基于几何和图像的混合式图形实时绘制算法*

郑文庭 鲍虎军 彭群生

(浙江大学 CAD&CG 国家重点实验室,杭州 310027)

Hanqiu Sun

(Department of Computer Science & Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong, China)

摘要 将基于几何和图像的图形绘制算法有机地结合起来,通过逆向 Warping 技术来实现两预测图像间的光滑过渡. 引入了深度值的非线性变换和二阶差分运算,从而实现了由深度图像到三维局部场景的快速层次重建. 利用所得到的三维几何模型及纹理映射技术,实现了中间过渡画面的实时生成. 实验结果表明,算法对场景复杂性的依赖程度较低,能在中低档图形工作站上实现复杂场景的实时绘制.

关键词 虚拟现实 纹理映射 消隐 层次细节模型

最近几年,复杂场景的实时绘制技术得到了极大的发展,出现了 3 类高效的技术,即快速可见性计算技术(Visibility Computation)[1~5]、层次细节模型技术(Level of Detail)[6,7]和基于图像的图形绘制技术(Image Based Rendering, 简称为 IBR)[8~12]. 特别是后者,由于它是一种与场景复杂性无关的图形绘制技术,因而用户可在普通工作站和个人计算机上实现复杂场景的实时漫游. 一般地,基于传统 IBR 技术的算法只能处理静态场景,且漫游时难以生成连续的视图变化. Eric Chen 的视图插值算法[8]首次实现了相邻深度图像之间的光滑过渡. 由于采用正向映射方法,该算法无法精确模拟一般透视变换,且在过渡画面上会产生大量由于画面扩张和可见性变化而形成的空洞,导致算法效率的严重降低. 这一缺陷严重制约了视图插值算法的广泛应用. 近来,一些研究者转而采用带纹理的简单几何(如四边形、多边形网格等)来逼近复杂景物几何[13~15],并直接绘制这些简单几何来生成连续画面. 由于简单几何具有与原始景物相近的深度值,因而这类算法能生成正确的前后遮挡关系,且能在一定的视点区域内保持所需的绘制精度. 但当不满足绘制精度时,算法需基于原几何自动生成新的纹理及其简化几何.

为克服单纯基于几何或图像的图形绘制技术的缺陷,本文给出了一种基于几何和图像的混合式实时图形绘制算法. 算法首先采用基于几何的快速图形绘制算法来生成预测采样深度图像,然后采用逆向图像 Warping 技术来生成中间过渡画面.

1 三维场景局部重建

为精确反映透视变换,在画面过渡过程中,必须考虑画面中各个象素可见点在三维空

1998-09-21 收稿,1999-01-25 收修改稿

* 霍英东青年教师基金和国家自然科学基金(批准号: 69673027)资助项目

间中的位置．本文根据给定的深度值逐帧建立局部三维场景的近似表示而无需考虑其相邻画面的对应关系．

1.1　由深度信息重建局部场景

假设 $I_{M\times N}$为给定图像(分辨率为 $M\times N$)，$z_{M\times N}$为其深度帧缓存，$\boldsymbol{EUVW}$ 为生成 I 的摄像机坐标系，其中 $\boldsymbol{E}(E_x, E_y, E_z)$为视点位置(如图 1 所示)，3 个单位坐标向量关于世界坐标系 $Oxyz$ 分别表示为 $\boldsymbol{U}(U_x, U_y, U_z)$、$\boldsymbol{V}(V_x, V_y, V_z)$和 $\boldsymbol{W}(W_x, W_y, W_z)$，则对任一象素 p，它所对应的三维可见点 P 为

$$P = H_E p, \tag{1}$$

其中 P 和 p 均为齐次坐标，即 $P(x_P, y_P, z_P, 1)$和 $p(x, y, 1, 1)$，H_E 为视点 E 处的 4×4 摄像机透视投影变换矩阵：

$$H_E = \begin{pmatrix} zU_x & 0 & 0 & E_x + zyV_x + zW_x \\ 0 & zV_y & 0 & E_y + zxU_y + zW_y \\ 0 & 0 & z & E_z + zxU_z + zyV_z \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

这里，$x = \left(\frac{i}{M} - \frac{1}{2}\right)\mathrm{tg}\frac{\alpha}{2}$，$y = \left(-\frac{j}{M} + \frac{N}{2M}\right)\mathrm{tg}\frac{\alpha}{2}$，$\alpha$ 为摄像机的水平广角，(i,j)为 p 点的屏幕坐标，$z = z(i,j)$．这样，图像 I 的任 4 个相邻象素 $p(i,j)$，$p(i+1, j)$，$p(i+1, j+1)$和 $p(i, j+1)$所对应的可见点 $P(i,j)$，$P(i+1, j)$，$P(i+1, j+1)$和 $P(i, j+1)$形成了一空间四边形．图像 I 上所有 4 个相邻象素所对应的可见点就形成了一张空间四边形网格，而图像 I 就是这一网格在当前视平面上的投影．因此，这样得到的空间四边形网格可以看作是原场景几何在当前视域中的一种近似．这样，当视点在当前视点附近移动时，我们可近似地采用这一近似网格几何模型来绘制画面．且可直接将 I 的颜色值映射到新的画面上去．

显然，所得到的空间网格共有 $M\times N$ 个空间四边形．若逐个绘制(每一网格需剖分成二个三角形)这些多边形，显然无法在普通硬件平台上实现实时绘制．一个自然的想法是将原属于同一多边形的各网格合并成一个网格，以减少需绘制的多边形个数．而多边形内部的细节则采用图像 I 作为纹理来恢复．

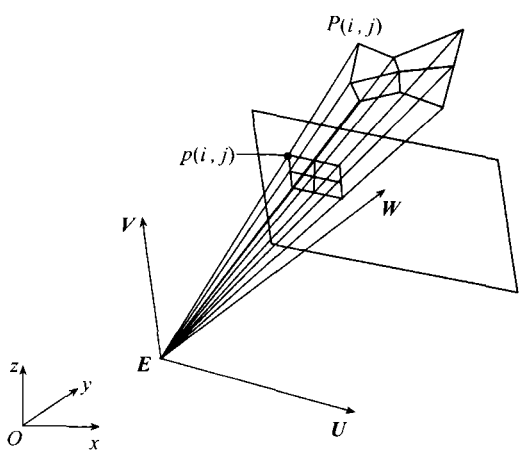


图 1　由深度信息恢复三维几何网格

1.2　局部场景的层次细节模型

一般地，若采用硬件 Z-buffer 算法来绘制场景，则我们可方便地得到二个帧缓存，一个为存贮当前视点和视线方向图像的颜色帧缓存，另一个为存贮该图像中每一象素可见点在当前摄像机坐标系中深度的深度帧缓存．硬件 Z-buffer 算法实际上作了一非线性变换，深度缓存中象素(i,j)处的深度值 $z'(i,j)$取为

$$z'(i,j) = 2^{24}\frac{z_{\mathrm{far}}}{z_{\mathrm{far}} - z_{\mathrm{near}}}\left(1 - \frac{z_{\mathrm{near}}}{z(i,j)}\right), \tag{2}$$

其中 $z(i,j)$为象素(i,j)所对应的可见点在摄像机坐标系中的 z 坐标，$z = z_{near}$和 $z = z_{far}$分别为当前视域的近距离平面和远距离平面. 值得注意的是，经过上述变换后，来自同一平面上的可见点的深度值 $z'(i,j)$仍为(i,j)的线性函数. 因此，若与(i,j)相邻的象素可见点来自于同一平面多边形，则深度值 $z'(i,j)$满足：

$$\Delta^2 z'(i,j) = 0,$$

其中 Δ^2 为二阶差分算子，其定义如下：

$$\Delta^2 z'(i,j) = 4z'(i,j) - z'(i,j-1) - z'(i,j+1) - z'(i-1,j) - z'(i+1,j).$$

对深度帧缓存器中的象素逐一做二阶差分就得到了一新的帧缓存(称为 Δ 缓存)，其中含有大量的取值为零的象素，由其组成的连通象素区域必由一空间平面多边形投影而成. 由此，我们就可恢复出该空间多边形位置.

对于不是由硬件 Z-buffer 绘制的画面，如用深度摄像机(range camera)拍摄的画面或用其他方式生成的画面，可应用式(2)对画面中每一象素可见点在摄像机坐标系的 z 值作上述变换，再重构其深度缓存器及 Δ 缓存器.

为满足实时绘制的需要，我们采用二叉剖分技术来实现区域的剖分，进而实现前述空间多边形网格的自动简化. 本质上，剖分所产生的二叉树是对当前视域内原始几何场景的一个层次细节逼近. 算法交替地沿水平、垂直方向剖分区域，其剖分位置则按以下规则确定：首先从 Δ 缓存器当前区域位于剖分方向任一侧的边界开始，逐行扫描该区域内所含象素. 若位于当前扫描线上的所有象素的 $\Delta^2 z'(i,j)$均小于给定阈值，则继续扫描下一行，直至当前扫描线上某一象素的 $\Delta^2 z'(i,j)$超出给定阈值为止. 若已扫描区域足够大，则取前一扫描线为剖分位置；否则从另一侧边界开始做类似扫描操作. 若上述二步不能确定剖分位置，则简单地选取中间扫描线作为剖分位置. 算法递归地执行这一步骤，直至所生成的子区域中所有象素可见点的 $\Delta^2 z'(i,j)$均小于给定阈值或子区域无法继续剖分为止. 图版 I-A 给出了一深度图像在不同阈值下的网格剖分及三维几何的重建结果. 由长方形网格 4 个顶点处的深度值就可恢复出其空间多边形的方位. 注意，当长方形网格退化为一个象素时，其 4 个角点处的深度值取为同一值，即其中心处的深度. 显然，上述二叉剖分的终判条件决定了剖分产生的网格的多少，即决定了恢复场量的细致程度. 因而，当终判条件由精致变为粗糙时，我们可得到当前局部场景的一个层次细节模型.

值得指出的是，本算法生成的网格在三维空间中并不保持其拓扑的连续性，尽管它在当前图像平面上是连续的. 为避免出现裂缝，所有生成网格的边界均向外扩张半个象素宽. 这样处理的优点是避免了后续 Warping 过程的纹理变形，保证了绘制的精度，但它亦将在新的视平面上产生空洞区域.

2 图像绘制

我们讨论当视点偏离当前视点后，画面的快速生成算法. 与传统方法不同，我们采用图像 Warping 来生成中间过渡画面.

2.1 逆向 Warping

上一节重建的三维网格在当前视域附近较好地逼近了原场景几何，当视点 E′位于当前视点 E 附近，且两者的视线方向相近时，我们可通过绘制该三维网格来近似得到所需画面. 为

避免光亮度的突变，我们将图像 I 作为纹理附加在三维网格上，并利用硬件纹理映射来逐一绘制每一空间四边形网格．算法先裁取当前网格在屏幕上投影区域所对应的图像并将它投影到空间四边形上，再把该空间四边形及其纹理投影到新的图像平面上来绘制该四边形．前一步其实是纹理映射的建立过程，由于这是一个非线性映射，因而我们无法采用硬件纹理映射技术来直接进行绘制．为此，我们首先将空间四边形投影到新的图像平面上得到一平面四边形，然后以原图像对应区域作为纹理图像来绘制该平面四边形，该四边形的顶点可由下式计算得到

$$q_i = H_{E'}^{-1}P_i = H_{E'}^{-1}H_E p_i \quad (i = 1, 2, 3, 4),$$

其中 q_i 为所欲绘制的平面四边形的顶点，P_i 空间四边形的顶点．

上述过程其实是一个逆向的映射过程，新的视平面上的平面四边形可看作是原图像上对应长方形的变形，因而我们称这一绘制方法为逆向 Warping 算法．如图 2 所示，空间四边形 $P_1P_2P_3P_4$ 对应于图像 I 上的象素区域 $p_1p_2p_3p_4$，四边形 $q_1q_2q_3q_4$ 则为空间四边形 $P_1P_2P_3P_4$ 在一新的邻近视平面上的投影区域，因而，我们的逆向 Warping 过程实际上是以 p_i 为纹理区域顶点绘制平面纹理四边形 $q_1q_2q_3q_4$ 的过程．

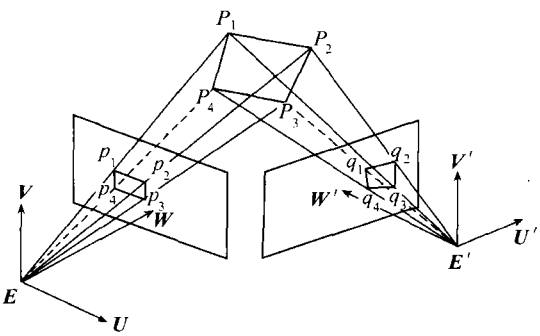


图 2　逆向 Warping 过程

注意，当恢复出空间四边形面积较大，且离视点较近时，利用上述逆向 Warping 技术可能会产生较大的纹理漂移，从而导致严重的视觉误差，这是由于所得到的空间四边形具有较大的深度变化．可将这类四边形适当地剖分得细一些以提高绘制精度．此时，可利用立体角作为度量标准来自适应剖分这类四边形．目前，本算法简单地采用均匀剖分技术．

2.2　空洞填补

与正向 Warping 技术相比，逆向 Warping 技术有效地消除了由于画面扩张而产生的大量细小空洞．但对由于新视点处的场景可见性变化而引起的空洞则仍然无法填补，其主要原因在于我们无法从原始图像 I 上获得空洞区域场景的任何信息．欲使上述逆向 Warping 算法可应用于虚拟现实系统，我们必须寻找有效的方法来填补这类空洞．而目前已有的填补空洞的方法[3,8]在计算速度上和图像质量上还不尽人意．为此，我们采用以下的策略来解决空洞填补问题：

(1) 采用两幅邻近视点的源深度图像来互补空洞，由于从两幅邻近视点的源图像重建的场景空间网格投影在同一中间视平面上所生成的画面具有互补部分空洞的特点，因而这一方法部分填补了空洞．绘制时，我们利用中间视点离两源图像对应视点的距离大小来确定叠加顺序，取距离小的源图像所生成的中间画面作为前景，另一中间画面作为背景，这样在一定程度上减少了图形走样．

(2) 对那些由于场景可见性变化，而在中间视平面引起的空洞，我们利用前一帧过渡画面作为背景，采用油画家算法直接将当前过渡画面绘制上去．这样，(1)中未获填充的空洞区域将显示前一帧画面内容，从而避免了前后帧中间画面的闪烁现象．

实时图像处理的结果表明,这一空洞填补方法快速、有效,生成图像质量大大好于视域插值算法,图4给出了中间画面上空洞的填补过程. 上述算法既可应用于计算机生成的图像,亦可应用于实拍图像间的过渡. 对于后者,我们需利用计算机视觉理论来恢复源图像的摄像机参数及其深度信息.

3 执行结果及讨论

我们在本实验室的SGI Octane工作站上实现了本算法. 源深度图像的生成采用Performer软件开发平台. 我们用MultiGen造型软件构造了一个场景来测试本算法. 测试图像的分辨率均为512×512.

该场景由100 000余个多边形构成,其中一些细节则采用纹理来描述,图版I-B为中间画面的生成过程:(a)和(b)是两幅相邻视点的深度源图像,(c)和(d)分别为依据图像(a)和(b)重建的三维场景网格投影在中间视点 E_M 的视平面上的绘制结果,其中 $E_M=(E_a+E_b)/2$ (E_a,E_b)为图像(a)和(b)的视点). 图中红色区域表示空洞区域. 图像(e)为(c)叠合在(d)上的合成结果. 从图中可以看出,(c)和(d)上的空洞区域具有互补性,因而图像(e)上的空洞区域大为减小. 本算法简单地用上一帧画面来填补所余空洞得图(f),它即为 E_M 处的最后显示画面. 从深度图像的三维网格重建开始到生成图像(f)的计算时间为0.048 s,这里没有考虑源图像的生成时间.

由这个例子不难发现,除了生成源图像外,中间过渡画面的绘制与场景复杂性的依赖程度较低,与图像分辨有关,因而本算法可根据硬件平台的性能来适当选取图像分辨率及网格剖分精度,以便使系统保持一定的刷新频率.

经测试发现,当景物离视点很近时,本算法将产生较大的纹理漂移,所生成的过渡画面走样严重. 这是由于近距离景物在相距很近的视平面上投影将产生很大的视差,在中间过渡画面上形成大范围的空洞,本算法的空洞补策略难以精确恢复这些信息所致. 为此,对计算机生成的源图像,我们可实时动态地将离当前视点很近的景物从源图像中剔除出去,然后在过渡画面的绘制过程中再以几何方式将它们绘制上去.

4 结论

本文详细介绍了基于深度图像Warping技术的实时图形绘制算法. 引入了深度的非线性变换及二阶差分算子,使得算法容易地由深度图像来重建共面空间网格. 从而极大减少了产生的四边形数目,提高了算法效率. 最后采用逆向Warping技术和有效的空洞填补策略,本算法生成了高质量的中间过渡画面. 实验结果表明,本算法能高质量地实时绘制复杂场景,且与场景几何的复杂性无关,仅与图像分辨率有关.

参 考 文 献

1 Greene N, Kass M, Miller G. Hierarchical Z-buffer visibility. Computer Graphics, 1993, 27(2): 231~238

2 Regan M, Pose R. Priority rendering with a virtual address recalculation pipeline. Computer Graphics, 1994, 28(3): 155~162

3 Zhang H S, Manocha D, Hudson T, et al. Visibility culling using hierarchical occlusion maps. Computer Graphics, 1997, 31(3): 77

~ 88

4 Teller S, Sequin C. Visibility preprocessing for interactive walkthroughs. Computer Graphics, 1991, 25(4): 61 ~ 69

5 Wang Y G, Bao H J, Peng Q S. Accelerated walkthroughs of virtual environments based on visibility preprocessing and simplification. Computer Graphics Forum, 1998, 17(3): 187 ~ 194

6 Hoppe H. Progressive meshes. Computer Graphics, 1996, 30(3): 99 ~ 108

7 Luebke D, Erikson C. View-dependent simplification of arbitrary polygonal environments. Computer Graphics, 1997, 31(3): 199 ~ 208

8 Chen S E, Williams L. View interpolation for image synthesis. Computer Graphics, 1993, 27(2): 279 ~ 288

9 Chen S E. Quich Time VR: an image-based approach to virtual environment navigation. Computer Graphics, 1995, 29(4): 29 ~ 38

10 McMillan L, Bishop G. Plenoptic modeling: an image-based rendering system. Computer Graphics, 1995, 29(4): 39 ~ 46

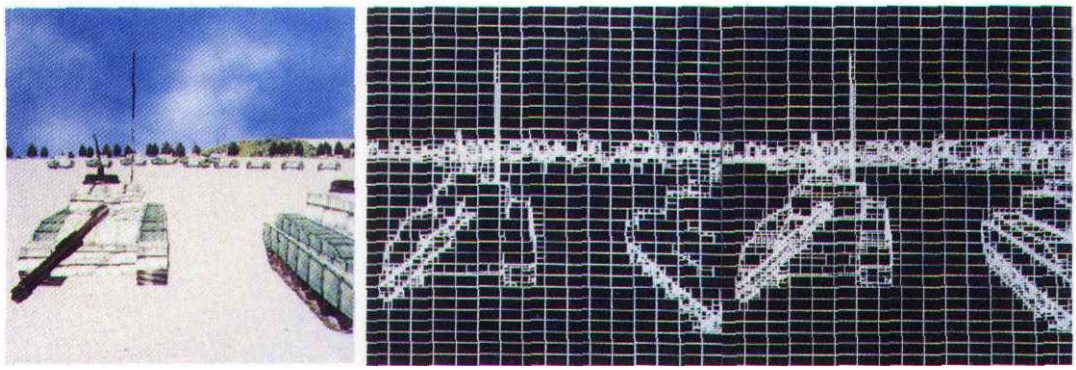
11 Levoy M, Hanrahan P. Light field rendering. Computer Graphics, 1996, 30(3): 31 ~ 42

12 Gortler S J, Grzeszczuk R, Szeliski R, et al. Lumigraph. Computer Graphics, 1996, 30(3): 43 ~ 56

13 Shade J, Lischinski D, Salesin D H, et al. Hierarchical image caching for accelerated walkthroughs of complex environments. Computer Graphics, 1996, 30(3): 75 ~ 82

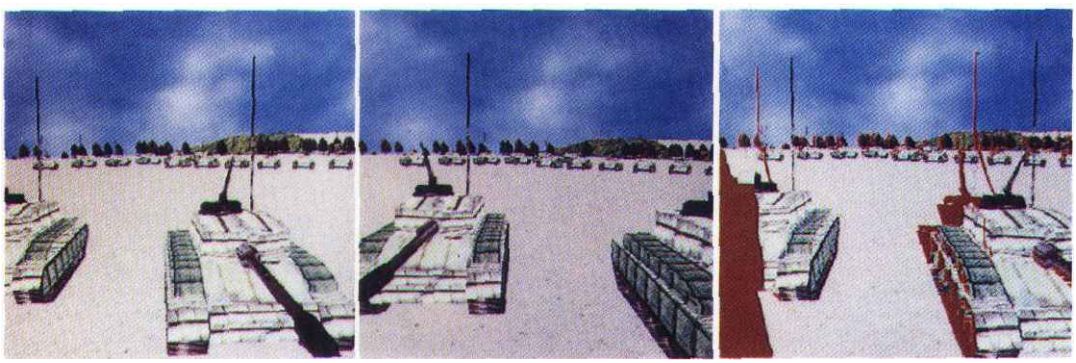
14 Sillion F X, Drettakis G, Bodelet B. Efficient imposter manipulation for real-time visualization of urban scenery. Computer Graphics, Forum, 1997, 16(3): 207 ~ 218

15 Torborg J, Kajiya J T. Talisman: commodity real-time graphics for PC. Computer Graphics, 1996, 30(3): 353 ~ 363

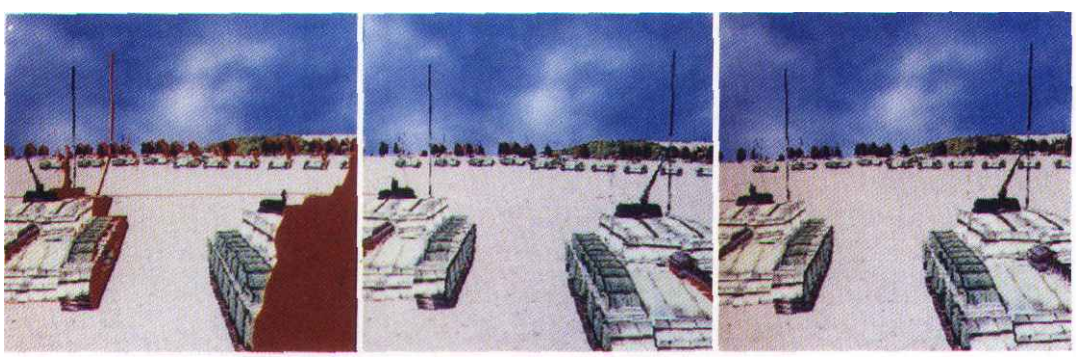


(a)　(b)　(c)

A



(a)　(b)　(c)



(d)　(e)　(f)

B

A. 深度图像的网格剖分：(a) 原始图像，(b) 阈值为 2^{15}时的剖分结果，(c) 阈值为 2^{11}时的剖分结果；B. 中间画面的生成过程