一种粗精度三维模型的生成与快速显示方法

赵 龙, 秦 昆
(武汉大学 遥感信息工程学院, 武汉 430079)

摘 要: 为了解决用传统方法简化后的复杂三维模型表面存在的裂缝,以及纹理拉伸、缺失、错位等严重影响模型视觉效果的问题,提出一种粗精度模型的生成与快速显示方法。该方法利用 Billboard作为模型的载体,利用 GPU技术实现 Billboard随视点的旋转功能,利用视点与 Billboard之间的角度动态地更换相应的纹理图片。实验表明,从远处看,用该方法生成的 Billboard与原始模型相差无几,完全可以用 Billboard替代原始模型;从近处看,用 Billboard替代的模型有细微的几何畸变。因此,该方法适合 LOD较粗级别的显示,特别是对于拥有大数据量的数字城市和数字地球系统,其不仅保证了远处模型的视觉效果,而且还大大提高了系统的运行效率。

关键词: 模型简化; Billboard; 图形处理器; 细节层次; 数字城市; 数字地球
中图分类号: TP391 文献标志码: A 文章编号: 1001-3695(2011)06-2381-05
doi 10 3969/j issn 1001-3695 2011 06 106

Method of generation and rapid showing for 3D rough models

ZHAO Long QIN Kun
(School of Remote Sensing & Information Engineering, Wuhan University, Wuhan 430079, China)

Abstract: In order to solve the problems that seriously affected optical effects of simplified complex 3D models generated by traditional methods, such as cracks, texture stretches, flaws, dislocations and so on, which existed on surfaces of the models, this paper proposed a method of generation and rapid showing for 3D rough models. It used Billboards as carriers of models, implemented the function of rotating Billboards followed by viewpoint with the usage of GPU technology, replaced with available texture image dynamically from the angle between viewpoint and Billboard. Experiments show that at a distance, Billboards generated by this method are almost the same with original models and can absolutely replace them; at close range, Billboards have inappreciable geometric distortion. Therefore, the method is suitable for coarse-level LOD, especially for digital city and digital earth system with magnanimity data, which not only guarantees the optical effects of original models, but also greatly enhances the efficiency of the system.

Key words: model simplification; Billboard; GPU(graphic process unit); LOD(level of detail); digital city; digital earth

0 引言

随着计算机技术和地理信息系统的发展,二维地图已经无法满足人们的需求,人们希望通过虚拟现实技术来实现显示效果逼真的三维地图,由此引发了数字城市和数字地球的迅猛发展。海量模型的复杂性和多样性已经大大超出计算机图形硬件的处理能力,为此人们提出了多种优化方案,其中细节层次(LOD)技术的出现,成为三维图形生成加速最有效的处理方法。该方法将不同逼近精度的几何模型组织在一起,这些几何模型都保持了原模型的可视特性,根据视点与模型的距离选择适当模型表示物体,一般来说距视点越近,选择模型的精度就越高,反之,模型精度就越低。因此,如何生成不同精度的模型成为 LOD技术的关键之一。20世纪 90年代以来,国内外许多学者就如何生成不同精度的简化模型进行了大量的研究。Wu等人[1]提出了基于随机多重选择的流式简化算法; Shaffer等人[2]提出了基于自适应空间划分的顶点聚类简化算法; Garland等人[3]结合顶点聚类和边折叠简化算法[4]的优点,提出了一种新的混合简化算法; DeCoro等人[5]使用 GPU的最新功能——几何着色器实现了基于顶点聚类的实时简化算法;费广正等人[6]在 2002年提出了基于细节迁移的快速外存模型简化方法;李蔚清等人[7]提出了一种基于特征的实时 LOD模型生成算法。

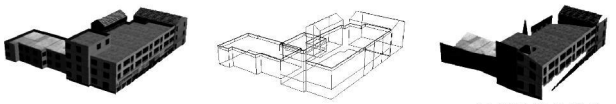
上述算法都是基于网格的简化,对单个复杂的网格模型的简化是很有效的,但是对于由多个简单模型组成的复合模型的简化不是很理想,特别是在模型简化程度比较高的情况下,模型的表面经常产生一些裂缝,纹理的拉伸、缺失、错位等现象,严重影响了模型的视觉效果。如图 1所示,(a)是一个建筑模型;从(b)中可以看出该建筑模型是由多个简单模型(主要是长方体)组成;(c)表示在简化程度比较高的情况下(简化后模型的边数目是原始模型的 20%)通过边折叠算法对建筑模型简化后的过简化模型。从图 1(c)中可以明显地看出模型的表面产生了一些裂缝,模型的纹理出现了拉伸、缺失、错位等现象。出现这些现象的原因是:虽然原始模型是一个复杂的模型,组成它的简单模型已经到了不可再简化的地步,如果对这些模型强行简化,必然会造成简化后的模型变形严重,本文称这种现象为过简化现象。

本文着重讨论了一种粗精度模型的生成与快速显示方法。

收稿日期: 2010-10-20; 修回日期: 2010-11-30

作者简介: 赵龙(1983-),男,硕士研究生,主要研究方向为空间分析、数字地球(305024828@qq.com);秦昆(1972-),男,教授,博士,主要研究方向为地理信息系统的理论和应用。

该方法不对模型简化,而是用一个 Billboard(广告板)来代替模型,然后从各个角度对原始模型拍照,得到一系列照片,根据当前视点与模型的方位关系确定显示在 Billboard中的照片。其中 Billboard的旋转与照片的确定是通过可编程 GPU 硬件技术来完成的,这样做不但可以将 CPU 从繁重的任务中解脱出来,而且也提高了 GPU 的利用效率。该算法与模型的复杂度无关,同时也不会在模型的表面产生纹理的拉伸、缺失和错位等现象。



(a) 原始模型 (b) 原始模型的网格结构 (c) 通过边折叠算法生成的简化模型(20%)

图 1 原始模型和通过边折叠算法简化的模型之间的对比

1 相关技术简介

Billboard技术是虚拟现实中一种常用技术,它实质上是用一个贴有图片的四边形来表示比较复杂的物体(如树木),该四边形可以绕任意点或任意轴旋转,但旋转的结果必须是四边形正向正对着视点。该技术的最大优点是能大大提高场景的显示速度。由于复杂的模型往往是由成千上万的三角形组成,如果用一个四边形来代替这些复杂的模型,其优势是不言而喻的。目前 Billboard技术已经成功运用到树木的模拟、三维云的模拟、火焰的动态模拟、阴影的模拟等方面[8~11]。本文把 Billboard作为照片的载体并代替原始模型显示到场景中,这样做的目的主要是为了简化模型,加快渲染速度。GPU 是三维图形处理器,它的设计初衷是为了减轻 CPU 的计算负担,承担一部分如矩阵计算、顶点计算、光照计算等图形处理功能。GPU 具有并行计算、高速浮点计算等特性,同 CPU 相比,GPU 的运算速度更快、效率更高。GPU 技术已经在几何计算、流体模拟、频谱的变换和滤波、数据管理等领域取得了可喜的成绩[12~15]。本文利用 GPU 实现了 Billboard的旋转平移和图片筛选,提高了 GPU 的利用效率,加快了场景的渲染速度。

2 模型照片的拍摄与存放

2.1 模型照片的拍摄

2.1.1 投影类型的选择及相机参数的设置

在透视投影(perspective projection)情况下,如图 2所示,如果在拍摄点 p 对物体拍照,将拍摄的照片显示在 Billboard上,此时从距离物体很远的视点 E 看 Billboard模型与看原始模型相比,必然存在几何畸变(除非视点 E 与拍摄点 p 重合)。为了简化计算和消除几何畸变,可以在正射投影(orthographic projection)下对物体拍照。从图 2中可以看出,当视点 E 到物体 AB 的距离 distance越大时(distance表示从视点 p 到模型包围球中心的距离),视点 E 能看到物体 AB 的视域夹角 θ 就越小。由于粗精度模型距离视点都比较远,可以近似地认为 EA 平行于 EB,所以在正射投影下拍摄可以抵消部分几何畸变。

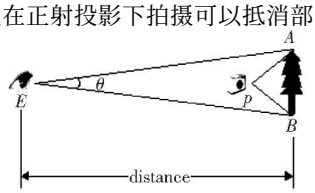


图 2 视点距物体的距离 distance 与 θ 之间的关系

正射投影参数包括正射投影面高度 h、宽度 w、近裁剪面的距离 z_{near}和远裁剪面的距离 z_{far}。其中,设 R 为包围球的半径,为了防止物体被裁剪掉,近裁剪距离的取值是 $0 < z_{near} < distance - R$,远裁剪面的取值是 $z_{far} > distance + R$。

对于正射投影参数高度 h 和宽度 w 的设置必须既要考虑到相机照出的照片能完全反映出模型的全貌,也要避免模型在照片中的显示区域过小。模型的显示区域是指模型在照片中的显示范围和位置。由于是正对相机拍照的,模型显示区域的中心与照片的中心必定重合,所以不需要考虑显示区域的位置,如图 3(a)所示。因此 w 和 h 必须既要都大于或等于包围球的直径 R,又不宜过大,过大会导致照片中模型的显示区域过小,这不但浪费存储资源,也会造成照片的严重失真,如图 3(b)和(d)所示。

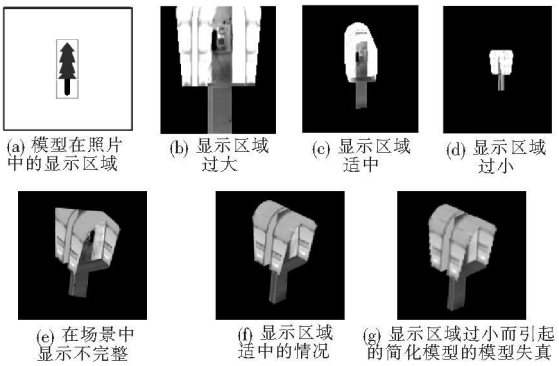


图 3 正摄投影参数设置的不同所引起的照片显示区域的不同以及模型的失真情况

在图 3(b)中,因为模型的显示区域大于照片的显示范围,模型的显示区域被裁剪而无法显示全貌,所以在场景中如图 3(e)所示,当将被裁剪的照片显示到 Billboard中时,从视点 E 看到的是不完整的模型;图 3(d)中模型的显示区域远远小于照片的显示范围,照片大部分区域未被利用,造成存储资源浪费;在图 3(g)中,由于模型的显示区域过小,当把照片显示到 Billboard中,与原始模型相比,失真严重。由于照片的宽高比 aspect与正射投影面的宽高比相同,故 $aspect = w/h$。而 $min(w, h) = R$,$min(w, h)$表示 w 和 h 的最小值,则

$$w = \begin{cases} R & aspect \leqslant 1 \\ R \times aspect & aspect > 1 \end{cases}$$
$$h = \begin{cases} R/aspect & aspect \leqslant 1 \\ R & aspect > 1 \end{cases} \quad (1)$$

2.1.2 相机的拍摄位置与相机拍摄姿态的设置

相机的拍摄点是分布在以模型包围球中心为原点的半球面上,如图 4(a)所示。半球表面分布若干条经线和纬线,每相邻两条经线相差的经度相同,每相邻两条纬线相差的纬度也相同。为了计算方便,为这些经线和纬线编号,指定与 z 轴正方向相交的经线为 0°经线,其经线编号为 1,从 y 轴正向向负向看,经线编号按逆时针递加;极点处(90°纬线处)编号为 0,距离极点最近的纬线编号为 1,按低纬度依次递加。经线和纬线的相交处即是相机的拍摄点。为了确保相机从各个角度都能看到模型的全貌,必须确保相机在包围球以外拍摄,即相机所在半球半径要大于模型包围球半径。为了防止误差,建议使半球半径略大于模型包围球半径。在拍摄过程中,相机要朝着包围盒的中心拍摄,相机的向上方向与相机所在半球面经线的切线方向一致,并朝向 y 轴的正向(相机向上方向的 y 分量大于或等于 0)。如图 4(b)所示,其中 p 点为拍摄点,O 点是模型包

围球的中心, pO 为相机拍摄方向, *up* 为相机向上方向, *up* 的计算方法 (以右手坐标系为准)为

$$up = \mathrm{nor}(pO) \times (\mathrm{nor}(Op) \times y) \quad (2)$$

其中: nor(pO)表示向量 pO 的单位向量, nor(Op)表示向量 Op的单位向量; *y* 是指向 *y* 轴正向的单位向量。特别地, 当相机在 90°纬线 (在极点处)拍摄的时候, 相机的向上方向为 *z* 轴, 拍摄方向为 *y* 轴的负方向。

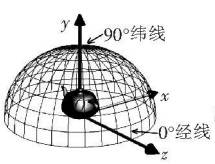
在拍摄的时候, 还必须注意将背景色的透明度设置为比较低的值, 建议设置为 0, 这样做是为了在显示过程中能方便地滤掉背景色。

2.2　模型照片的存放

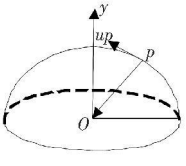
由上面分析可得, 拍摄点的个数等于模型照片的个数。当分布在半球上的经线数目 *T* 与纬线数目 *Q* 比较多的时候 (*Q* 不包含 90°纬线, *T* 和 *Q* 由用户指定), 产生的照片数量将非常庞大, 其具体的计算式为

$$n\mathrm{Count} = Q \times T + 1 \quad (3)$$

其中: *n*Count表示照片的数量, *Q* 表示经线数目, *T* 表示纬线数目 (*Q* 不包含 90度纬线), + 1代表加上在极点处拍摄的照片。如果将这些照片分别存放在文件系统中, 不但会导致文件系统中文件的数量非常庞大, 也会因频繁访问 I/O而造成系统效率降低。为了解决这个问题, 本文将同一模型生成的所有照片合并到一张图片中, 如图 5所示。



(a) 相机拍摄点的分布



(b) 相机的拍摄姿态

图 4　拍摄点分布即相机的拍摄姿态

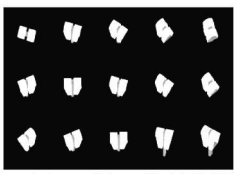


图 5　同一模型的照片合并到一张图片中

为了提高效率, 将图片的宽和高 (像素值)设置为固定值或由用户指定, 而每张照片的宽和高 (像素值)随着照片的数量而改变, 但照片的宽高比始终为 aspect。例如, 一张 1024 × 1024的图片, 假定照片的 aspect= 1, 如果存放一张照片, 那么照片的宽和高都是 1024; 如果存放四张照片, 那么照片的宽和高都是 512。为了最大限度地利用图片的存储空间, 可以采用递归试探的方法来确定照片的宽和高。该方法的主要输入参数有: 图片的宽 *TW* 和高 *TH*, 照片的数量 *n*Count, 照片的宽高比 aspect和试探值 *N*。*N* 表示图片能存放照片的列数, *N* 的初始值为 1。其主要步骤为:

a)根据 *N* 确定照片的宽度 *PW* 和高度 *PH*, $PW = \blacktriangledown TW/N\lrcorner$, $PH = \blacktriangledown PW/\mathrm{aspect}\lrcorner$, 其中▼」表示向下取整。

b)判断 $PW \times PH \times n\mathrm{Count}$是否大于 $TW \times TH$, 如果大于, 则说明 $TW \times TH$ 的图片无法容纳 *n*Count个 $PW \times PH$ 照片, 此时 *N* ++, 转到步骤 a)。

c)计算图片能容纳照片的行数 $DH = TH/PH$。

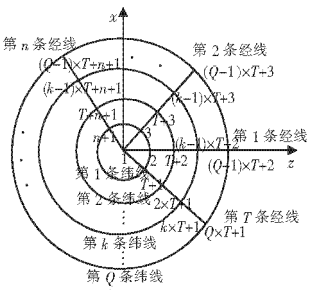
d)判断 $DH \times N$ 是否小于 *n*Count, $DH \times N$ 表示 $TW \times TH$ 的图片能容纳 $PW \times PH$ 照片的最大数目。如果小于 *n*Count, 则说明 $TW \times TH$ 的图片无法容纳 *n*Count个尺寸为 $PW \times PH$ 的照片, 此时 *N* ++, 转到步骤 a), 否则转到步骤 e)。

e)返回 *N* 值, 跳出递归。

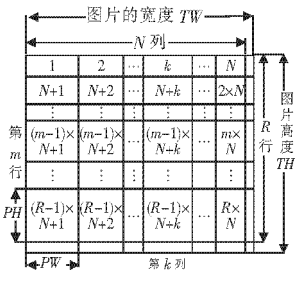
f)得到 *N* 值, 代入公式 $PW = \blacktriangledown TW/N\lrcorner$, $PH = \blacktriangledown PW/\mathrm{aspect}\lrcorner$, 得到 *PW* 和 *PH* 的值。

一旦确定了照片的宽 *PW*、高 *PH* 和图片容纳照片的列数 *N*, 就可以将同一模型所有照片按从左到右、从上到下的顺序拷贝到图片的相应区域了。

当然这些照片在图片中的位置不是随意的, 而是要与半球面上的拍摄点所在经线和纬线的编号建立某种对应关系, 为了建立对应关系。要对所有的拍摄点和照片建立索引, 如图 6所示。



(a) 拍摄点索引图(从极点俯视)



(b) 照片在图片中的索引图

图 6　拍摄点的索引图和照片在图片中的索引图

从图 6(a)中可以看到, 对于一个有 *Q* 条纬线和 *T* 条经线的半球来说, 拍摄点的索引取值是 [1, $Q \times T + 1$], 索引为 1的拍摄点是指在极点处的拍摄点。在第 *k* ($1 \leqslant k \leqslant Q$)条纬线和第 *n* ($1 \leqslant n \leqslant T$)条经线交会处的索引 $P_{\mathrm{index}} = (k-1) \times T + n + 1$。在图 6(b)中照片的数量 *n*Count与拍摄点的数量是相同的, 也就是说 $Q \times T + 1 = N \times R = n\mathrm{Count}$, 其中 $R = \blacktriangledown TH/PH\lrcorner$。拍摄点的索引值与照片在图片中的索引值是一一对应的, 如果拍摄点 *p* 的索引值是 P_{index}, 则将在拍摄点 *p* 拍摄的照片放在索引值为 P_{index}图片的位置。根据 P_{index}反算出该照片在图片中的行号 *h* 和列号 *l* 其计算公式如下:

$$h = \blacktriangledown (P_{\mathrm{index}} - 1)/N\lrcorner$$
$$l = (P_{\mathrm{index}} - 1)\%N + 1$$

其中, % 表示取余数, 将 $P_{\mathrm{index}} = (k-1) \times T + n + 1$代入得:

$$\begin{cases} h = \blacktriangledown [(k-1) \times T + n]/N\lrcorner + 1 \\ l = [(k-1) \times T + n]\%N + 1 \end{cases} \quad (4)$$

式 (4)反映出拍摄点所在的经线编号 *n* 和纬线编号 *k* 与所拍照片在图片中的行号 *h* 和列号 *l* 的对应关系。

3　Billboard的生成、旋转平移策略和贴图策略

3.1　Billboard的生成

Billboard的宽和高与正射投影参数中的宽 *w* 和高 *h* 保持一致, 否则将会出现拉伸变形。为了计算方便, 将 Billboard四边形的中心点设置为坐标原点, 四边形的法向量是 *z* 轴的正方向。此外, 还有四边形的纹理坐标信息。由于 $TW \geqslant N \times PW$, $TH \geqslant R \times PH$, 在通常情况下, $TW = N \times PW$, $TH = R \times PH$ 是无法满足的, 纹理坐标 *u* 的取值为 [0, $(N \times PW)/TW$]; *v* 的取值为 [0, $(R \times PH)/TH$]。四边形四个顶点的纹理坐标如图 7所示。

由于 Billboard的旋转和贴图都是由 GPU 完成的, 必须把一些必要的参数传输给 GPU, 这些参数包括照片的宽高比 aspect(float型)、模型的包围球中心 center(三个浮点型, 分别代表 *x*、*y*、*z* 分量)、拍摄点所在半球经线的数目 *T* (int型)、纬线的数目 *Q* (int型)、图片中照片的列数 *N* (int型)、纹理坐标 *u* 方向上的最大取值 $u\mathrm{Max} = (N \times PW)/TW$ (float型)、纹理坐标 *v* 方向上的最大值 $v\mathrm{Max} = (R \times PH)/TH$ (float型)。因为 Billboard不需要法向量、纹理坐标 1和纹理坐标 2的信息, 所以可以把这些参数写入到模型的法向量、纹理坐标 1和纹理坐标 2

的信息,如表 1所示。GPU要用到这些信息的时候,可以直接访问法向量寄存器(Normal)、纹理坐标 1寄存器(TexCoord1)和纹理坐标 2寄存器(TexCoord2)。此外,场景当前的视点坐标 E可由相机矩阵(worldview matrix)算出。

表 1 GPU程序所需参数存放表

存放位置	x分量	y分量	z分量
法向量	center x	center y	center z
纹理坐标 1	aspect	N	uMax
纹理坐标 2	T	Q	vMax

3.2 GPU下的旋转平移策略

传统的 Billboard旋转结果是 Billboard的正向正对着视点。为了最大限度地保持物体的外观,本文采用的 Billboard旋转策略并不严格遵守 Billboard的正向始终正对着视点这一原则,而是将 Billboard的正向正对着与 OE 最接近的拍摄点 P,如图 8所示。这样做的目的是为了减少 Billboard在旋转过程中由于切换照片而造成的视觉抖动。

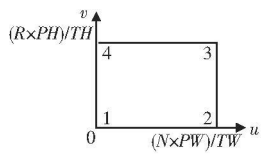


图 7 Billboard四边形四个点的纹理坐标值

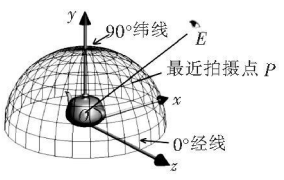


图 8 距模型视线方向 OE 最近的拍摄点 P

图 8中,O点是模型的中心点,其坐标值为 center,E点为当前的视点。为确定最近拍摄点的位置,先确定距 OE 最近的纬线 Lat的编号和经线 Lon的编号。

在确定纬线的编号时,如图 9所示。其中 OE′是 OE 在平面 xoz上的投影,α 表示 OE 与平面 xoz的夹角,特别地,当 OE 的 y分量为负值时,将 α赋值为 0。第 $k(0\leqslant k\leqslant Q)$条纬线与平面 xoz夹角 β可以通过$\frac{(Q-k)\pi}{2Q}$计算出,如果 OE 距离第 k条纬线最近且 OE 的 y分量不为负值,则 α必定在 $[\beta-\pi/(4Q),\ \beta+\pi/(4Q)]$内,即图 9中两虚线之间部分。由以上分析可得 Lat

$$\alpha=\arccos(\mathrm{nor}(OE)\times\mathrm{nor}(OE'))$$

$$\mathrm{Lat}=Q-\left\lfloor(\alpha-\pi/(4Q))/(\pi/(2Q))+1\right\rfloor$$

化简得到

$$\mathrm{Lat}=Q-\left\lfloor(4Q\alpha+\pi)/(2\pi)\right\rfloor \tag{5}$$

编号为 Lat的纬线的纬度 LatD为

$$\mathrm{LatD}=\left\lfloor(4Q\alpha+\pi)/(2\pi)\right\rfloor\times(\pi/(2Q)) \tag{6}$$

nor(OE)是 OE 向量的单位向量,nor(OE′)是向量 OE′的单位向量。

在确定经线的编号时,如图 10所示。其中 OE′是 OE 在平面 xoz上的投影,θ表示 OE′与 z轴的夹角,γ表示第 k条经线与 z轴的夹角。如果 OE 离第 k条经线最近,则必定有 OE′离第 k条经线最近,则 θ必定在 $[\gamma-\pi/T,\ \gamma+\pi/T]$之内,其中 $\gamma=(k-1)\times2\pi/T$。由以上分析可得 Lon

$$\theta=\begin{cases}\arccos(\mathrm{nor}(OE')\times z) & (z\times OE')\times y\geqslant 0\\ 2\pi-\arccos(\mathrm{nor}(OE')\times z) & (z\times OE')\times y<0\end{cases}$$

$$\mathrm{Lon}=\left\lfloor\frac{(\theta+\pi/T)-\left\lfloor(\theta+\pi/T)/(2\pi)\right\rfloor\times2\pi}{2\pi/T}\right\rfloor+1$$

上式分子 $(\theta+\pi/T)-\left\lfloor(\theta+\pi/T)/(2\pi)\right\rfloor\times2\pi$是为了防止 $(\theta+\pi/T)>2\pi$。上式化简得

$$\mathrm{Lon}=\left\lfloor(T\theta+\pi)/(2\pi)-T\times\left\lfloor\frac{\theta+\pi/T}{2\pi}\right\rfloor\right\rfloor+1$$

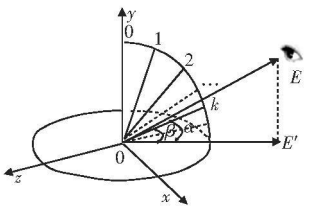


图 9 确定纬线的编号 Lat

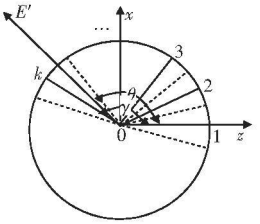


图 10 确定经线的编号 Lon

编号为 Lon的经线的经度 LonD为

$$\mathrm{LonD}=\left\lfloor(T\theta+\pi)/2\pi-T\times\left\lfloor\frac{\theta+\pi/T}{2\pi}\right\rfloor\right\rfloor\times2\pi/T$$

特别地,当 Lat=Q时,即拍摄点在极点处,此时

$$\mathrm{Lon}=1,\quad \mathrm{LonD}=0$$

综上所述可得:

$$\mathrm{Lon}=\begin{cases}1 & \mathrm{Lat}=Q\\ \left\lfloor(T\theta+\pi)/(2\pi)-T\times\left\lfloor\frac{\theta+\pi/T}{2\pi}\right\rfloor\right\rfloor+1 & \mathrm{Lat}<Q\end{cases} \tag{7}$$

$$\mathrm{LonD}=\begin{cases}0 & \mathrm{Lat}=Q\\ \left\lfloor(T\theta+\pi)/(2\pi)-T\times\left\lfloor\frac{\theta+\pi/T}{2\pi}\right\rfloor\right\rfloor\times2\pi/T & \mathrm{Lat}<Q\end{cases} \tag{8}$$

根据式(6)和(8)计算出的结果可以得到一个从中心点 O到最近拍摄点 p的单位向量 V:

$$V=(\sin(\mathrm{LonD})\times\cos(\mathrm{LatD}),\ \sin(\mathrm{LatD}),\ \cos(\mathrm{LonD})\cos(\mathrm{LatD}))$$

因为 Billboard的初始朝向为 z轴正方向,而 Billboard的最终朝向为 V,所以只需要构建一个从 z轴到 V 的旋转矩阵即可。旋转矩阵的构造分两步:

a)绕 x正向轴顺时针 LatD(弧度):

$$\mathrm{MatX}=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos(\mathrm{LatD}) & -\sin(\mathrm{LatD})\\ 0 & \sin(\mathrm{LatD}) & \cos(\mathrm{LatD})\end{bmatrix}$$

b)绕 y正向轴逆时针旋转 LonD(弧度):

$$\mathrm{MatY}=\begin{bmatrix}\cos(\mathrm{LonD}) & 0 & -\sin(\mathrm{LonD})\\ 0 & 1 & 0\\ \sin(\mathrm{LonD}) & 0 & \cos(\mathrm{LonD})\end{bmatrix}$$

$$\mathrm{RoMat}=\mathrm{MatX}\times\mathrm{MatY}=$$

$$\begin{bmatrix}\cos(\mathrm{LonD}) & 0 & -\sin(\mathrm{LonD})\\ -\sin(\mathrm{LatD})\times\sin(\mathrm{LonD}) & \cos(\mathrm{LatD}) & -\sin(\mathrm{LatD})\times\cos(\mathrm{LonD})\\ \cos(\mathrm{LatD})\times\sin(\mathrm{LonD}) & \sin(\mathrm{LatD}) & \cos(\mathrm{LatD})\times\cos(\mathrm{LonD})\end{bmatrix}$$

最后还要将 Billboard平移到 O点,O点的坐标为 center,其平移矩阵为

$$\mathrm{tranMat}=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ \mathrm{center}.x & \mathrm{center}.y & \mathrm{center}.z & 1\end{bmatrix}$$

所以 Billboard的最终旋转平移矩阵为

$$\mathrm{RoTranMat}=\begin{bmatrix}\cos(\mathrm{LonD}) & 0 & -\sin(\mathrm{LonD}) & 0\\ -\sin(\mathrm{LatD})\times\sin(\mathrm{LonD}) & \cos(\mathrm{LatD}) & -\sin(\mathrm{LatD})\times\cos(\mathrm{LonD}) & 0\\ \cos(\mathrm{LaD})\times\sin(\mathrm{LonD}) & \sin(\mathrm{LatD}) & \cos(\mathrm{LatD})\times\cos(\mathrm{LonD}) & 0\\ \mathrm{center}.x & \mathrm{center}.y & \mathrm{center}.z & 1\end{bmatrix} \tag{9}$$

3.3 GPU下的贴图策略

从图 6可以看出,当前 Billboard显示内容是 R行 N列照片,如何从 R行 N列照片中选取合适的照片是贴图策略的关键。现在已经计算出距模型视线方向 OE 最近的拍摄点的纬度编号 Lat和精度编号 Lon,根据式(4)可以计算出相应照片在图片中的行号 H和列号 L。

现在只需要将 Billboard中的纹理坐标值修改为行号为 H、

列号为 L的照片所对应的纹理坐标值就可以了, 如图 11所示。

要将当前的纹理坐标约束到图 11中的小方格子里, 可以分为两步:

a)对当前的纹理坐标缩放。

由于当前的纹理坐标范围为

$$0 \leqslant u \leqslant u\mathrm{Max},\quad 0 \leqslant v \leqslant v\mathrm{Max}$$

而缩放后的纹理坐标应当为

$$0 \leqslant u \leqslant w,\quad 0 \leqslant v \leqslant h$$

其中: $w = u\mathrm{Max}/N$, $h = v\mathrm{Max}/(N \times \mathrm{aspect})$, aspect是照片的宽高比。利用缩放矩阵 scaleMat就可以完成缩放:

$$\mathrm{scaleMat} = \begin{bmatrix} 1/N & 0 \\ 0 & 1/(N \times \mathrm{aspect}) \end{bmatrix}$$

b)将纹理坐标平移到与小方格重合。

只需要将缩放后的纹理坐标加上照片左上点 1的纹理坐标即可。左上点 1的纹理坐标可以通过

$$\begin{cases} u_1 = (L-1) \times w \\ v_1 = (H-1) \times h \end{cases}$$

把 w和 h代入, 最终的平移缩放矩阵为

$$\mathrm{scaTran} = \begin{bmatrix} \frac{1}{N} & 0 & 0 \\ 0 & \frac{1}{(N \times \mathrm{aspect})} & 0 \\ \frac{(L-1) \times u\mathrm{Max}}{N} & \frac{(H-1) \times v\mathrm{Max}}{(N \times \mathrm{aspect})} & 1 \end{bmatrix} \quad (10)$$

最后, 在 GPU pixel shader程序中根据 alpha值用 clip函数将背景色滤掉, 如图 12所示。

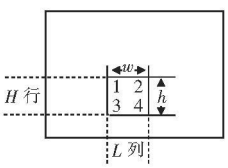
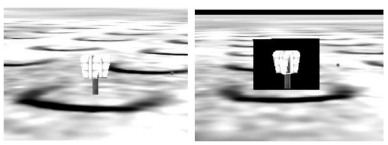


图 11　确定第 H行 L列照片的纹理坐标值



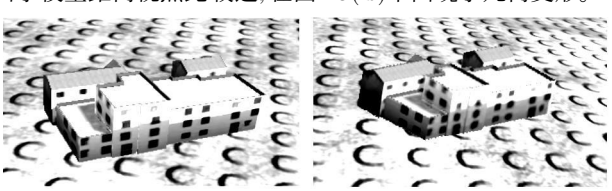
(a) 滤掉背景色后的 Billboard　(b) 未滤掉背景色的 Billboard

图 12 滤掉背景色和未滤掉背景色的 Billboard

4　实验结果分析

4.1　视觉效果比较

第一组实验是原始模型与用本方法生成的粗精度模型的视觉比较, 如图 13(a)和(b)所示。其中, 原始模型由 256个三角形组成, 纹理量共计 2.06 MB(包括 2张屋顶照片, 5张墙壁的照片)。粗精度模型只有 2个三角形, 纹理量共计 684 KB, 由于模型距离视点比较近, 在图 13(b)中出现了几何变形。



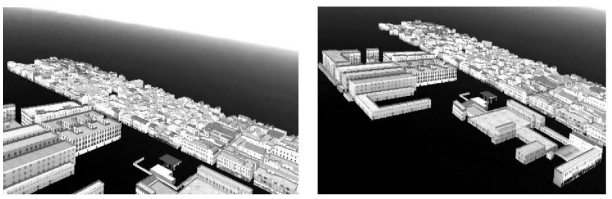
(a) 原始模型　(b) 用本文方法生成的粗精度模型, TW= TH=2048, Lat=4, Lon=20, aspect=1

图 13　原始模型与用本文方法生成的粗精度模型的比较

第二组实验是对比全部用原始模型的场景与近处用原始模型、远处用本文方法生成的粗精度模型的场景之间的差异, 如图 14所示。

4.2　运行效率比较

实验所用计算机配置 Pentium　Dual-Core CPU E5300@ 2.60 GHz@ 2.62 GHz内存 2.0 GB; 显示卡 NVIDIA GeForce GTS 250; 操作系统 MicrosoftWindows XP Professional版本 2002 Service Pack 3; Direct版本为 DirecX 9.0c。编译环境为 Microsoft Visual Studio 2005 Service Pack 1+ OGRE 1.7 SDK。实验效率如表 2所示。



(a) 全部用原始模型的场景　(b) 近处用原始模型、远处用本文方法生成的模型

图 14　大规模场景效果的比较

表 2　运行效率对照表

名称	原始模型	边折叠算法			粗精度模型(本文方法)
		简化到 20%	简化到 50%	简化到 80%	
三角形数量/个	60 000 628	11 860 002	29 920 002	47 980 002	20 000
平均帧率/fps	0.8	4.0	1.5	0.9	18.6
最好帧率/fps	1.3	5.8	2.7	1.6	31.5
最坏帧率/fps	0.6	3.9	0.8	0.5	18.0
模型数量/个	10 000	10 000	10 000	10 000	10 000

5　结束语

本文提出了一种粗精度三维模型的生成与快速显示方法, 从不同角度对原始模型拍照, 根据当前视点与模型的关系确定 Billboard朝向和显示的内容, 并详细介绍了照片的拍摄、存放, Billboard的生成、GPU下的旋转平移策略和 GPU下的贴图策略。其目的就是把对三角网的简化问题转换为图片的显示问题, 但是由于产生大量图片, 对图片的处理(如图片压缩)成为制约显示效率的关键问题。另外, 由于用本方法生成的模型放在距离视点近处会产生几何变形, 在具体实践过程中, 不宜将这些模型的 LOD距离设置过小, 过小不但产生几何变形, 也会导致这些模型与相邻的层次模型之间不能形成光滑的视觉过渡, 进而严重影响到场景的视觉效果。建议将本算法与引言中所提及的简化算法结合在一起, 形成多层次(多于 2个)的或动态的 LOD模型, 这样做不但可以提高系统的运行效率, 而且也可以形成更为光滑的视觉过渡。

参考文献:

[1] WU Jian-hua, KOBBELT L. A stream algorithm for the decimation of massive meshes[C]//Proc of Graphics Interface Conference. 2003: 185-192

[2] SHAFFER E, GARLAND M. Efficient adaptive simplification of massive meshes[C]//Proc of IEEE Visualization Conference. Washington DC: IEEE Computer Society, 2001: 127-134

[3] GARLAND M, SHAFFER E. A multiphase approach to efficient surface simplification[C]//Proc of IEEE Visualization Conference. Washington DC: IEEE Computer Society, 2002: 117-124.

[4] HOPPE H. Smooth view-dependent level-of-detail control and its application to terrain rendering[C]//Proc of IEEE Visualization Conference. Washington DC: IEEE Computer Society, 1998: 35-42.

[5] DeCORO C, TATARCHUK N. Real-time mesh simplification using the GPU[C]//Proc of the Symposium on Interactive 3D Graphics and Games. 2007: 161-166

[6] 费广正, 蔡康颖, 吴恩华. 基于细节迁移的快速外存模型简化方法[J]. 软件学报, 2002, 12(11): 1630-1638　(下转第 2388页)

个状态。而状态初始概率中 $\pi(s) = 1$, 其余初始状态概率为 0。

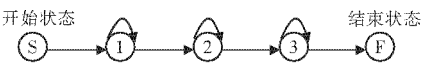


图 8　系统采用的隐性马尔可夫模型拓扑结构

本文中对模型的训练采用了传统的 Baum-Welch算法, 通过一系列离散的观测序列循环训练 HMM 的参数。模型验证和概率计算通过传统的前 /后向算法进行。

建立模型的另一主要问题在于状态数量选择。在理想情况下, 一个左向箭头的表示可能只需要两个隐藏状态 S1、S2。S1表示直线的特征, S2 表示箭头特征。考虑由于用户的手绘风格以及特征提取阶段可能出现的误差, 两个状态的 HMM 模型并不能够提供良好的分类结果, 系统通过细化状态数量的方式来建立可以良好分辨手势的模型。经过实验, 在 4~ 6个状态下, 模型具有较好的识别能力。图 9给出了矩形在不同状态数目下的识别率, 可以看出在 5个隐藏状态下, 模型具有最好的识别效果。

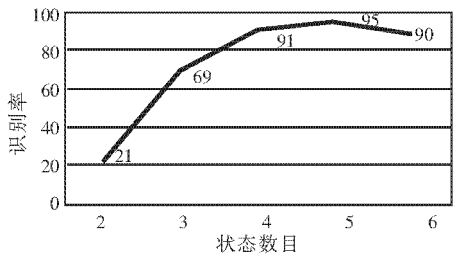
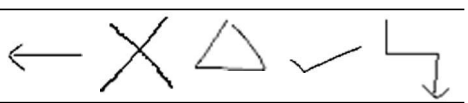


图 9　不同状态数下矩形的识别率

3　实验结果

利用本文所述方法, 系统分别用鼠标模拟笔画输入和手写板输入测试了手势识别率。在训练阶段, 系统采集了 12个用户书写的 25种手势, 每种手势同一用户书写 3次, 使用这 36个训练数据对识别器进行循环训练。在识别效果测试中, 测试用户数量为 30人, 且均没有提前训练测试用户的绘制风格, 每人随机分配 6种手势, 每种手势绘制 4次。表 1给出了部分手势的最终测试结果以及相应的识别率。

表 1　手势符号的识别率 /%

手势符号					
鼠标输入	94	95	91	94	92
手写板输入	95	96	94	97	95
Derk[8]HMM	90	91	91	92	88

从表 1中可以看到, 由于鼠标在输入笔画时容易产生抖动、漂移, 不如手写板输入手势流畅, 造成了在个别手势特别是涉及曲线的手势上识别率稍低; 但是由于在手势的预处理部分去除了冗余点并优化了特征的提取, 平均识别率仍然能在 93%以上。采用手写板输入的成绩要更好, 识别率在现阶段平均达到了 95%, 部分手势达到了 97%。与基于传统的特征点采集算法和特征提取算法的 HMM 建模相比较 (Derk的 HMM), 本系统识别率有了明显的提高, 尤其是在手势符号较为复杂的情况下。鉴于基于 SVM、ANN 的机器学习方法版本众多、实现方法各异以及测试时所采用的手势模型不同, 系统无法与这些机器学习方法进行直观比较。与其他机器学习方法相比, 虽然其他方法也能够提供 90%以上的识别率, 基本满足手势识别的应用需求, 但 HMM 自身特性决定了当有新手势加入时不需要对整个系统模型的现有手势进行重新学习, 从而减小了系统的重构负担, 这是 HMM 学习方法的优势。

4　结束语

本文建立了一个基于隐性马尔可夫模型的手势识别框架, 并在重采样和特征提取阶段提出了优化方法。实验结果表明, 本系统在用户手势识别的准确性方面相较于未优化前有了一定提高。下一步的工作目标是将手势识别结合到基于手绘的三维建模的原型系统上, 为建模系统提供手势功能支持, 并在实际操作中体现出手势交互相较于 WIMP交互方式的优越性。系统在手势设计的合理性以及对用户语义的理解上仍有较大的提升空间, 该部分工作会在实践中继续完善。

参考文献:

[1] 孙正兴, 冯桂焕, 周若鸿. 基于草图的人机交互技术研究进展[J]. 计算机辅助设计与图形学学报, 2005, 17(9): 1889-1899.

[2] RUBINE D. Specifying gestures by example[J]. Computer Graphics, 1991, 21(4): 329-337.

[3] MARK W. Design and implementation of a stroke interface library [C]//IEEE Region 4 Student Paper Contest, 1997.

[4] SUN Zheng-xing, LIU Wen-yin, PENG Bin-bin, *et al*. User adaptation for online sketchy shape recognition[C]//Lecture Notes in Computer Science. Berlin: Springer-Verlag, 2004: 303-314.

[5] GOLUBITSKY O, WATT S. Online recognition of multi-stroke symbols with orthogonal series[C]//Proc of the 10th International Conference on Document Analysis and Recognition. 2009: 1265-1269.

[6] WILLEMS D. Iconic and multi-stroke gesture recognition[J]. Pattern Recognition, 2009, 42(12): 3303-3312.

[7] LI X, YEUNG D. On-line handwritten alphanumeric character recognition using dominant points in strokes[J]. Pattern Recognition, 1997, 30(1): 31-44.

[8] DERK A, CRAIG B. Hidden Markov model symbol recognition for sketch-based interfaces[C]//Proc of AAAI Fall Symposium. 2004: 15-21.

(上接第 2385页)

[7] 李蔚清, 洪云轩, 吴惠中. 一种基于特征的实时 LOD模型生成算法[J]. 系统仿真学报, 2005, 17(2): 429-431.

[8] 刘晓东, 熊海桥, 蒋立华, 等. 利用 Billboard实现虚拟植物集群生长显示[J]. 计算机工程, 2003, 29(13): 52-54.

[9] 黄炳, 陈俊丽, 万旺根. 飞行仿真中三维云场景的渲染[J]. 上海大学学报: 自然科学版, 2009, 15(4): 342-345.

[10] 梁伟, 刘群, 吴渝. 飞机尾翼空中失火的场景模拟[J]. 计算机工程与应用, 2010, 46(9): 163-165.

[11] 肖永辉, 徐青, 周杨. 三维城市景观阴影绘制算法的研究与实现[J]. 测绘科学技术学报, 2006, 23(4): 308-309.

[12] 张楠, 王建立, 王鸣浩. 基于图形处理器的边缘检测算法[J]. 计算机科学, 2010, 37(1): 265-267.

[13] 柳有权, 刘学慧, 吴恩华. 基于 GPU带有复杂边界的三维实时流体模拟[J]. 软件学报, 2006, 17(3): 568-576.

[14] HOPF M, ERTL T. Hardware based wavelet transformations[C]//Proc of Vision, Modelling and Visualization Conference 1999: 317-328.

[15] 周国亮, 冯海军, 何国明, 等. 图形处理器在数据管理领域的应用研究综述[J]. 计算机科学与探索, 2010, 4(4): 289-303.