基于 CNSDTF 和 OpenFlight 的空间数据互操作研究

常 远 秦小麟 王筱成 夏 斌
(南京航空航天大学信息科学与技术学院,南京 210016)

摘 要 多源空间数据共享一直是 VR-GIS(虚拟现实地理信息系统)应用系统开发中需要解决的重要问题。分析了 CNSDTF(中国地球空间数据交换格式)和 OpenFlight(一种应用于 VR 空间数据格式)的结构和特点,提出了以 CNSDTF 标准为空间数据转换中介,通过将 OpenFlight 数据格式转换成为符合 CNSDTF 标准的数据格式,具体实现了一种基于我国空间数据标准的空间数据互操作应用的方法。

关键词 GIS 空间数据 互操作 CNSDTF OpenFlight 虚拟现实
中图法分类号: TP331 **文献标识码**: A **文章编号**: 1006-8961(2005)12-1554-06

Research on the Spatial Data Interoperability Based on CNSDTF and OpenFlight

CHANG Yuan, QIN Xiao-lin, WANG Xiao-cheng, XIA Bin
(*Information Science and Technology Institute, Nanjing University of Aeronautics and Astronautics, Nanjing* 210016)

Abstract Sharing between various spatial data has always been a main problem to the development of VR-GIS, which is a new field integrating GIS and VR (Virtual Reality). This paper analyzed the structure and feature of CNSDTF (Chinese National Geo-spatial data transfer format) and OpenFlight, a spatial data standard format for VR. Then by regarding CNSDTF as a spatial data media-format, the format transformation from OpenFlight to CNSDTF was implemented. Thus this study advanced an effective solution for Spatial Data Interoperability based on CNSDTF.

Keywords GIS, spatial data, interoperability, CNSDTF, OpenFlight, VR

1 引 言

VR-GIS(虚拟现实地理信息系统)是一门综合了虚拟现实和地理信息系统的技术。由于 GIS 软件与虚拟现实 3 维建模软件的多样性,且每一种软件都有自己特定的数据类型、存储格式,因此,空间数据格式的不统一成为制约 VR-GIS 发展的重要因素。如何能够有效地实现 GIS 软件与 VR 软件间数据的共享与互操作,是 VR-GIS 需要重点解决的问题。

空间数据交换标准是国家空间数据基础设施的一个重要组成部分,也是目前实现空间数据共享的一个必要手段。NSDI(美国国家空间数据协会)制定了统一的空间数据格式规范 SDTS(spatial data transformation standards),我国也于 1999 年发布了"中华人民共和国国家标准地球空间数据交换格式"(简称 CNSDTF)[1]。该标准包括了几何坐标、投影、拓扑关系、属性数据、数据词典,同时规定了矢量和栅格两种空间数据的交换格式,适用于多种矢量数据、影像数据和格网 GIS 数据以及 DEM(数字高程模型)等的数据交换[2]。

基金项目:国家自然科学基金项目(49971063);江苏省自然科学基金项目(BK2001045)
收稿日期:2005-01-10;**改回日期**:2005-04-15

第一作者简介:常远(1981~),男。现为南京航空航天大学硕士研究生。主要研究方向为空间数据互操作、XML。E-mail:yuan_chang@126.com

OpenFlight 是一种用于存储空间数据和图像信息的空间数据格式,它能很好地与大多数 3 维建模软件融合,被认为是目前虚拟现实领域中空间数据格式的工业标准。CNSDTF 内容完备,包容性强,基本上包括了 GIS 数据体系中的所有内容。从其他 GIS 软件的数据转换到 CNSDTF,可以做到无信息丢失。如果能实现 CNSDTF 数据格式与 OpenFlight 数据格式之间的转换,对解决 VR 和 GIS 之间的空间数据互操作,促进二者的融合具有重要意义[3]。

2　CNSDTF 的数据结构组成以及数据转换方法

2.1　CNSDTF 数据结构组成

CNSDTF 空间矢量数据交换文件由 6 部分组成[4]:

〈矢量数据交换格式〉::=〈文件头〉〈要素类型参数〉〈属性数据结构〉〈几何图形数据〉〈注记〉〈属性数据〉

矢量数据交换格式的内容和结构如表 1 所示。其中“::=”符号表示其左边部分是由右边部分所组成;“〈〉”表示其中的内容应当被取代;“{}”表示其中的内容可以重复。

2.2　数据转换方法

通过 CNSDTF 转换数据的方法就是以 CNSDTF 为转换中介格式,其他类型的空间数据均通过与它的交互,实现空间数据的互操作。具体过程如下:

(1)通过 API 取出源数据格式中各相关要素的空间数据及其属性信息、标记信息。

(2)根据信息的不同属性、不同要素类型、不同的描述对象分别存入交换格式的不同部分,并与 CNSDTF 保持语义上的一致。

(3)文件保存为“*.vct”格式。

转换过程如图 1 所示。

表 1　CNSDTF 矢量数据交换格式的内容

Tab.1　The content of vector in CNSDTF

结构名称	组成部分	所描述内容
〈文件头〉	::=〈文件标志〉〈坐标单位〉〈坐标系〉〈投影类型〉〈原图比例尺分母〉〈有效数据时间〉	说明文件所包含的数据的一些信息。文件头以“Head Begin”和“Head End”作为开始和结束标志。
〈要素类型参数〉	::=〈要素类型编码〉,〈要素类型名称〉,〈几何类型〉,〈缺省颜色〉,〈属性表名〉{,〈用户项〉}	要素的基本属性信息、名称、类型等等。以“FeatureCodeBegin”和“FeatureCodeEnd”作为开始和结束标志。
〈属性数据结构〉	::={〈属性表定义〉},为每个要素定义一个属性表	为每个要素定义一个属性表。以“TableStructureEnd”和“TableStructureEnd”作为开始和结束标志。
〈几何图形数据〉	::={〈点状要素〉},{〈线状要素〉},{〈面状要素〉}	描述几何对象的坐标信息及拓扑关系。按几何类型,图形数据分为 3 大类:点、线、面。分别以“PointBegin”、“PointEnd”、“LineBegin”、“LineEnd”、“PolygonBegin”、“PolygonEnd”作为开始和结束标志。
〈注记〉	::=〈字体〉〈颜色〉〈字型〉〈大小〉〈间隔〉〈注记内容〉〈注记位置点数〉	描述注记的坐标位置、注记内容、字体、颜色、字型、大小等参数。以“AnnotationBegin”、“AnnotationEnd”作为开始和结束标志。
〈属性数据〉	::={〈属性表〉}	以“AttributeBegin”和“AttributeEnd”作为开始和结束标志。属性数据可由多个属性表组成,每一属性表的属性相对集中,由属性表名作为开始标志了,“Table End”作为结束标志。每个对象的属性占一行。

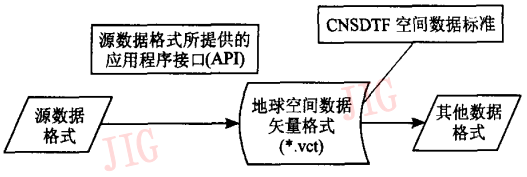


图 1　通过 CNSDTF 转换数据

Fig.1　Transform spatial data via CNSDTF

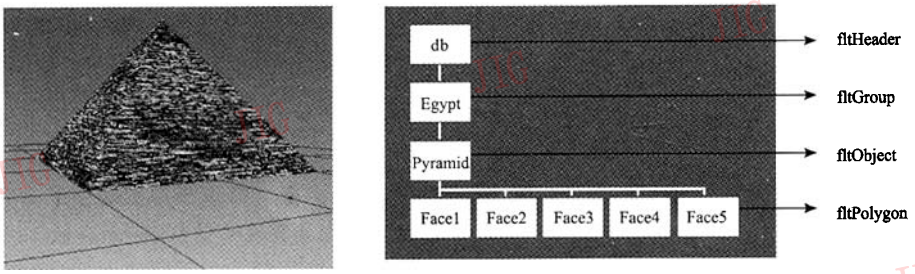
3　OpenFlight 格式数据结构

OpenFlight 是一种典型的层次型数据存储格式,其各层次数据之间通过树状结构特征描述各类要素之间的联系,子结点在空间位置上或功能上隶属于父结点,兄弟节点之间则存在着相关性。树状

结构中的每一个结点是一个记录类型，表示了一种类型的要素，其数据结构中存储该要素的空间信息和属性信息。

OpenFlight 把整个空间模型按照空间的位置分布和相关性，组织成逻辑上的组。例如，一个面或多边形是由几个顶点构成，几个相关的面（polygon）组成了一个对象体（object），几个对象又被组织成一个组（group），把空间对象有序地组织起来。

图 2 表明了一个金字塔的 3 维模型与 OpenFlight 数据格式之间的对应关系[5]。



(a) 金字塔所对应的层次结构　　(b) OpenFlight 树形结构

图 2　金字塔 3 维模型及其层次结构

Fig. 2　Pyramid 3D model and hierarchy

其中，fltHeader、fltGroup、fltObject、fltPolygon 是节点的记录类型名，即描述节点的数据结构名（OpenFlight 中的记录类型名都采用 flt * 的格式）。在图 2 的层次结构图中，“db”是数据库头节点，“Egypt”是组节点，其下只有一个对象节点，即“Pyramid”对象节点，表示金字塔对象；“Face1，…，Face5”则是几何节点，表示组成金字塔的 5 个面，这 5 个面分别由 3 个顶点组成，顶点则是 OpenFlight 中的基本几何元素。除了存储点、线、面等基本的几何要素之外，OpenFlight 中还储存了 LOD、DOF 等相关数据来支持实时场景截取，以及 3 维建模中的场景灯光、透明性、纹理、材质等其他属性数据，本文讨论的数据转换只涉及 OpenFlight 中与空间有关的点、线、面等要素，以及基本的属性信息。

4　两种数据格式之间的差异性和兼容性以及数据转换

任何两种数据格式之间的转换都要考虑到两者之间的差异性和兼容性，尤其是语义上的差异性[6]。表 2 描述了 CNSDTF 数据格式和 OpenFlight 数据格式的对应部分在语义上的主要差异和相关。

表 2　CNSDTF 与 OpenFlight 数据格式的对应部分

Tab. 2　Compare CNSDTF with OpenFlight

所描述的内容	CNSDTF 格式对应部分	OpenFlight 数据格式对应部分
对整个文件格式的宏观描述	文件头部分	Header 节点，对整个数据库文件的描述信息存放在 fltHeader 数据结构中。
要素的主要属性信息，如名称、类型、颜色、属性表	要素类型参数部分	每个节点要素的部分属性信息存储在其数据结构中。
要素的属性表，内容包括属性项个数、属性表名和字段描述	属性数据结构部分	每遍历到一个要素就为其建立一个属性表。
点状、线状、面状要素及其空间信息	几何图形数据部分	OpenFlight 中主要的要素类型是面状要素，即 Polygon、Mesh 节点
对要素的文字描述	注记部分	Text 节点，fltText 数据结构中是对其父节点的文字描述信息。
要素的全部非空间属性信息，以 CNSDTF 规定的格式存放在前面所定义的属性表中	属性数据部分	每个要素节点的非空间属性信息。

由表 2 可以看出,CNSDTF 文件格式中对不同类型、不同用途的数据进行了分类,并将相同类型、相同用途的数据集中存储。例如,几何图形数据部分集中存储了文件中所有的空间信息,属性数据结构和属性数据部分则存储了其他所有非空间属性信息,注记部分则是所有的文字描述,文件头和要素类型参数部分则描述了整个文件和要素的全局信息。

而在 OpenFlight 数据格式中,一个节点就是一个要素。有关该要素所有的信息,一部分存储在该节点的数据结构中,另一部分则以该节点子节点的形式存储。

根据 CNSDTF 和 OpenFlight 两种数据格式的差异性,可以在遍历转换的过程中采用如下的方案来保证这两种空间格式的兼容性:

(1)将目标输出文件按照 CNSDTF 的结构组成划分为 6 个部分,分别存储文件头、要素类型参数、属性数据结构、几何图形数据、注记、属性数据。

(2)为了提高转换效率,在遍历 OpenFlight 数据格式的同时写入 CNSDTF 数据格式。

(3)遍历 OpenFlight 的头节点,将该节点中的数据存储到 CNSDTF 的文件头部分。

(4)当当前访问节点为组节点或对象节点时,其中有一部分数据是要素的全局信息,需要写入要素类型参数部分。

(5)组节点和对象节点的下一层是面、线、点等节点,取出其中的空间信息写入几何图形数据部分。

(6)文字数据节点是对某个节点的注释,一般以该节点的子节点形式存在,取出其中的内容和信息写入 CNSDTF 注记部分。

整个数据转换的流程如图 3 所示。

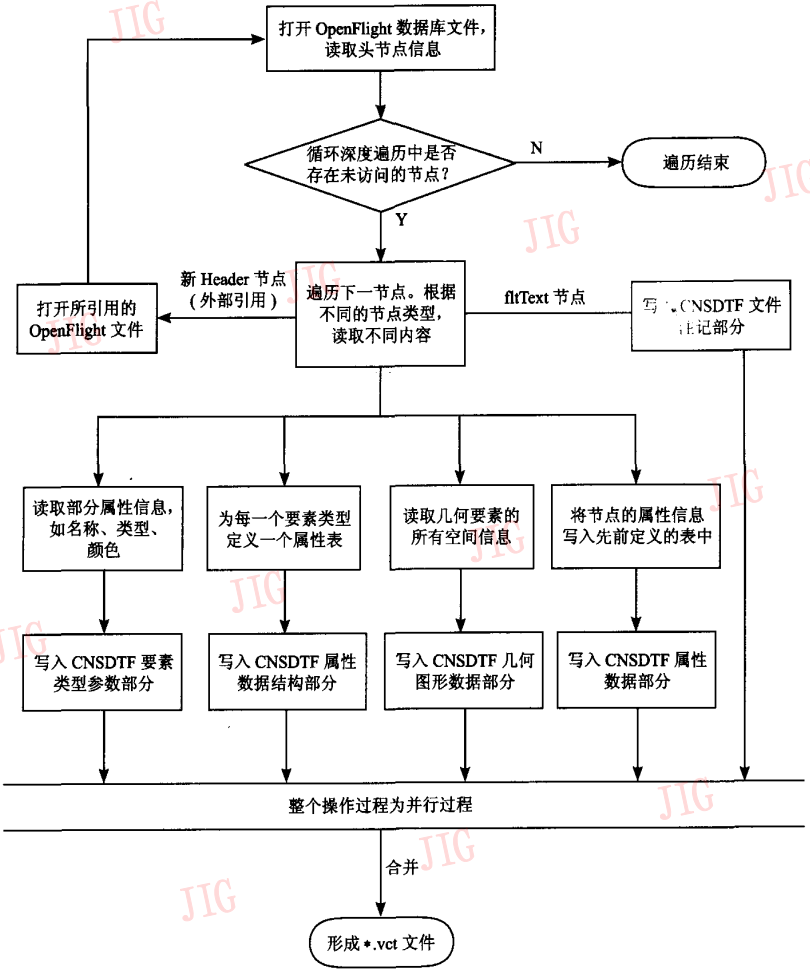


图 3　OpenFlight 和 CNSDTF 之间空间数据转换流程图

Fig. 3　Flow chart for spatial transformation between openFlight and CNSDTF

5 OpenFlight 与 CNSDTF 转换的具体实现

在进行数据转换之前,需要从 OpenFlight 中读取出相关数据。OpenFlight 对外提供了应用程序接口(API),OpenFlight API 是提供给外部应用程序的一组标准 C 语言的头文件和函数库。外部应用程序通过 API 中提供的函数,可以读取 OpenFlight 数据格式中的相关数据。

由于空间数据量非常庞大,以及空间数据层次结构的复杂性,读取算法必须保证在读取过程中不遗漏数据。根据 OpenFlight 树状层次结构的特点,可以采用深度优先的遍历方式。OpenFlight API 中提供了 mgGetChild( )、mgGetNext( )、mgGetParent( ) 3 个函数来控制遍历的方向。在节点的处理上采用递归算法,除此之外,在遍历的过程中还要判断一些特殊的节点,对这些节点进行特别的处理。

(1) 对 subface 节点的处理[7]

所谓 subface 是在一个面(face 节点,即 fltPolygon)中嵌入一个子面,它有自己的属性信息,因此在遍历的过程中要判断面节点是否存在子面,以免丢失这些要素的信息。

(2) 对引用节点的处理

在 3 维建模的过程中经常会将一个现有的模型进行各种操作,如复制、转移、变换等。如果存储变换前后所有模型的数据,数据量将是十分庞大的,从而导致效率的低下。在 OpenFlight 文件中把变换后的节点看作是对变换前节点的一个引用,而不是完全的拷贝,重复的属性信息只存储一次。因此,在遍历 OpenFlight 文件时,如果判断出一个节点是引用节点,则通过相应函数取出该节点的数据。

在 OpenFlight 中有两种引用方式,一种是内部引用,是指引用本文件中的一个节点;另一种是外部引用,是指引用另一个 OpenFlight 文件作为一个节点。为了节省存储空间,在原数据格式文件中只保存所引用文件的文件名,被引用文件与源文件位于相同的路径下。当遇到外部引用节点时,应该根据文件名打开所引用的文件,继而读取该文件中的数据。

具体的转换流程如下,其中,以 mg 开头命名的函数都是 OpenFlight API 中定义的,mgrec 是描述存储 OpenFlight 节点的通用的数据结构。

```
void main (int argc,char *argv[])
{
  mgInit (&argc, argv); /* 根据输入参数初始化 API 调用 */
  mgrec db = mgOpenDb ("*.flt"); /* 打开 OpenFlight 文件,返回树形结构的 Header 节点 */
  Traverse (db) ; /* 从 Header 节点开始,从上到下,从左到右遍历 */
  mgCloseDb (db); /* 关闭 OpenFlight 文件 */
  mgExit ( ); /* 退出 API 调用 */
}
static void Traverse (mgrec *node)
{
  mgrec *thisnode = node
  int nodetype = mgGetCode (thisnode) /* 获取节点类型 */
  switch (nodetype) { /* 根据节点的不同类型,采用不同的存取方法 */
    case flt***** : {
      Access*****(thisnode);
      Transform ( ) /* 将读出的数据写入 CNSDTF 文件的对应部分 */
    }
    case fltXref: { /* 判断出外部引用 */
      mgrec *newDb;
      newDb = mgOpenDb (filename);
      Traverse (newDb); /* 打开所引用的外部文件并且遍历所引用的文件 */
    }
  }
  if (nodetype) {
    if (mgGetChild(thisnode))
      Traverse(mgGetChild(thisnode)); /* 递归,遍历子节点 */
    if (mgGetNestedChild (thisnode)) /* 判断是否存在 subface 节点 */
      Traverse (mgGetNestedChild (thisnode)); /* 遍历 subfaces 节点 */
    if (mgGetReference (thisnode) && mgIsFirstInstance (thisnode))
      Traverse(mgGetReference (thisnode));/* 遍历引用节点 */
    if (mgGetNext(thisnode)) /* 向右访问兄弟节点 */
      Traverse(mgGetNext(thisnode));
  }
}
```

6 结　论

将应用于 VR 的 OpenFlight 空间数据格式转换为 CNSDTF 空间数据交换格式,实现了 GIS 领域和 VR 领域中两种空间数据标准的互操作,并给出了一个具体转换算法。该方法能够不丢失数据地从 OpenFlight 数据格式中取出数据,并且可以按照 CNSDTF 标准中所规定地格式存储数据,形成结构良好的 CNSDTF 文件,是一种有效的 GIS 与 VR 之间空间数据互操作的解决方案。

参考文献(References)

1 Wang Yan-dong, Gong Jian-ya. The data transfer method based on geo-spatial data transfer format[J]. Acta Geodaetica et Cartographica Sinica, 2000, **29**(2): 142 ~ 148. [王艳东, 龚健雅. 基于中国地球空间数据交换格式的数据转换方法[J]. 测绘学报, 2000, **29**(2):142 ~ 148.]

2 Huang Xing-yuan, Ma Jin-song, Tang Qin. An Introduction to Geographic Information System [M]. Beijing: Higher Education Press, 2001:87 ~ 90. [黄杏元,马劲松,汤勤. 地理信息系统概论[M]. 北京:高等教育出版社, 2001:87 ~ 90.]

3 Wang Yong-hong. Study on the interconnection and interoperability between urban 3D visualization and geographic information system [A]. In: International Geoscience and Remote Sensing Symposium [C], Toronto, Ont, Canada, 2002, **6**: 3535 ~ 3537.

4 GB/T 17798-1999, Chinese National Geo-spatial data transfer format [S]. [GB/T 17798-1999,中华人民共和国地球空间数据交换格式[S].]

5 Multigen-Paradigm. OpenFlight® Scene Description Database Specification, version 15. 8 [EB/OL]. http://www. multigen-paradigm. com/support/dc_files/of_spec_15_8. pdf, 2003-05-31/2004-10-28.

6 Lam, Sylvia. GIS interoperability: Current activities and military implications[A]. In: Proceedings of SPIE—The International Society for Optical Engineering[C], Orlando, FL, USA, 1997, **3085**:106 ~ 114.

7 Multigen-Paradigm. OpenFlight® API User's Guide V2. 6 [EB/OL]. http://www. multigen-paradigm. com/support/dc _ files/, 2003-04-08/2004-10-08.