGL 应用(第三版)[M]，高等教育出版社，2003。

5 郭景、雷鸣，3DS MAX 模型在 OpenGL 中的读取与重现[J]，自动化与仪表，2002(5):46－49。