基于图象的绘制中的纹理编辑

张志强 孙济洲
(天津大学电信学院计算机系,天津 300072)

摘 要 现有的基于图象的绘制(IBR)系统中,只能提供新视点的图象,并不具有对场景编辑的功能.由于 IBR 系统的场景模型不同于传统图形学,难于直接应用传统图形学中已有的方法进行编辑.为此提出了一种修改 IBR 场景中物体的表面纹理的方法,这种方法通过在原始的参考图象上指出编辑区域并给出纹理定义,修改各个视点上的纹理.这种方法不限制编辑区域的形状、尺寸,并且考虑了亮度信息,适应了环境光照,使新纹理与原有场景更好地融合在一起.实验证明,场景的复杂程度并不影响本方法的效果,但是由于本方法不能得到场景物体准确的三维数学模型,无法用现有图形引擎进行绘制,因此无法实现硬件加速,不适用于过大的场景.

关键词 计算机图象处理(520·6040) 传统图形学 基于图象的绘制 纹理 光照 参考图象 亮度
中图法分类号: TP391.41 **文献标识码**: A **文章编号**: 1006-8961(2003)08-0912-06

Texture Editing in Image Based Rendering

ZHANG Zhi-qiang, SUN Ji-zhou
(*Department of Computer Science and Technology, Tianjin University, Tianjin* 300072)

Abstract In the existing IBR systems, they can only provide new view point of the image and have not the function of editing of scene. Because the models of the IBR systems are different from the traditional computer graphics, it is difficult to apply the methods of the traditional computer graphics directly. In this paper, a method is put forward to modify the texture of surface of scene model. This method point out the edit field in the original reference image, define the texture and can modify the texture of every view point. In addition, its advantages also include that it need not the shape, size and so on of the edit field and it considers the brightness of environment to merge the new texture and the original scene better. Proved by experiment, this method can't be limited by the complexity of scene. But because the method can't get the scene's mathematic expression, it can't be rendered by current graphics engineer and can't fit to too large scene.

Keywords Computer image processing, Traditional computer graphics, Image based rendering, Texture, Reference image, Brightness

0 引 言

为了生成具有照片真实感的图形,在传统图形学中,首先要建立场景的三维几何模型,然后对场景物体表面的材料、光照、纹理等进行处理.如果场景比较复杂,建立场景的几何模型就十分困难,甚至不可能得到场景的三维表达.并且其后的处理过程对硬件的要求比较高,绘制一幅具有照片真实感图象的时间相当长.为了解决这一问题,出现了基于图象的绘制方法(IBR).IBR 不是用三维模型来编码一个场景,而是通过一组预先获得的图象来产生场景的新视图.在大多数 IBR 方法中,处理的速度与场景的复杂度无关,这样就有效地解决了上述传统图形学的问题,能够快速地生成具有照片真实感的图象.

当前,IBR 这类基于图象的方法得到了计算机图形学界的重视,但是如何修改这种基于图象的表示并没有得到广泛的研究,而且 IBR 系统中并没有支持一些标准的编辑操作,比如合成一系列物体到一个场景中,修改场景中物体的表面纹理和材质特性,以及

基金项目:天津市自然科学基金(003600411)
收稿日期:2002-09-12;改回日期:2003-03-03

修改场景的光照情况等，然而在 IBR 系统中提供编辑功能的必要性是显而易见的，因为如果提供了编辑操作，在建筑装潢业中，建筑师就可以首先拍摄建筑物的一系列照片，然后通过 IBR 的方法恢复建筑物场景，最后用 IBR 系统的编辑操作修改场景的特性，比如墙壁的纹理和材质，在场景中添加物体等，此后建筑师可以观察编辑后的场景来检查效果.

在 IBR 系统中，实现这类编辑操作的难点在于：在 IBR 这类基于图象的方法中，并不知道场景中的各个元素的形状、大小以及表面纹理、材质特性，尤其是在不知道场景的几何描述的情况下，修改场景的结构相当困难，而上述参数在传统图形学的基于几何的方法中都是已知的，所以，在 IBR 中只有通过修改输入的参考照片来修改场景，但是，修改所有的参考照片是不现实的，因为这样会需要大量的时间和造成修改的不一致.

当前的基于 IBR 的交互编辑研究方法有全光编辑系统[1]和基于图象的交互编辑工具[2]. 在全光编辑系统中，修改一幅原始图象会反映到组成这个场景的其余参考图象中. 在一个典型的交互中，用户修改了一幅原始图象，这个变化被重新解释为一个全光函数的变量的变化，这个全光函数决定着整个场景. 系统分解全光函数为形状和光照分量，这些分量随修改的变化而更新，可以显示为一个新版本的采样图象，所以，如果有一个图象序列按顺序作为 IBR 的输入，就可以看到编辑后场景的连续的图象，但是由于此方法要求原始图象序列的连续性，因此，需要大量的参考图象. Rangaswamy 制作了一个基于 Image-Based Rendering 的交互编辑工具[2]. 这个工具的核心算法基于 McMillan 的 Warping 方法，它需要输入的数据包括参考图象、图象的深度信息、相机的位置和投影矩阵，然后采用 McMillan 的 Warping 方法将图象的像素投影到 3D 空间，这中间需要计算视差等数据；它可以在场景中添加、删除物体，修改物体和重新施加光照. 但这种方法要求图象的深度信息，在一些应用环境下，得到深度信息也是很困难的.

本文提出了一种修改 IBR 场景中物体的表面纹理的方法. 这种方法不要求原始图象序列的连续性，只需要有限角度的图象就能完成对场景的编辑，从而大大地降低了所需图象的数量，而且允许在新视点观察编辑的效果. 这种方法不要求深度信息，从而降低了计算的难度.

由于 IBR 方法的模型是通过一系列的参考照片建模而成的，因此具有真实场景中的光照特性，如果只是直接把指定纹理映射到物体的表面，那么新的纹理会与模型的其他部分不搭配，而且还掩盖了物体表面细节，严重地影响了模型的真实感，所以本文提出了一种绘制方法，保证了被映射纹理最大限度的符合真实场景中的光照特性和物体的表面特征.

该方法具有以下优点：

(1) 不限制空间中编辑区域的形状，它可以是矩形、圆形，甚至是一个曲面.

(2) 通常纹理图案的尺寸和物体的大小不同，在传统的纹理映射中需要定义纹理的缠绕方式，而该方法由于采用了以参考图象为中心的映射方法，因此避免了纹理图案尺寸与物体大小不同所造成的影响.

(3) 考虑了真实场景中的亮度，使纹理与模型更好地融合在一起.

1　用空间雕刻算法得到 voxel 模型

空间雕刻[3,4]是基于 voxel 表达的体积建模[5]的一种方法，最初假设空间中的所有 voxels 是重建模型的一部分，然后选择空间最外层的一个 voxel，找到它在每一幅参考图象所在平面上的投影点(成像点)，如果这些投影点的颜色不一致，这个 voxel 就不是重建的物体的表面 voxel，这个 voxel 就会被雕刻掉[4]. 这个过程一直重复下去，直到所有的表面 voxel 在每一幅图象上的投影都满足照片一致性为止.

这个算法最重要的概念就是照片一致性. 对于空间中的一个 voxel，可以找到它在参考图象平面的投影点，如果它在各个图象中的投影点颜色一致，那么就称这个点满足照片一致性. 在空间雕刻算法中，需要定义一个一致性函数，用这个函数来确定一个 voxel 是否满足照片一致性，满足照片一致性的 voxel 就在重建模型上[3].

空间雕刻算法假设所有的表面都满足 Lambertian 模型. 这个模型假设空间中的任意一点在各个方向上发出的光线相等，也就是假设重建模型上的一个 voxel 在不同参考图象上的成像点颜色都一样，因为在实际中不存在完全符合这种模型的物体，所以必须在一致性函数中引入一个噪声参数. 文献[4]中用方差阈值作为一致性函数. 如果一个 voxel 在每一幅参考图象上的投影的方差小于一个用户指定的全局阈值，那么就可以认为它在各个参考图象上的成像点颜色相等，那么这个 voxel 就在

重建模型上.

上述的这种方法要求用户指出一个输入图象噪声的全局的估计值(全局阈值).这些噪声通常被认为是每个像素独立的传感噪声(sensor noise),通常是由于光照和定标误差导致的,所以,在一个输入图象的序列里,对噪声给出一个全局的估计是不适当的.在空间雕刻算法中,如果对误差估计得过低,那么就会有许多本来是表面点的 voxel 被雕刻掉,如果对噪声估计得过高,那么就不能正确的恢复物体的形状.在实际应用中,如一个场景中包括镜子和墙壁,其中镜子是镜面反射,而墙壁等物体是漫反射,所以如果规定唯一的噪声值,显然是不对的.文献[6]中提出了一种可变阈值的方法,这种方法同时计算在所有阈值下的重建模型,绘制图象中的一个像素时,这个像素确定从相机光心到物体上一点的一条光线,选择这条光线上最适合的 voxel 来进行绘制.这就保证了图象的每一部分都达到了最佳的重建效果,而不是用一个全局的阈值来重建.在实现上,给空间中的每一个 voxel 增加一个参数,这个参数是保留相应 voxel 的最小的阈值,当生成新视点的图象时,将每一个像素定义的光线投到空间中,选择这条光线上具有最小的阈值的 voxel 用来绘制.本文就采用这种方法得到 voxel 模型.

用三角面片的方法绘制 voxel 集合,效率十分低,因为每个 voxel 都有 12 个三角面片,在本文中,将重建模型表示为一组平行的平面,每一个平面看作一幅 32 位图象,这个图象的分辨率由原始图象确定[6],每一个像素为 32 位的 RGBA.这个 RGB 部分保存颜色向量,A 部分(a 通道)不再用来保存透明度信息,而是保存判断这个 voxel 是否是模型一部分的标志(阈值),如果估计的噪声大于这个阈值,这个 voxel 就是模型的一部分,否则就不是.设 $N\times N$ 的一个 voxel 集合,如果每个 voxel 单独表示,需要 $6N^3$ 个四边形,采用图层的方法,则只需 $3N$ 个四边形(如图 1 所示).用图层表示 voxel,大大降低了处理的计算量.

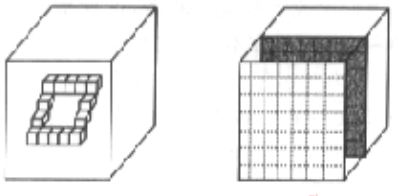


图 1 用图层来表达 voxel 集合

应用空间雕刻算法,首先需要判断一个 voxel 在一幅参考图象上是否可见.如果一个 voxel 和相机之间的 voxel 都是空的(即不在重建模型上),那么这个 voxel 在这幅图象上是可见的.确定一个 voxel 是否可见时,如果判断这个 voxel 确定的光线上的所有 voxel,那么计算量将相当大,其实,这是不必要的.在这个算法中,由于每个平面是按照从外到里的顺序处理的,所以每一个平面的处理结果可以缓存起来,当处理一个新平面时再更新缓存的值.由于原始的参考图象上的每一个像素都定义了从照相机到 voxel 模型的一条光线,这就意味着原始图象可以用来缓存这个值,如图 2 所示.在这个过程中,将 voxel 的 a 值二值化,将每一条光线上阈值最小的 voxel 的 a 值设为 0,其余的设为 255.

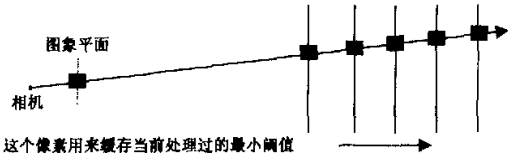


图 2 图象平面缓存是否可见信息

2 纹理编辑

IBR 系统的纹理编辑如图 3 所示,其不同于传统图形学上的纹理映射.本文中提出的纹理编辑方法,通过在原始的参考图象中指出纹理编辑的区域,可以将模型的纹理修改为指定的纹理,为了使重新施加的纹理能更好的融合到场景中去,在编辑过程中需要考虑场景中的亮度信息.

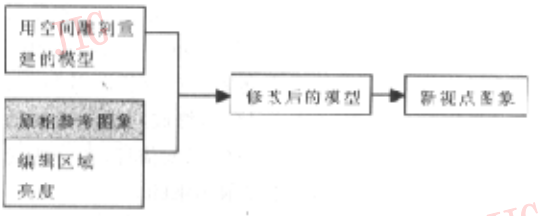


图 3 IBR 系统的纹理编辑

2.1 纹理的定义

纹理的定义有连续法和离散法两种.连续法把纹理函数定义为一个二元函数,函数的定义域就是纹理空间.离散法把纹理定义在一个二维数组中,代表纹理空间中行间隔、列间隔固定的一组网格点上

的纹理值. 网格点之间的其他点的纹理值可通过网格点的插值获得. 通过纹理空间与物体空间之间的坐标变换，可以把纹理映射到物体表面.

在三维可视化领域，离散法的纹理定义是最常用的. 该定义所采用的二级数组可以代表一个字符位图，而该字符位图既可以是用程序生成的各种图形，亦可以是扫描输入的数字化图象. 由于在现实中存在着大量的图象文件，而且这种纹理的真实感也是别的纹理不可比拟的，因此本文中用图象文件作为纹理的定义.

2.2　IBR 模型与纹理的映射

传统的纹理映射实现方法有正向映射和反向映射[7]两种. 正向映射是由纹理空间向图象空间映射，即将在纹理空间中预先定义的二维纹理映射到对象空间中的三维景物表面上，接着再进一步映射到图象空间的二维平面上. 正向映射的缺点在于容易产生图形的走样，由于纹理空间到对象空间的映射和对象空间到图象空间的映射是两个完全不同性质的过程，因此纹理像素和屏幕显示像素之间不是一一对应的，这样屏幕显示的纹理图象往往会出现严重的失真现象. 在一些情况下，某些显示像素内不包含任何纹理采样值，在另一些情况下，显示图象可能面目全非，尤其是当纹理映射到一个与其形状完全不同的区域时，情况更为严重，因此在正向映射方法中，须着重考虑的是图形的反走样问题. 逆映射是指由屏幕空间到景物空间、再由景物空间到纹理空间的一种映射方法. 具体地说，首先将屏幕空间上的一像素区域映射到曲面上，即进行图象空间至景物空间的映射，然后把该像素在曲面上的映射曲面区域映射到纹理空间.

空间雕刻方法中无场景中物体的几何描述，而是通过一些离散的 voxel 集合来描述物体的形状和颜色，也就无法直接使用正向映射或反向映射的方法.

在传统的图形学上，给出图象上一点，很难确定对应三维物体上的一点，光线追踪法虽然可以做到这点，但其花费极大. 然而在空间雕刻方法中，场景模型的表示方法可以用一种简单的方法达到光线追踪法的效果，这样就可以建立以原始的参考图象为中心的纹理映射方法，这种方法实现的步骤如下.

(1)从图象到纹理的映射　假设(x_m, y_m, z_m)为相机平面(图象平面)上的一点，(x_t, y_t, z_t)为纹理空间上的一点，那么在射影空间，它们之间存在如下变换

$$\begin{pmatrix} x_m \\ y_m \\ z_m \\ 1 \end{pmatrix} = \begin{pmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \\ p_{41} & p_{42} & p_{43} & p_{44} \end{pmatrix} \begin{pmatrix} x_t \\ y_t \\ z_t \\ 1 \end{pmatrix} \tag{1}$$

为了得到图象平面与纹理平面的映射，不妨设z_m、z_t为 0，则式(1)化简为：

$$\begin{pmatrix} x_m \\ y_m \\ 1 \end{pmatrix} = \begin{pmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{41} & p_{42} & p_{44} \end{pmatrix} \begin{pmatrix} x_t \\ y_t \\ 1 \end{pmatrix} = \boldsymbol{P} \begin{pmatrix} x_t \\ y_t \\ 1 \end{pmatrix} \tag{2}$$

矩阵 $\boldsymbol{P}$ 有 9 个参数，由于在射影变换中 $k\boldsymbol{P}$ 和 $\boldsymbol{P}$ 表示同一个变换，所以 $\boldsymbol{P}$ 的自由度为 8，一对对应点确定二个方程，所以只需要 4 对对应点就可以在相差一个常数因子的意义下得到图象和纹理平面的映射. 本文中纹理是长方形的，所以可以通过 4 个顶点来计算这个变换矩阵.

(2)从图象到模型的映射　这一步就是找出图象上一点 $\boldsymbol{P}_m$ 在空间中的位置，由于模型由空间雕刻方法重建，即找出图象点在模型中对应的 voxel. 图象中的一个点与相机的光心定义了一条光线，这条光线和模型相交的 voxel 即是所求的 voxel. 如果用这个方法求解 $\boldsymbol{P}_m$ 不但比较困难(因为不知道模型的几何描述)，而且计算量相当大. 从空间雕刻的方法可以看出，模型空间是用三组平行的平面(x, y, z 方向)表示的，平面的每一个像素可以看作 voxel 在此平面对应方向上的一个面，所以可以按到图象距离从近到远的顺序处理各个平面，观察图象点在这个平面上的对应 voxel 是不是物体的表面 voxel. 在处理空间中每一个平面时具体操作如下：

① 通过这幅图象的投影矩阵 $\boldsymbol{P}$ 计算从图象平面到这个平面的变换矩阵. 在空间雕刻算法中，首先需要定义 voxel 集合空间的位置和大小，即 voxel 空间的 8 个顶点在空间中的坐标，就可以计算出正在处理的平面的 4 个顶点的空间坐标，另外由于已经计算出各个参考图象对应的投影矩阵，因此可以计算出这 4 个顶点在图象上的投影坐标，此外，在空间雕刻过程中定义此平面的宽度 w 和高度 h，为计算方便，不妨设这 4 个顶点在正在处理的平面上的坐标为$(0,0)$、$(w,0)$、(w,h)、$(0,h)$，然后，通过式(2)计算出变换矩阵.

② 给定图象上的一个像素 $\boldsymbol{P}_m$，用上面计算出的变换矩阵求出像素 $\boldsymbol{P}_m$ 在正在处理的平面上的对应像素 $\boldsymbol{P}_t$. 检查像素 $\boldsymbol{P}_t$ 的 α 值是否是零，如果是零

说明像素 P_t 是像素 P_m 在空间中的对应点，否则这个平面上没有与像素 P_m 对应的点.

由于空间中 voxel 并不与图象中的像素一一对应，空间中的一个 voxel 可能对应参考图象上的数个像素，因此采用超采样的方法来减少这种情况带来的误差，即首先确定超采样的倍数 t，然后通过将参考图象的坐标乘以 t 转化为超采样后的坐标，通过将对角线上的元素乘以 t 来修改图象平面到模型空间平面的变换矩阵. 处理空间中的每一平面时都用超采样后的坐标计算，显著提高了处理的效果.

2.3 亮度

用上面方法修改的纹理由于没有考虑亮度的因素，经过纹理编辑后，掩盖了物体的表面细节(见图 4(d))，而且在整个场景中的纹理的亮度一样，不能符合施加纹理区域的原始明暗状况(见图 4(e)，原来场景中侧面的亮度远远低于正面，但是施加纹理后，纹理的亮度相同)，与原来的物体不能很好的融合在一起，所以必须加入对亮度的处理，才能达到更好的效果.

假设场景物体表面为漫反射，场景中的各部分在每个视点上具有相同的光强，即在视点 A 中，如果区域 $\boldsymbol{R}$ 比区域 $\boldsymbol{T}$ 亮度高，那么在视点 B 中，区域 $\boldsymbol{R}$ 的亮度也比区域 $\boldsymbol{T}$ 的亮度高. 根据经验可知，即使场景物体表面不完全为漫反射，大多数情况也会满足此假设，这样通过一幅原始参考图象即可以得到场景中各个部分的光强，并以此作为参考，计算新施加纹理的亮度.

亮度采用 HSI 模型中的亮度分量. 颜色表示方法常见的模型包括 RGB(表示红、绿、蓝)模型、HSI(表示色相、饱和度、亮度)等. HSI 模型中的亮度是颜色的相对明暗程度，通常以 0%(黑)到 100%(白)的百分比来度量. HSI 色彩空间和 RGB 色彩空间只是同一物理量的不同表示法，因而它们之间存在着转换关系，对于任何 3 个在[0,1]范围内的 RGB 值，RGB 模型到 HSI 模型的转化公式为

$$I=\frac{1}{3}(R+G+B)$$

$$S=1-\frac{3}{(R+G+B)}[\min(R,G,B)] \tag{3}$$

$$H=\arccos\left\{\frac{[(R-B)+(R-B)]/2}{[(R-G)^2+(R-B)(G-B)]^{1/2}}\right\}$$

若设 S,I 的值在[0,1]之间，R,G,B 的值也在[0,1]之间，则 HSI 到 RGB 的转换公式为：

当 H 在[0°,120°]之间

$$B=I(1-S)$$

$$R=I\left[1+\frac{S\cos H}{\cos(60°-H)}\right] \tag{4}$$

$$G=3I-(B+R)$$

当 H 在[120°,240°]之间

$$R=I(1-S)$$

$$G=I\left[1+\frac{S\cos(H-120°)}{\cos H}\right] \tag{5}$$

$$B=3I-(R+G)$$

当 H 在[240°,360°]之间

$$G=I(1-S)$$

$$B=I\left[1+\frac{S\cos(H-240°)}{\cos(300°-H)}\right] \tag{6}$$

$$R=3I-(G+B)$$

文中方法的思路是以原始参考纹理的颜色为基础，并且考虑原始参考图象上的对应区域的亮度信息，方法的具体实现如下：

当确定场景中一点的颜色时：

(1) 通过 2.2 节中的方法得到对应参考图象上的点 P_m 和纹理上的点 P_t，

(2) 用式(3)计算 P_t 和 P_m 的亮度 I_t 和 I_m.

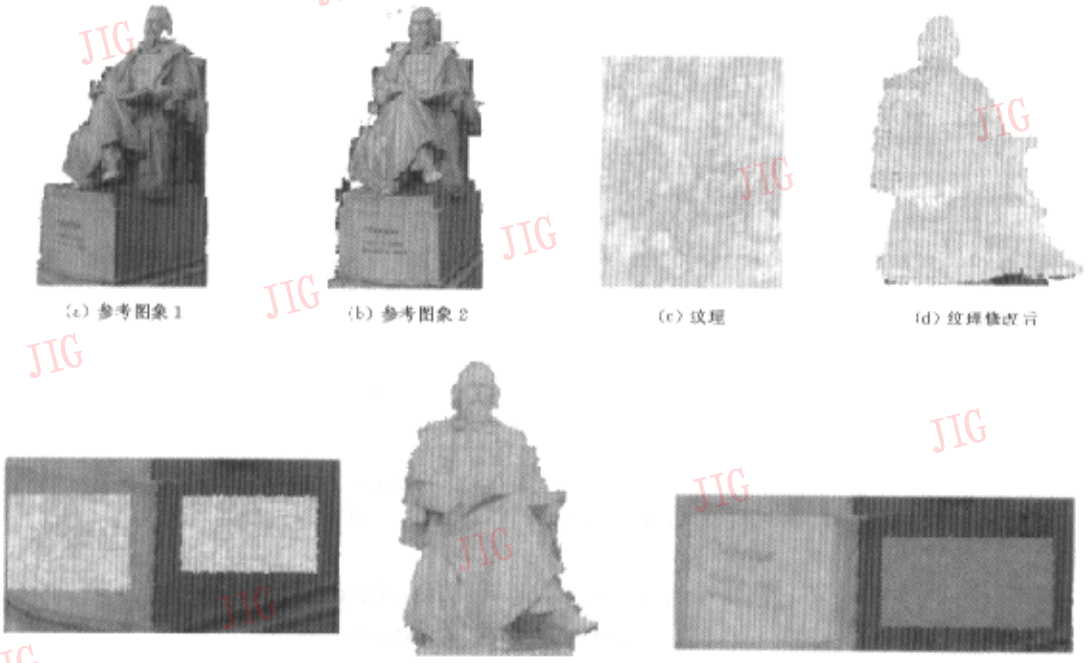
(3) 为了不影响纹理的颜色，新亮度必须考虑纹理的亮度信息和参考图象的亮度. 定义函数 f，$I_{new}=f(I_t,I_m)$. 这里 $f(I_t,I_m)=\sqrt{I_t\times I_m}$，用这个公式计算新的亮度值.

(4) 使用式(4)或式(5)或式(6)重新计算此点的 RGB 值，其中 H 和 S 为第 2 步根据 P_t 计算得出，I 的值为第 3 步计算的 I_{new}.

2.4 实验结果和结论

实验共用 8 张参考图象，图 4 中的(a)，(b)为其中的两张，(c)为所用的纹理图象，(d)和(e)是没有考虑亮度的情况下得到的结果，它完全抹煞了物体的表面特征，并且在图 4(e)中的两个面上，纹理的亮度是一样的，完全忽略了光照的影响，严重影响了真实感，即便如此，它还是验证了本方法的一些优点，如不限制编辑区域的形状；不管场景中的物体多么复杂(在本实验中，雕像的形状非常不规则)，都可以完成纹理编辑. 在图 4(f)中考虑了亮度的影响，较好的保留了物体的表面特征；在图 4(g)中右侧表面纹理的亮度明显比左侧的暗，使纹理具有了符合真实场景的明暗度，显著提高了真实感.

本方法存在处理速度慢、无法直接应用于实时处理领域的问题，下一阶段将对算法进行改进，简化



(a) 参考图象 1　(b) 参考图象 2　(c) 纹理　(d) 纹理修改后

(e) 修改两个面　(f) 加入亮度处理后　(g) 加入亮度处理后

图 4　实验结果

计算，提高处理速度.

参 考 文 献

1 Seitz S M, Kutulakos K N. Plenoptic image editing[A]. In: IEEE International Conference on Computer Vision Proceedings [C]. Mumbai, India, 1998: 17～24.

2 Sudeep Rangaswamy. Interactive editing tools for image-based rendering system[D]. Massachusetts: Department of Electrical Engineering and Computer Science, 1998.

3 Seitz M N, Dyer C R. Photorealistic scene reconstruction by voxel coloring[J]. International Journal of Computer Vision, 1999,25(3):151～173.

4 Kutulakos K N, Seitz S M. A theory of shape by space carving [J]. International Journal of Computer Vision, 2000,38(3): 198～218.

5 Dyer D R. Volumetric scene reconstruction from multiple views [A]. In: Foundations of Image Understanding [C]. Boston, USA, 2001:469～489.

6 Broadhurst A, Drummond T. A probabilistic framework for space carving [A]. In: IEEE International Conference of Computer Vision[C]. Vancouver, Canada, 2001: 388～393.

7 孙家广，杨长贵. 计算机图形学(新版)[M]. 北京：清华大学出版社，1995,497～504.

张志强　1978 年生，现为天津大学电信学院计算机系硕士研究生. 主要研究方向为基于图象的造型与绘制、计算机图形学.

孙济洲　1949 年生，天津大学计算机系教授、博导. 研究方向为计算机图形学、计算机体系结构、并行计算等. 发表论文 30 余篇.