存储遥感影像的一种多层 AVL tree 结构①

章文涛，吴玲琦

(武汉邮电科学研究院 虹信公司，武汉 430073)

摘　要：针对遥感影像数量庞大不利于发布和管理的特点，提出了一种用于离线卫星地图的空间索引结构，采用 AVL tree 把所需要的遥感影像数据存放在本地若干文件中，能以极快的响应速度将所需要的数据提供给桌面应用平台。

关键词：卫星影像；金字塔；AVL；行号树；列号树

Structure Based on AVL Tree for Storing Aerial Pictures

ZHANG Wen-Tao, WU Ling-Qi

(HongXin TeleCom Co., Ltd., Wuhan Research Institute of Posts and Telecommunications, Wuhan 430073, China)

Abstract: This paper presents a new structure for storing numerous aerial pictures. Storing aerial pictures in some files at local machine by using AVL tree, it can provide required aerial-picture data to application of desktop quickly.

Key words: aerial picture; picture pyramid; AVL; row number tree; column number tree

Google 地图在 2005 年一经推出即风靡全球，它使用户足不出户就可以将全球尽收眼底，它显示的底层是卫星遥感影像，这些影像数据是卫星影像和航拍数据的整合，具有信息丰富，影像直观，真实景观和测绘精度兼有，以及用途广等特点。随后，国内一些在线地图平台也相继推出了支持卫星遥感影像的地图服务。

然而，由于这些服务都必须基于互联网实现，且必须使用这些服务提供商提供的 API 进行操作，因此其应用范围受到了极大的限制，同时国内的一些在线地图平台目前并没有提供相应的 API 供开发者使用。这些情况导致无法在桌面应用平台（特别是在某些非 24 小时互联网在线的情况下）使用这些影像数据。

本文设计了一种用于卫星地图数据离线保存的空间索引结构，它把所需要的卫星影像数据利用在线时段通过文件方式存放在本地，在非在线时段采用实时访问文件的方法获取，能以极快的响应速度提供给非在线桌面应用平台。

1　在线影像地图结构

目前各在线地图服务提供商提供的卫星遥感影像一般都采用了影像金字塔技术，该技术对原始影像进行不同级别的采样，并生成分辨率不同的影像数据分级存储，金字塔底层影像分辨率最高，也最清晰，最高可以达到分米级，目前常见金字塔层数为 17-20 层，切换不同的层次可以达到放大和缩小的效果。

地图服务提供商将地图进行投影（一般采用墨卡托投影）后对每层采用网格样式进行组织，每幅图片占有其中一个网格（由行号和列号唯一确定），图片长宽均为 256 像素，一般采用 JPEG 或 PNG 格式，每一层影像都由这些图片拼接而成，如图 1 所示。

	第 1 列	第 2 列	……	第 N 列
第 1 行	(1, 1)	(1, 2)	……	(1, N)
第 2 行	(2, 1)	(2, 2)	……	(2, N)
	……	……	……	……
第 M 行	(M, 1)	(M, 2)	……	(M, N)

图 1　影像数据的组织

在金字塔顶端，只包含一幅图片（行列号均为 0），第二层该图片被分为 4 幅，依次每往下一层，上一层

① 基金项目湖北省软件专项项目(鄂信息联[2009]124 号)
收稿时间:2010-11-10;收到修改稿时间:2010-12-22

的每幅图片都将被分为 4 幅，因此金字塔底端图片数量为 22(n-1)幅（n 为金字塔层数）。

在互联网上，每一幅图片都对应一个 URL，该 URL 中包含图片所处金字塔层号、图片行号及列号等信息，给定一个正确的经纬度并确定层号后，经过一定转换，可以计算出该地点所在图片的行号和列号，并能计算出该点在对应图片中的位置，桌面应用平台可以通过这些信息组成 URL 在在线时段连接到网络并获取该图片用于显示和处理。

如果要在桌面应用平台中非在线的使用这种卫星遥感影像，需要从地图服务提供商处（或在在线时段）获取相应城市的金字塔各层影像图片，如城市地理区域很大，要求的分辨率很高，图片数量将非常庞大，如使用零散图片文件方式存储影像，占用的磁盘空间将会十分巨大，同时零碎的文件也不便于管理，频繁打开关闭文件将造成应用系统的处理效率低下。

因此需要设计一种存储结构，能将所需的影像图片存放在单个/若干个数据文件中，并能够根据任意给定的影像图片行列号，快速获取到对应的影像数据。由于 AVL tree[1,2]，是一种具有良好搜索效率的数据结构，其应可以满足这个设计需求。

2 AVL tree（Adelson-Velskii-Landis tree）

树（tree）是一种十分基础的计算数据机构，它由节点和边构成。二叉搜索树可以提供对数时间的元素访问，因此它具有良好的搜索效率，它的任何节点最多只允许两个子节点，分别称为左子节点和右子节点，并要求任何节点的键值一定大于其左子树中的每一个节点的键值，并小于其右子树中的每一个节点的键值。如果输入值不够随机，二叉搜索树可能会失去平衡，此时会造成搜索效率低落。

AVL tree 是一个加上了额外平衡条件的二叉搜索树，它要求任何节点的左右子树的高度相差最多 1，因此能够保证“对数深度（logarithmic depth）”平衡状态，使元素的平均访问时间较少。由于离线影像地图文件是预先制作好的，在获取图片过程中不必考虑平衡二叉树插入与删除操作的效率，因此 AVL tree 足以满足我们的需要。

3 遥感影像离线数据空间索引结构

影像金字塔的每一层数据中，由行号和列号可以唯一确定一幅显示图片，因此行号和列号均可当作 AVL tree 中的键值。

首先将影像金字塔中某一层的第 Y 列所有图片存放到文件中，每一个图片数据最开始四个字节记录该图片实际内容所占字节数，紧接着存放图片实际内容。

然后对该层第 Y 列的所有行建立一个“行号树”，其中节点的键值为该层第 Y 列所有的行号，记为 X，节点的实际数据为图片(X,Y)的数据在文件中的存储位置，如图 2 所示。

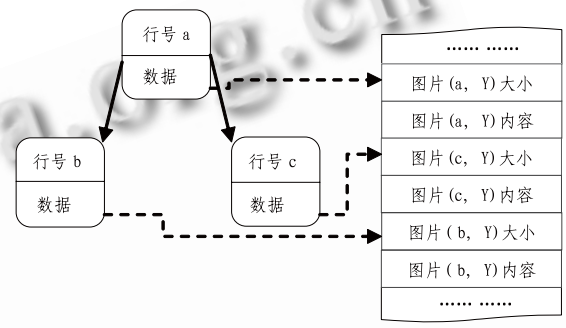

图 2 列号 Y 的“行号树”

依次对所有行建立“行号树”，再根据对金字塔该层的列建立一个“列号树”，其中节点的键值为该层所有列号，节点的实际数据为该列“行号树”中的根节点，如图 3 所示。

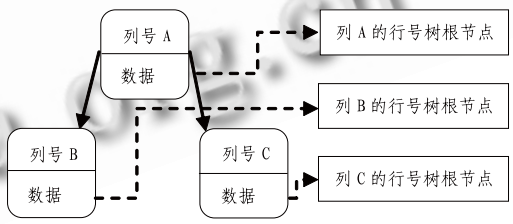

图 3 影像金字塔中某一层的“列号树”

对影像金字塔所有层建立“列号树”后，最后，使用一个数组存储每一层的“列号树”根节点在文件中的位置，这样可以很容易的获取影像金字塔指定层的某一图片数据。

4 实际应用

4.1 建立存储

在存储树节点时，先存储其键值数据，然后存储节点数据（对于“行号树”为影像图片在文件中位置，对于“列号树”为“行号树”根节点在文件中位置），

其次存储左子节点在文件中位置（没有该节点记 0），最后存储右子节点在文件中位置（没有该节点记 0）。

由于需要保持树的平衡，树必须先建立好，又由于树上一级节点存放下级的位置，因此需要先存储图片数据，再存储“行号树”，最后存储“列号树”，每个树都采用后序遍历的方式，最后存储的总是根节点数据。

4.2 影像检索

对于给定金字塔一层的某一图层编号（包括行号和列号），根据二叉搜索树的原理，可获取对应的影像数据。主要步骤如下：1：初始化。读取影像金字塔每一层数据在文件中的起始位置，即上节描述的“列号树”根节点位置；2：根据给定的层号，通过第一步读取的位置信息，定位到文件指定位置，结合二叉树检索算法，不用读取所有节点数据，即可读取到键值为指定列号的节点；3：根据第二步所获取的“列号树”节点信息，其数据部分即“行号树”根节点在文件中的位置，结合检索算法，读取到键值为指定行号的节点；4：根据第三步所获取的“行号树”节点信息，根据其数据中存储的影像图片位置定位到文件并读取出影像数据图片内容。

另外建立适当的缓冲机制，可以使读取节点的步骤减少，进一步提高检索效率。

4.3 性能对比

为体现本文描述结构的性能，选用某城市卫星地图制作测试用地图文件，并与使用零散图片文件存储影像的方式进行了对比。

试验计算机采用英特尔奔腾 2.4G 双核 CPU，2G 内存，操作系统为 Windows XP，程序采用 C++编写，表 1 为读取该城市影像金字塔一层中的所有图片内容所花费的时间对比（单位为毫秒）。

表 1　读取相同影像数量所花费的时间对比

层号	图片数量	本文结构(ms)	零散文件(ms)
6	16	0	0
5	56	0	16
4	224	0	47
3	837	16	188
2	3180	46	1422
1	12600	125	7156
0	49504	485	28500

表 2 为在相同时间内读取影像金字塔最底层中的图片数量对比（由于计时方式和表 1 不同，因此结果和表 1 有一定差异，但不影响对比结果）

表 2　相同时间读取的影像数量对比

花费时间(ms)	本文结构	零散文件
50	4159	159
100	8458	311
200	17869	569
300	26402	758
400	36608	1138
500	46799	1430

从表 1 和表 2 可以看出，采用本文结构读取数据效率大大高于零散文件方式，如果采用合适的缓冲算法，其优越性将更加无可比拟。

4.4 实际效果

图 4 为本文算法应用于试验用桌面平台的效果图，可以快速对地图进行缩放、漫游等操作，其实时性完全满足现实需求。

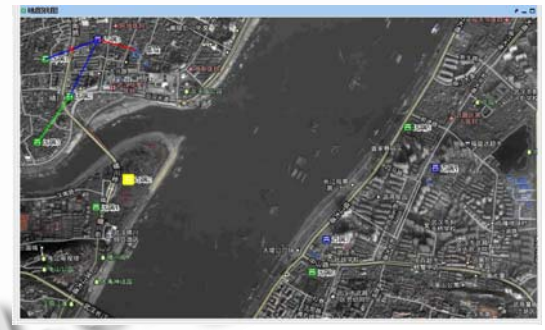

图 4　实际效果图

5　结论

本文提出采用 AVL tree 对影像数据建立多层索引结构，实现了对 GB 级的影像数据的实用管理和显示，结合内存映射、双缓冲等技术，提高了影像地图数据的处理效率，实践证明，该方法完全能满足桌面应用平台的实时性显示要求。

参考文献

1 Adelson-Velskil GM, Landis EM, An algorithm for the organization of information. Proc. of the USSR Academy of Sciences, 1962,146:236-266.

2 Sedgewick R, Addison-wesley. Algorithms, 1983. 191-199.