Computer Science and Application 计算机科学与应用, 2021, 11(10), 2579-2589
Published Online October 2021 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2021.1110261

Hans汉斯

基于Shapefile和OpenGL的电子地图绘制技术研究

朱国涛，胡文婷

海军航空大学青岛校区，山东 青岛

收稿日期：2021年9月23日；录用日期：2021年10月22日；发布日期：2021年10月29日

摘 要

针对航空装备模拟训练领域对电子地图显示的需求，介绍了Shapefile矢量电子地图的格式和基于SHAPELIB的矢量特征和属性读取，基于OpenGL图形引擎实现了电子地图点、线、面三种特征的绘制方法，基于Windows GDI引擎实现了OpenGL环境下的文字渲染和输出，最后设计和实现了电子地图可视化编辑工具和电子地图实时渲染模块，解决了军机仿真中机载电子地图加载和显示、二维战场态势电子地图显示的问题。

关键词

电子地图，Shapefile，OpenGL实时绘制，多边形栅格化

Research on Electronic Map Rendering Technology Based on Shapefile and OpenGL

Guotao Zhu, Wenting Hu

Naval Aviation University, Qingdao Branch, Qingdao Shandong

Received: Sep. 23rd, 2021; accepted: Oct. 22nd, 2021; published: Oct. 29th, 2021

Abstract

According to the requirements of electronic map display in the field of aviation equipment simulation training, this paper introduces the format of Shapefile and the reading of vector features and attributes based on SHAPELIB, realizes the drawing method of electronic map point, line and surface based on OpenGL graphics engine, and realizes text rendering and output in OpenGL environment based on Windows GDI engine. Finally, the electronic map visual editing tool and the

文章引用: 朱国涛, 胡文婷. 基于 Shapefile 和 OpenGL 的电子地图绘制技术研究[J]. 计算机科学与应用, 2021, 11(10): 2579-2589. DOI: 10.12677/csa.2021.1110261

electronic map real-time rendering module are designed and implemented, which solves the problems of airborne electronic map loading and display and two-dimensional battlefield situation electronic map display in military aircraft simulation.

Keywords

Electronic Map, Shapefile, OpenGL Real-Time Rendering, Polygon Tessellation

1. 引言

与传统的纸质地图相比，电子地图具有信息载负量大，阅读方便，具有动态显示功能，可显示大区域详情避免图幅拼接，可进行各种内容叠加，可进行更广泛的信息共享，具有更丰富的色彩，生产周期短，更新及时等特点，在各个行业领域都具有广泛的应用[1] [2]。

在航空装备模拟训练领域，电子地图也是不可或缺的功能之一。它不仅是构建战场想定编辑系统、作战态势显示系统的基础，而且目前大部分军用飞机模拟训练系统的建设也需要构建具有电子地图显示、浏览功能的机载电子地图仿真模块，因此无论是海军航空兵模拟仿真训练体系建设的要求，还是具体的飞机模拟训练系统建设、应用的具体需求，都需要针对此类应用设计和构建一个数据来源多样、运行稳定可靠、产品自主可控的电子地图模块。

2. 矢量电子地图 Shapefile 格式简介

目前随着地理信息系统技术的发展，各公司、机构都发布了自己的电子地图数据格式，这些电子地图格式各有特点。考虑到地理信息数据获取和交换的难易程度，我们选取 Shapefile 格式作为矢量数据格式。

2.1. Shapefile 地图格式

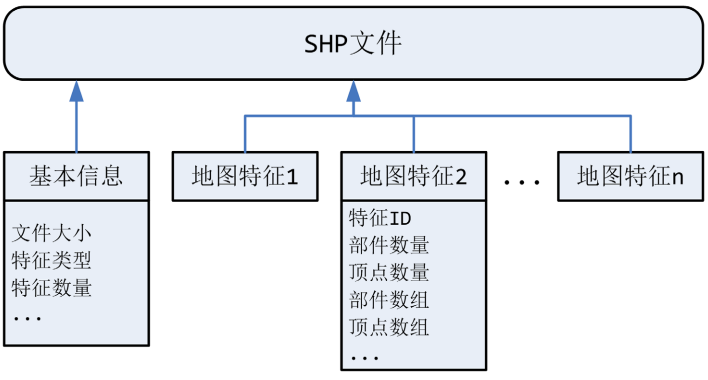

Figure 1. Structure and composition of Shapefile
图 1. Shapefile 文件结构和组成

Shapefile 文件是一种地理空间数据开发格式，由美国环境系统研究所(简称 ESRI)制定，目前该文件格式已经作为一个开放式的标准广泛应用于地理信息软件界，基本上诸如 MAPINFO、ARCGIS 之类的绝

大部分地理信息系统软件都能够支持导入、导出该格式的矢量地图数据[3] [4] [5]。

一个基本的 Shapefile 至少包含以下三个文件：分别是“.SHP”、“.SHX”、“.DBF”，其结构如图 1 所示。

.SHP 文件是图形格式文件。它用于保存地图特征的空间坐标信息(例如 X，Y，Z，通常 X 对应经度值，Y 对应纬度值，Z 对应高度值)。图 1 为.SHP 文件的结构及组成。一个.SHP 文件由一个固定长度基本信息头部和多个地图特征信息组成，其中基本信息包括文件大小、特征数量、特征类型(点、线、面)等要素，地图特征由特征 ID、部件数量、部件数组、顶点数组等要素组成。

.SHP 文件的地图特征类型有三类：点特征、线特征、面特征。点特征比较简单，通常直接由点的位置，也就是经纬度描述。线特征则是由一组点坐标串组成的；面特征由多个子环构成，所谓子环是一个至少有四个顶点构成的封闭且无自交现象的环，并且环上的第一个顶点必须和最后一个顶点重合，且顶点的顺序表示了面的方向(逆时针方向为正)。如图 2 所示的面特征，由三个子环即 V1→V6，V7→V13，V14→V17 组成，其中 V1→V6，V7→V13 两个子环分别以逆时针方向顶点分别描述了两个多边形面，而 V14→V17 则以顺时针顺序的顶点描述了一个被剔除的空洞。

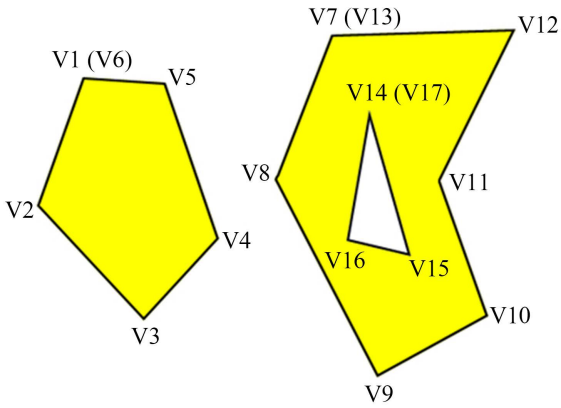

Figure 2. Face features described by Shapefile
图 2. Shapefile 描述的面特征

.DBF 文件是属性数据格式，它存储在由 xBase 数据库处理系统产生的数据库文件中，其中存储着每个地图特征的属性数据，如特征名称、特征的国标分类码等属性信息。

.SHX 文件是图形索引格式，在地图渲染中不使用该文件，所以不做过多介绍。

2.2. Shapefile 地图数据读取

Shapefile 作为一种在 90 年代初就已被应用的文件格式，有着良好的操作性和可交换性，大部分 GIS 软件都能够支持 SHAPE FILE 格式地理数据的导入、导出，但是依照 Shapefile 文件格式定义编程读取 Shapefile 文件中的地图特征数据仍然是一项繁琐的工作，幸运的是我们找到了一个可以进行 Shapefile 文件读写的开源库 SHAPELIB。

SHAPELIB (全称是 Shapefile C Library)目前的版本是 1.5.2，它是一个用于读写 ESRI 公司的 Shapefile 格式文件的 C 语言底层类开源库，它为 Shapefile 文件的读写提供了一组 C 接口的 API 函数，用于从 Shapefile 读取或者写入地图特征信息，它便于集成且使用简单，可以降低在其它应用程序中读写 Shapefile 地图特征信息的编程工作量，使用 SHAPELIB 从 Shapefile 文件中读取地图特征的空间位置信息和特征属性信息的代码如下：

```
// 首先打开 SHP 文件
SHPHandle hSHP = ::SHPOpen( shpfile , "rb" );
ASSERT(hSHP);
// 如果此 SHP 文件中存储的地图特征类型是面特征，则进入分支
if( hSHP->nShapeType == SHPT_POLYGON )
{
   // 获取 SHP 文件中特征数量
   int num;
   SHPGetInfo( hSHP, &num, 0 , 0 , 0 );
   // 依次循环读取 num 个特征
   for( int i=0 ; i<num ; i++ )
   {
      SHPObject* shapeObj = SHPReadObject( hSHP, i );
for( int j=0 ; j<shapeObj->nVertices ; j++ )
{
    // 依次读取特征顶点数组的坐标值
    _arCoord.PushIn(shapeObj->padfX[j] , shapeObj->padfY[j] );
}

for( int j=0 ; j<shapeObj->nVertices ; j++ )
{
    // 依次读取特征顶点部件的值(部件用于区分环的开始和结束)
    _arpSeps.PushIn( shapeObj->panPartStart[j] );
}
      SHPDestroyObject( shapeObj );
   }
}
// 如果 SHP 文件中的地图特征是线特征、点特征
// 则进入其它分支,此处代码省略
  ...
// 最终读取完成后，关闭文件
SHPClose(hSHP);
```

3. 基于 OpenGL 的电子地图绘制技术

地图作为客观世界与地理信息的显示载体，地图符号是描述地图信息的重要部分，在电子地图中，符号信息对于确定地图中的物体、地标和区域具有重要的作用，因此选用合适的矢量图元绘制样式将对电子地图的信息展示效果至关重要[6]。与 Shapefile 中的点、线、面三种特征相对应，地图符号通常也分为点符号、线符号和面符号，此外每种类型的符号都可能需要通过文字对其进行标注。

3.1. 点特征的绘制

点符号适合显示如机场、雷达、城市、村庄、飞机、舰船等具体的地图点特征，实践中可以采用栅

格图片对地图中的点特征进行表示，首先我们建立并制作了栅格点符号图库。如图 3 所示，该符号图库是一张底色透明、前景色为白色的 PNG 图片，图片大小为 64 × 64 像素，包含 4 行 × 4 列共计 16 个栅格点符号，每个栅格点符号的大小都是 16 × 16 像素。

Figure 3. Grid point symbol library example
图 3. 栅格点符号图库示例

在绘制上述符号时，首先根据符号类型计算符号在栅格图片中的纹理坐标，然后使用 OpenGL 纹理在屏幕上，以特征所在位置为中心绘制一个 16 × 16 的小矩形，部分绘制代码如下。

```
// 设置混合颜色，即符号的颜色
glColor4fv( spotColor );
// 绑定栅格符号库的纹理 ID
glBindTexture( GL_TEXTURE_2D , _npTexID );
// 计算纹理坐标参数
float nU1 = ( spotStyle % ( _bitmapW / 16 ) )*16.0f/_bitmapW;
float nV1 = ( spotStyle / ( _bitmapH / 16 ) )*16.0f/_bitmapW;
float nU2 = nU1+16.0f/_bitmapW;
float nV2 = nV1+16.0f/_bitmapH;

// 开始 QUADS 图元绘制
glBegin( GL_QUADS );
{
  // 输出左上角点
  ::glTexCoord2f(nU1 , nV1 );
  ::glVertex2f( px-8 , py+8);
  // 输出左下角点
  ::glTexCoord2f(nU1 , nV2 );
  ::glVertex2f( px-8 , py-8 );
  // 输出右下角点
  ::glTexCoord2f(nU2 , nV2 );
  ::glVertex2f( px+8 , py-8 );
   // 输出右上角点
  ::glTexCoord2f(nU2 , nV1 );
  ::glVertex2f( px+8 , py+8 );
}
glEnd();
// 绘制完成后，禁用纹理
glBindTexture( GL_TEXTURE_2D , 0 );
```

上述代码中，spotColor 指的是栅格点符号的混合颜色、spotStyle 指的是栅格点符号的样式，上述两个值由用户读取 Shapefile 后额外设置。程序执行过程为：首先设置符号颜色，并绑定符号库对应的纹理 ID；然后根据符号的 spotStyle 值计算栅格点符号在符号库中的相对位置(用纹理坐标表示)；最后依次绑定纹理坐标并将纹理贴到其对应的小正方形区域。

3.2. 线特征的绘制

线符号是通过定义图元样式如线宽、线型、颜色来表达河流、道路、边界、航线等地图线特征，在 OpenGL 中绘制时即线符号除顶点信息外，还要具备颜色 lineColor、线宽 lineWidth、线型 lineStyle 等三个要素。在 OpenGL 中，先特征的绘制比较简单，其代码如下：

```
// 设置线特征的颜色
glColor4fv( lineColor );
// 设置线特征的线宽
glLineWidth( lineWidth );
// 设置线特征的样式
glLineStipple( lineWidth , lineStyle );
// 开启 LINE_STRIP 图元绘制
glVertexPointer( 2 , GL_DOUBLE , 0 , _arCoord );
for( unsigned int i=0 ; i<_arpSeps.num()-1 ; i++ )
   ::glDrawArrays( GL_LINE_STRIP , _arpSeps[i] , _arpSeps[i+1]-_arpSeps[i] );
// 恢复默认样式,即默认为实线
glLineStipple( 1 , 0xFFFF );
```

对于部分铁路、公路等负责线特征，可以通过两次绘制方法进行。

3.3. 面特征的绘制

面符号主要通过填充颜色、透明度等表达陆地、水域、绿地、空域等地图面特征，采用 OpenGL 进行颜色填充面符号的绘制时，OpenGL 能够直接支持的只有简单的凸多边形，如图 4(a)所示，而从 Shapefile 中读取的面特征通常是一个复杂的面，大概率出现图 4(b)凹多边形或者图 4(c)带有空洞的多边形等情况，这种复杂面如果不经处理直接在 OpenGL 中绘制通常都很难得到预期正确结果。

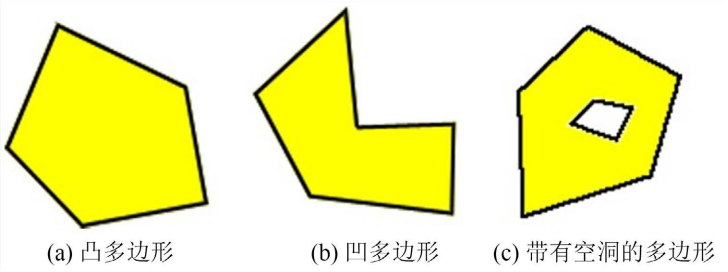

Figure 4. Simple faces in OpenGL and complex faces in Shapefile
图 4. OpenGL 中的简单面和 Shapefile 中的复杂面

从 Shapefile 中读取的面特征(使用多边形表示)有可能是非凸多边形，需要将其分解成更小的凸多边

形(如三角形)，再组合起来显示。OpenGL 工具函数库(即 GLU)提供了一组函数来完成这个操作，这个分解的过程叫做分格化(Tessellation)。使用 GLU 函数库进行多边形分格化的代码如下：

```
// 创建一个分格化器
GLUtesselator*    nTesselator = ::gluNewTess();
// 设置分格化函数的回调
gluTessCallback(nTesselator, GLU_TESS_BEGIN_DATA ,(void(__stdcall*)())TessEntiesB2);
gluTessCallback(nTesselator, GLU_TESS_END_DATA , (void(__stdcall*)())TessEntiesE2);
gluTessCallback(nTesselator,GLU_TESS_VERTEX_DATA, (void(__stdcall*)())TessEntiesV2);
gluTessCallback(nTesselator,GLU_TESS_COMBINE_DATA, (void(__stdcall*)())TessCombine2);
gluTessCallback(nTesselator, GLU_TESS_ERROR_DATA      , (void(__stdcall*)())TessErrorCB2);
// 设置分格化缠绕规则
::gluTessProperty( nTesselator , GLU_TESS_WINDING_RULE ,GLU_TESS_WINDING_ODD );
// 获取面特征内子环数量
int numCount = mapBinItemPtr->_arpSeps.numValid()-1;
// 开始面特征
::gluTessBeginPolygon( nTesselator, mapBinItemPtr );
for( int i=0 ; i<numCount ; i++ )
{
   int nS = mapBinItemPtr->_arpSeps[i];
   int nE = mapBinItemPtr->_arpSeps[i+1];
   // 开始子环
   gluTessBeginContour( nTesselator );
   for( int j=nS ; j<nE ; j++ )
      // 将子环的所有顶点都输入到分格化器
gluTessVertex(nTesselator,mapBinItemPtr->_arCoord[j].val,
mapBinItemPtr->_arCoord[j].val );
   // 结束子环
   gluTessEndContour( nTesselator );
}
// 结束面特征
::gluTessEndPolygon( nTesselator );
// 删除分格化器
::gluDeleteTess( nTesselator );
```

通过上述代码将面特征的所有子环以及子环的顶点信息输入到分格化器，分格化器在结束面特征绘制之后，就会根据我置的分格化缠绕规则进行分格化计算并将结果通过设置的回调函数通知我们。在回调函数中可以得到分格化后的结果，即复杂的非凸多边形被分格化成一组三角形，如图 5 所示。

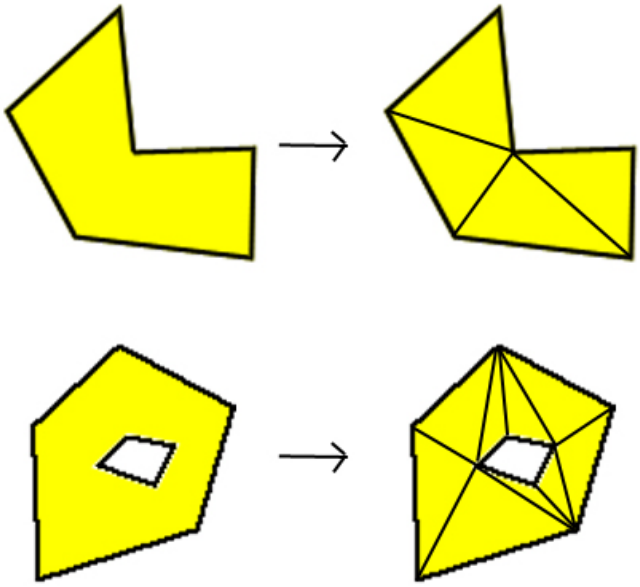

Figure 5. Schematic diagram of polygon meshing results
图 5. 多边形分格化结果示意图

多边形分格化完成以后对于颜色填充面符号的绘制，可以使用多边形风格化得到的一组三角面片来代替，代码如下：

```
// 设置填充的颜色
glColor4fv( fillColor );
glVertexPointer( 2 , GL_DOUBLE , 0 , _arCoord );
glDrawElements( GL_TRIANGLES , _indicesnum , GL_UNSIGNED_INT , _indices );
```

3.4. 文字符号的绘制

在绘制地图矢量特征时，通常需要使用文字对特征进行标注，这就需要在 OpenGL 中显示和输出字体符号。OpenGL 绘制字体有两种方式：栅格方式和纹理方式，前者使用 OpenGL 的栅格图像函数 glRasterPos 进行定位，然后将字体图片通过 glBitmap 函数直接绘制到 OpenGL 颜色缓冲区内；后者将需要绘制的字体图片制作成纹理，然后通过纹理贴图的方式将字体显示到世界坐标系中的某个方块上[7]。这两者各有优缺点，我们选用栅格方式进行绘图，共有三个阶段。

绘制准备阶段的主要内容是初始化字体引擎。字体引擎用于将文字转换成对应的文字图片，OpenGL 本身并没有内置字体引擎，在 Windows 下可以使用 GDI 的字体引擎来完成此功能。字体引擎的初始化主要有三个参数：字体类型，字体高度和是否粗体。字体创建完毕后，字体绘制的流程如图 6。

当用户指定一个起始位置绘制一个字符串时，首先需要将字符串由多字节字符集转换成 Unicode 字符集，这样使得字符串中每个字符都占用两个字节，长度相等便于索引。然后依次从 Unicode 字符串中取出字符，并检查该字符是否已经绘制过，如果没有绘制过则调用 GDI 生成该字符对应的图片、获取图片宽度、生成对应纹理并存入字符缓冲区；否则在字符缓冲区中查找该字符的缓冲数据，并使用缓冲的字符图片和纹理进行纹理贴图绘制。最后绘制完成一个字符后，起始位置会根据该字符图片的宽度改变，这样便能够从左至右绘制整个字符串。

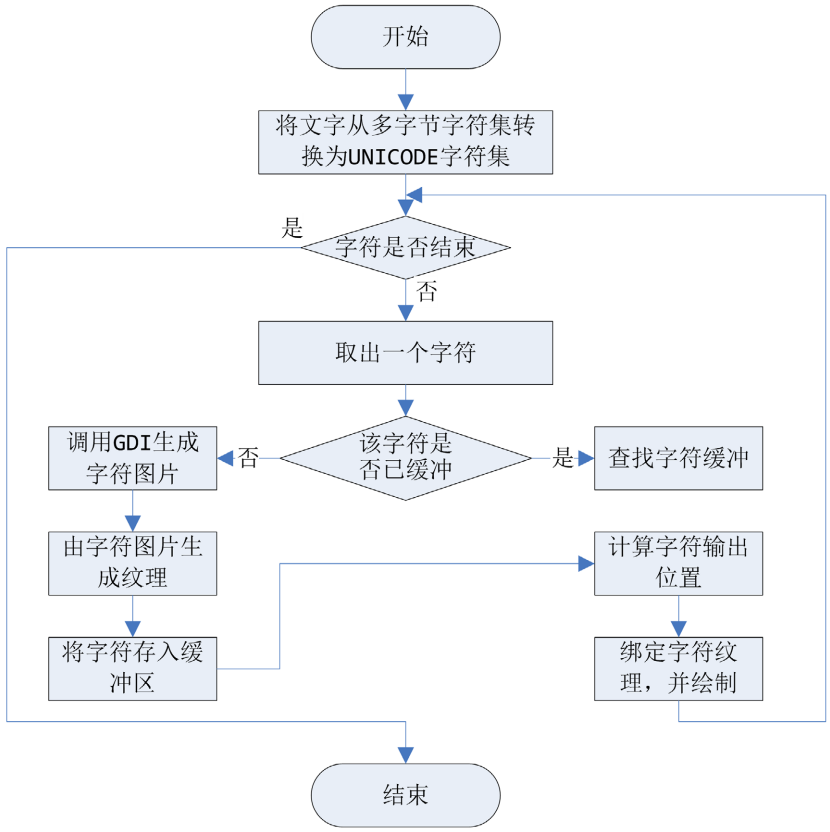

Figure 6. Text string drawing process
图 6. 文本字符串绘制的流程

4. 电子地图模块的总体设计

基于 Shapefile 和 OpenGL 的电子地图模块结构如图 7 所示，由地图编辑工具和地图绘制两部分组成。

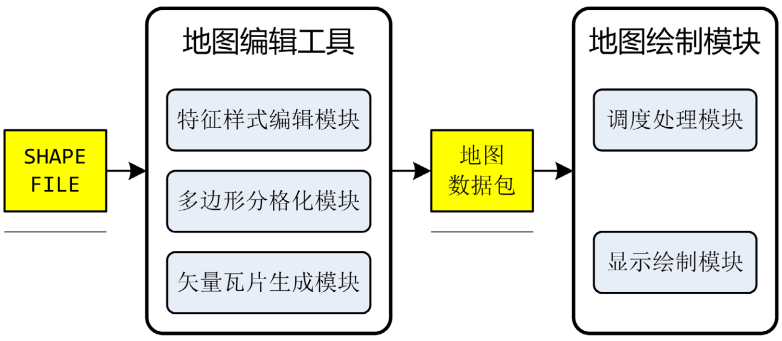

Figure 7. Structure of electronic map module
图 7. 电子地图模块总体结构

地图编辑工具是一个可视化的编辑工具，其界面如图 8 所示，它负责加载 Shapefile 文件并通过可视化的编辑环境设置特征的绘制样式、可见性，并且在输出时对地图中的面特征进行分格化处理，对地图矢量特征进行瓦片切割以便生成指定区域的矢量瓦片，最终处理结果保存成一个地图数据包文件。

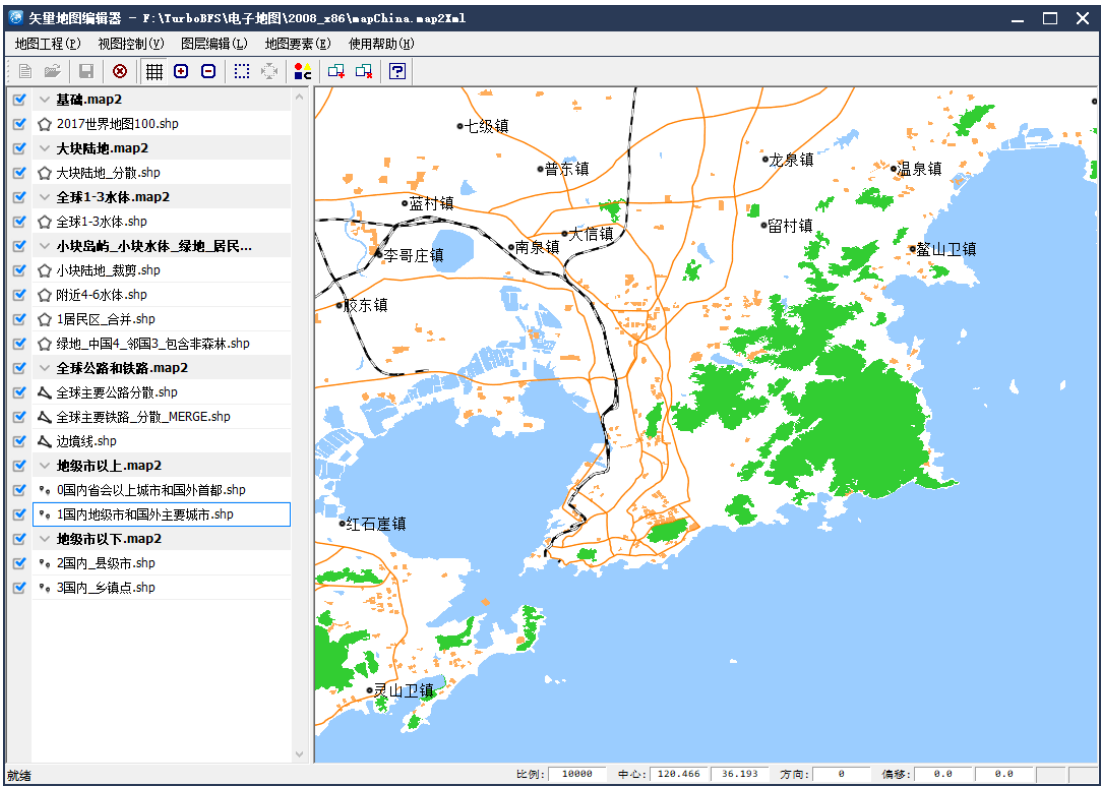

Figure 8. The Interface of Map's visual editing tool
图 8. 地图可视化编辑工具界面

地图绘制模块负责加载地图数据包并基于 OpenGL 渲染显示指定区域的地图图像，该模块提供 SDK 以便集成到其它程序中，使用该模块制作的某二次雷达程序、机载电子地图仿真程序实现效果如图 9 所示，该电子地图模块具有支持平移、旋转和无级缩放，支持不同的地图投影方式，可以通过可视化编辑工具修改地图的显示样式等功能特点，能够满足机载地图显示仿真、作战推演仿真、二维战场态势等系统对地图显示的要求，且其数据来源广泛，数据交互较容易，具有一定的应用价值。

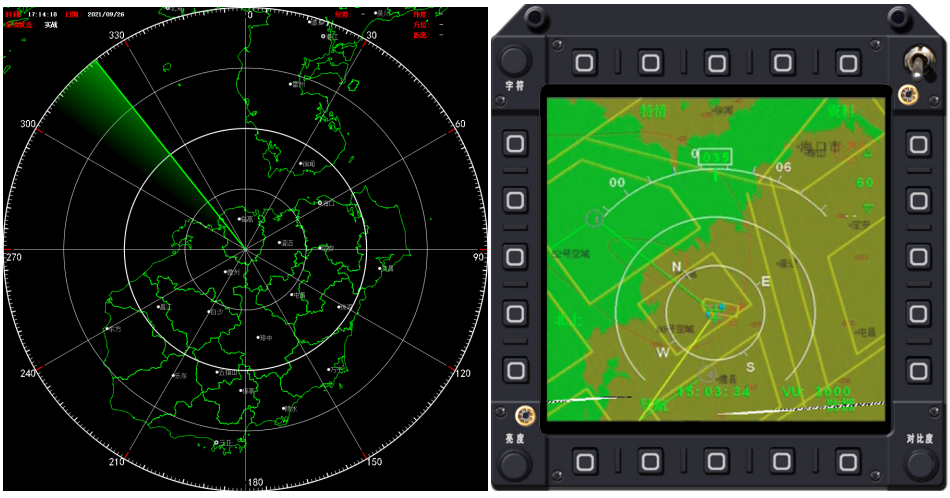

Figure 9. Application example of electronic map module
图 9. 电子地图模块应用示例

5. 结束语

无论是海军航空兵模拟仿真训练体系建设的要求，还是军机仿真电子地图显示的具体应用需求，考虑到地图数据获取和交换的便捷性，基于 Shapefile 电子地图格式和 OpenGL 图形渲染引擎，实现了通用的二维电子地图模块及可视化的地图编辑工具，该模块在硬件配置为 Intel Core i7-6700HQ 处理器，英伟达 GeForce 940MX 显卡，8GB DDR4 2133MHz 内存的硬件环境中，实测该地图模块的刷新率稳定保持在 50 到 60 赫兹。表明该地图模块可以维持较高的渲染效率，其数据易于获取、显示内容丰富、操作界面友好、信息直观立体，因此在航空装备仿真等方面具有一定的应用价值。

参考文献

[1] 赵永辉, 段云龙, 郭新望, 张明. 谷歌地图关键技术剖析于应用[J]. 海洋测绘, 2019, 39(3): 67-70.

[2] 何碧荣, 蔡倩. 基于 WEB 墨卡托投影的导航电子地图设计[J]. 计算机测量与控制, 2017, 25(1): 119-122.

[3] 李楚蒙, 熊庭刚. 基于 OpenVG 加速的矢量地图系统研究与设计[J]. 计算机与数字工程, 2017, 45(10): 2032-2036.

[4] 什么是 Shapefile? [Z/OL]. https://desktop.arcgis.com/en/arcmap/latest/manage-data/Shapefiles/what-is-a-Shapefile.htm

[5] 陈举平, 丁建勋. 矢量瓦片地图关键技术研究[J]. 地理空间信息, 2017, 15(8): 44-47.

[6] 朱秀丽, 周治武, 李静, 等. 网络矢量地图瓦片技术研究[J]. 测绘通报, 2016(11): 106-109, 117.

[7] 周玉科, 周成虎, 马廷, 等. 数字地图要素样式结构化存储表达研究与实现[J]. 地理与地理信息科学, 2012(3): 7-10.