基于直接投射扩散的烟雾阴影实时模拟

湛永松[1)] 杨明浩[2)] 石民勇[1)] 费广正[1)]

[1)](中国传媒大学数字技术与艺术研发中心,北京 100024) [2)](中国科学院自动化研究所数字互动媒体实验室,北京 100080)

摘 要 为了提高烟雾模拟的真实感,提出了一种基于直接投射扩散的烟雾阴影实时生成算法。该方法将 3 维烟雾密度场直接投射到 2 维平面上,并保存于光亮缓存中,首先生成一张能描述阴影浓度信息的阴影纹理;然后为了解决阴影浓度过估计问题,还新颖地采用流体力学领域的扩散方程来修正浓度分布情况;最后,在可编程图形硬件上,采取投影纹理映射技术将阴影纹理投射到场景中的相应表面。实验证明,该方法所需计算开销小,能简单高效地实时生成计算机 3 维游戏和动画的烟雾阴影效果。

关键词 阴影纹理 直接投射扩散 过估计 投影纹理映射 烟雾阴影

中图法分类号:TP391.9 **文献标识码**:A **文章编号**:1006-8961(2008)11-2231-07

Real Time Smoke Shadow Simulation Based on Direct Projection and Diffusion

ZHAN Yong-song[1)], YANG Ming-hao[2)], SHI Min-yong[1)], FEI Guang-zheng[1)]

[1)] (*Digital Technology and Digital Art R&D Center, Communication University of China, Beijing* 100024)

[2)] (*Digital Interactive Media Laboratory, Institute of Automation, Chinese Academy of Sciences, Beijing* 100080)

Abstract To improve the realistic effect of smoke simulation, a real time smoke shadow simulation algorithm based on direct projection and diffusion is proposed. Firstly, the 3D smoke density field is directly projected to a 2D plane and accumulated into a light buffer to generate a texture which describes the shadow energy. Then, diffusion is used as a novel energy spread scheme to deal with the shadow excessive accumulation induced by direct projection. Finally, the shadow texture is mapped to a corresponding surface in 3D scene by projective texture based on programmable graphics hardware. Experiment results show that the algorithm is efficient, simple and effective to achieve real time smoke shadow for 3D computer games and animation.

Keywords shadow texture, direct projection and diffusion, excessive accumulation, projective texture, smoke shadow

1 引 言

阴影是现实生活中很常见的光照自然现象,它是由于光源被物体遮挡而在其后产生的较暗区域。在真实感图形学中,由于通过阴影可以提供物体的位置和方向信息,从而可反映出物体之间的相互空间关系,并能增加图形的立体感和场景的层次感。但目前对烟雾阴影模拟的研究通常为人们所忽略,这是因为烟雾属于低反射率参与介质(participating media)[1],不能使用传统的造型方法建模,因此常规的阴影模拟算法并不适用于生成烟雾阴影;此外,由于流体模拟的计算开销较大,难以获得实时的模拟效果。

在处理传统的基于面片表示的空间实体时,一种常用的阴影生成算法是影域多面体法。影域

基金项目:教育部科技研究重点项目(JK2G02035);国家自然科学基金项目(60403037)

收稿日期:2006-07-10;**改回日期**:2007-07-19

第一作者简介:湛永松(1979 ~),男。2007 年获中国传媒大学通信与信息系统专业工学博士学位,现为香港城市大学博士后研究人员,Senior Research Associate。主要研究方向为计算机流体动画技术、计算机视觉、GIS 开发与应用等。E-mail: yszhan@cityu.edu.hk

(shadow volume)是一个以光源直接照射面为顶面,由其轮廓边与光源所张平面为侧面而形成的半开放区域,任何包含于影域之内的景物表面必处于该光源的阴影中。Crow 根据该原理提出了在图像空间判定某一景物表面或其中一部分是否包含于影域多面体内的算法[2]。该算法是基于扫描线的消隐算法,其在绘制时,是将所有的阴影边界多边形如同普通多边形一样参加扫描和排序。另一种常见的阴影生成算法为阴影细节多边形法,其先以光源为视点,再按景物空间消隐算法求取景物表面上的阴影区域,并将它们作为阴影细节附在表面多边形上。该算法在取不同视点绘制画面时,其生成的阴影细节多边形能够重复使用,从而极大提高了阴影的生成效果。基于 Weiler-Atherton 多边形裁剪的阴影细节多边形算法即是根据这一原理提出[3]。该算法分为两步实现,即首先取光源方向为视线方向进行消隐,确定对光源不可见的表面阴影细节多边形,并通过赋予相同的标识数来将它们与所覆盖的原始景物多边形相关联;然后取视线方向对景物进行第 2 次消隐,此时的阴影细节多边形只是一种表面细节,它们并不作为独立的景物多边形加入第 2 次消隐。在绘制时,如果多边形上某一部分相对视点可见,但被阴影细节多边形所覆盖,则该区域的光亮度按阴影处理。此外,常见的阴影生成算法还包括 Z 缓存器法[4]和阴影测试线法[5]。但上述方法都不适用于烟雾等低反射率参与介质的阴影生成。

基于物理的流体模拟是近年来计算机图形学领域的一个研究热点,但相关研究并未涉及对烟雾阴影的模拟[6,7]。Splatting 算法是一种以物体空间为处理顺序的直接体绘制算法[8],它源于对 3 维重构的分析,即根据某个重构核,通过卷积运算将离散的 3 维数据场重构成连续数据。Splatting 算法将每个体素视为空间中一个采样点的 3 维核函数,先沿视线方向进行光亮度积分来获得 3 维核的 2 维投影(称为 footprints,足迹函数);然后在图像平面上累积其贡献,但庞大的 3 维数据处理量会限制绘制速度。

受 Splatting 算法启示,本文提出了一种基于直接投射扩散(direct projection and diffusion,DPD)的烟雾阴影实时模拟方法。文中以 3 维密度场来描述烟雾,同时通过将密度场投射到 2 维平面来生成一张能描述阴影浓度信息的阴影纹理,并保存于光亮缓存。由于直接投射所生成的阴影纹理会存在浓度过估计问题,因此本文新颖地利用流体力学中的扩散方程来修正浓度分布情况,并在可编程图形硬件上采取投影纹理映射(projective texture)[9]的方法,先将阴影纹理投射到场景中相应表面,然后再将该表面映射回 2 维视平面上,最终即可获得投影正确的实时烟雾阴影效果。

2 DPD 算法

该算法的目的是生成一张 2 维纹理用来描述烟雾阴影浓度信息,包括直接投射和能量扩散两个阶段。算法首先将 3 维烟雾密度场中各体素直接投射到 2 维图像平面,以获得一张阴影纹理;然后再通过流体力学领域的扩散策略来进行能量扩散,以修正阴影纹理的浓度过估计问题;最终生成一张浓度分布和谐的 2 维阴影纹理。

2.1 直接投射

烟雾被描述为一个经过网格划分的 3 维密度场,每个网格对应一个体素,而烟雾浓度则定义于各网格的中心位置,图 1 为该网格划分的 2 维示意图。直接投射就是将 3 维密度场中的各体素直接投射到 2 维图像平面上,并将体素值累加到相应的 2 维光亮缓存中。该过程并未进行能量扩散处理,属于一种简单的前向行为,即各体素不经过任何加权处理即进行直接投射。图 2 所示为直接投射方式。

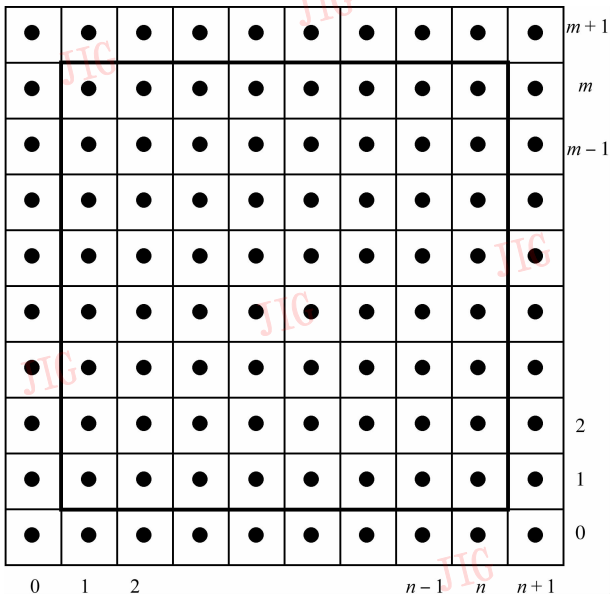


图 1 烟雾密度场网格划分 2 维图示

Fig. 1 Density field defined in the center of each voxel

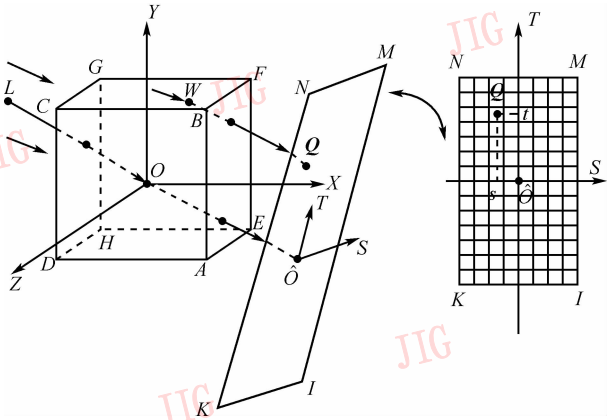


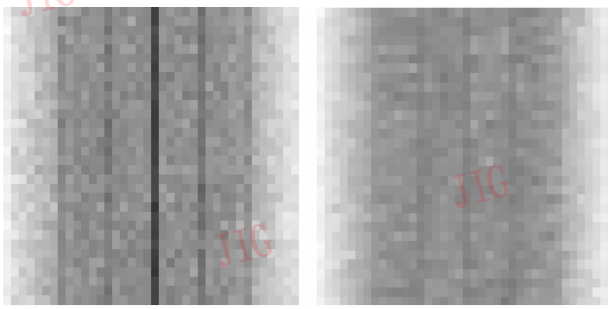
图 2　直接投射示意图

Fig. 2　Direct projection

图 2 中,OX、OY、OZ 分别代表 3 维世界坐标系的正向,L 为光源位置,$ABCDEFGH$ 为烟雾的包围盒,O 为坐标系原点,即包围盒中心,W 为当前待投射的任意体素。2 维面 $KIMN$ 为投影面,它与 LO 垂直,$\hat{O}S$、$\hat{O}T$ 分别为该平面的坐标系正向。在直接投射的过程中,可将入射光线视为平行光。点 $\hat{O}$ 和点 $\boldsymbol{Q}$ 为原点 O 和体素 W 分别被投射到 2 维平面上的点。

投射步骤如下:首先根据光源位置和烟雾包围盒信息确定 2 维投影面,并将其划分为如图 2 右边图所示的 2 维网格,该网格对应一个 2 维光亮缓存 buf。对于 3 维密度场中的任意体素 W,首先计算其在投影平面上的相应位置 $\boldsymbol{Q}$,然后将其值直接累加到对应的光亮缓存中。假设当前体素 W 的密度值为 D_W,投影点 $\boldsymbol{Q}$ 在投影平面上的横纵坐标分别为 s 和 t,则 2 维光亮缓存 buf 可通过以下方式进行简单的累加求取:$buf[s][t] += D_W$。

直接投射的结果是将 3 维烟雾密度场生成为一张 2 维阴影纹理(如图 3(a)所示),其中 3 维密度场网格划分为 32 × 32 × 48,阴影纹理网格划分为 64 × 48。由于光亮缓存中只保存了烟雾密度信息,而且密度越大的地方,相应的阴影浓度更深,因此该纹理能大致描述出阴影浓度的分布情况。由于直接投射的过程只涉及到投影点位置确定和浓度值累加,也避免了 Splatting 算法中对所有 3 维数据进行能量扩散的处理,因而所需运算量更少。



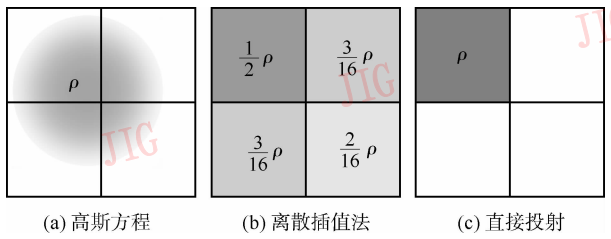
(a) 能量扩散前的阴影纹理　　(b) 能量扩散后的阴影纹理

图 3　能量扩散前后的阴影纹理

Fig. 3　Shadow texture before and after diffusion

2.2　能量扩散

为了实现能量扩散,Splatting 算法是通过高斯方程作为 3 维核来生成足迹函数 footprints(如图 4(a)所示);此外,也可通过如图 4(b)所示的离散插值法来扩散能量。本文的直接投射采取了如图 4(c)所示方式来将各体素直接投射到光亮缓存,由于忽略了能量的扩散过程,因此阴影纹理中会出现浓度过估计的问题,即阴影浓度分布不均匀,高低浓度区之间差别明显的现象(如图 3(a)所示)。



(a) 高斯方程　　(b) 离散插值法　　(c) 直接投射

图 4　能量扩散方式

Fig. 4　Energy spread schemes

针对烟雾的流体扩散(diffusion)特性(如图 5 所示),本文提出了一种新颖的能量扩散方法,即先将直接投射所生成的阴影纹理作为处理对象,然后采取流体力学领域的扩散策略[6]在 2 维平面上进行能量扩散。

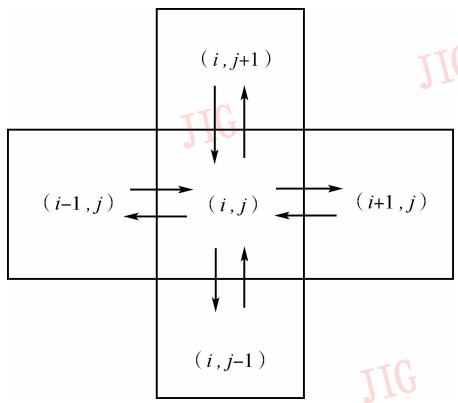


图 5　流体扩散特性

Fig. 5　Fluid diffusion

流体运动规律受 Navier-Stokes 方程[10]支配，该方程的扩散项如式(1)所示。其中，$\boldsymbol{u}$ 为速度，v 为流体粘度系数，t 为时间。本算法将 $\boldsymbol{u}$ 视为密度，v 为一个控制扩散速度的参数，采用式(2)所示的隐式方程求解扩散项。根据图 5 可见，该隐式方程的数值求解可采取如式(3)所示的有限差分法完成，其中 h 代表网格跨度，可用于调整阴影浓度分布。

$$\frac{\partial \boldsymbol{u}}{\partial t} = v\nabla^2\boldsymbol{u} \tag{1}$$

$$(I - v\Delta t\nabla^2)\boldsymbol{u}_1 = \boldsymbol{u}_0 \tag{2}$$

$$u_1(i,j) = (h^2u_0(i,j) + v\Delta t(u_0(i-1,j) + u_0(i+1,j) + u_0(i,j-1) + u_0(i,j+1))) \times \frac{1}{h^2+4v\Delta t} \tag{3}$$

与 Splatting 算法相比，DPD 算法是针对烟雾的流体特性，采取流体力学领域中的扩散方程将能量处理转移到 2 维平面进行，从而有效减少了所需处理的数据量。图 3(b)为采用流体力学扩散策略对图 3(a)进行能量扩散后所获得的阴影纹理图，显然图 3(b)比图 3(a)具有更和谐的浓度分布，从而有效解决了阴影浓度的过估计问题。

3 投影纹理映射

前面使用 DPD 算法生成了一张描述阴影浓度信息的 2 维纹理。为了获得最终的烟雾阴影效果，本文采取投影纹理映射[9]技术将其投射到 3 维场景中的相应表面。投影纹理映射是通过类似于投影机播放幻灯片的方式将纹理投影到场景中，其与普通纹理映射的区别在于纹理坐标的计算方式不同。普通纹理映射通常采用非齐次纹理坐标(s,t,r)，其在进行纹理映射时，是先通过对纹理坐标进行插值来计算物体上各点的纹理坐标，然后使用该纹理坐标来索引纹理图像。而投影纹理映射则采用齐次纹理坐标，其在进行纹理映射时是首先对齐次纹理坐标进行插值得到 4 个分量的纹理坐标(s,t,r,q)，然后对该纹理坐标进行透视变换得到最终的纹理坐标$(s/q,t/q,r/q)$，并用来对纹理图像进行索引。

投影纹理映射并不要求为每个顶点明确指定纹理坐标，而是需要明确指定相应的变换矩阵即纹理矩阵(texture matrix)，以便根据物体空间的每个顶点位置来计算纹理坐标。纹理坐标计算公式如式(4)所示，其中 $\boldsymbol{M}$ 为模型矩阵，$\boldsymbol{V}$ 为视图矩阵，$\boldsymbol{P}$ 为投影矩阵。由于将投影空间转换为标准坐标空间后，每个顶点坐标的取值范围为$[-1,1]$，而纹理坐标的范围是从 0 到 1，所以通过最后一个矩阵对顶点坐标进行缩放和偏移，以便使得最终生成的纹理坐标落在$[0,1]$范围内。

$$\begin{bmatrix} s \\ t \\ r \\ q \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \boldsymbol{P} \cdot \boldsymbol{V} \cdot \boldsymbol{M} \cdot \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \tag{4}$$

投影纹理能先把场景的空间坐标映射到 2 维纹理空间，并可确定场景中各顶点映射到纹理的哪个部分，然后将该纹理位置当作纹理坐标赋给顶点，并在渲染时，把适当的一块纹理应用到每个三角形上，其原理如图 6 所示。

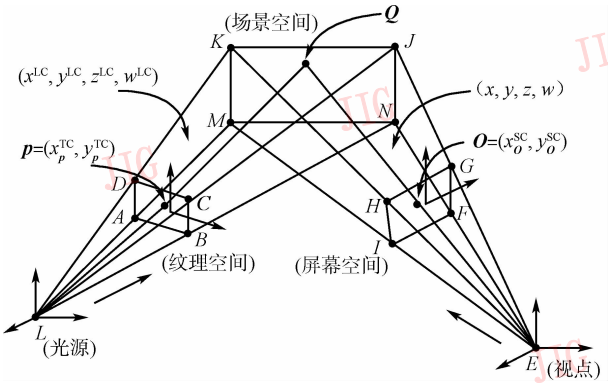


图 6 投影纹理原理图

Fig. 6 Projective texture

图 6 中 L、E 分别代表光源和视点的位置，$ABCD$ 为 2 维阴影纹理，$IFGH$ 为屏幕，$MNJK$ 为场景空间平面。投影纹理使用到以下 4 个坐标系：

(1)视点坐标系(eye coordinator, EC)。它是裁剪坐标系，其是以视点位置 E 为坐标原点，各点的坐标值可表示为齐次坐标$[x, y, z, w]$。

(2)光源坐标系(light coordinator, LC)。它是裁剪坐标系，其是以光源位置 L 为坐标原点，各点的坐标值可表示为齐次坐标$[x^{LC}, y^{LC}, z^{LC}, w^{LC}]$。

(3)屏幕坐标系(screen coordinator, SC)。它是 2 维坐标系，其对应于屏幕空间，各点坐标用$[x^{SC}, y^{SC}]$表示。其中，$x^{SC} = x/w$，$y^{SC} = y/w$。

(4)纹理坐标系(texture coordinator, TC)。它是 2 维坐标系，其对应于待投影到场景的 2 维阴影纹

理,各点坐标用$[x^{TC}, y^{TC}]$表示。其中,$x^{TC} = x^{LC}/w^{LC}$,$y^{TC} = y^{LC}/w^{LC}$。

如图 6 所示,屏幕坐标系上点 $\boldsymbol{O}$ 为已知点,它对应于场景中的点 $\boldsymbol{Q}$,而点 $\boldsymbol{Q}$ 又是从阴影纹理上的点 $\boldsymbol{p}$ 投影而来。投影纹理的目的是为已知的点 $\boldsymbol{O} = [x_O^{SC}, y_O^{SC}]$ 寻找对应点 $\boldsymbol{p} = [x_p^{TC}, y_p^{TC}]$,并建立两者的转换关系,其步骤如下:

(1)根据点 $\boldsymbol{O} = [x_O^{SC}, y_O^{SC}]$,求取点 $\boldsymbol{Q}$ 在视点坐标系中的坐标 $\boldsymbol{Q}^{EC}$。

(2)由于光源坐标系与视点坐标系为仿射关系,两者之间必然存在仿射矩阵 $\boldsymbol{M}_{affine}$,从而可将点 $\boldsymbol{Q}$ 在视点坐标系的坐标转换为光源坐标系下的坐标,即 $\boldsymbol{Q}^{LC} = \boldsymbol{M}_{affine} \times \boldsymbol{Q}^{EC}$。

(3)求取点 $\boldsymbol{p} = [x_p^{TC}, y_p^{TC}]$。根据已求得的 $\boldsymbol{Q}^{LC} = [x_Q^{LC}, y_Q^{LC}, z_Q^{LC}, w_Q^{LC}]$,通过式(4)求取点 $\boldsymbol{Q}$ 在纹理空间的对应纹理坐标。

于是屏幕上点 $\boldsymbol{O}$ 的颜色值 C_o 为纹理空间上点 $\boldsymbol{p}$ 的颜色值 C_p 与场景中对应点 $\boldsymbol{Q}$ 的颜色值 C_Q 的合成(如式(5)所示),其中 α 为控制合成效果的系数。

$$C_o = C_p \times \alpha + C_Q \times (1 - \alpha) \tag{5}$$

由于目前的可编程图形硬件语言 Cg[11] 对投影纹理映射有很好的支持,因此,本文充分利用可编程图形硬件的高速运算优势,在 GPU 上完成投影纹理映射过程,即首先采用 Cg 的顶点程序(vertex program)来计算场景中各顶点所对应的投影纹理坐标,再将经过插值生成的四元纹理坐标集传给片段程序(fragment program),最后由片段程序求取最终的颜色效果。

顶点程序从物体空间位置计算投影纹理坐标的 Cg 语句为

```
float4 texCoordProj = mul(textureMatrix,position);
```

片段程序负责接收经过插值的四元纹理坐标集。当其使用纹理坐标集来存储一个 2 维纹理时,标准库函数 tex2Dproj 需要使用纹理坐标 q(第 4 分量)除以纹理坐标 s、t(第 1 和第 2 分量)。片段程序中对应的 Cg 语句为

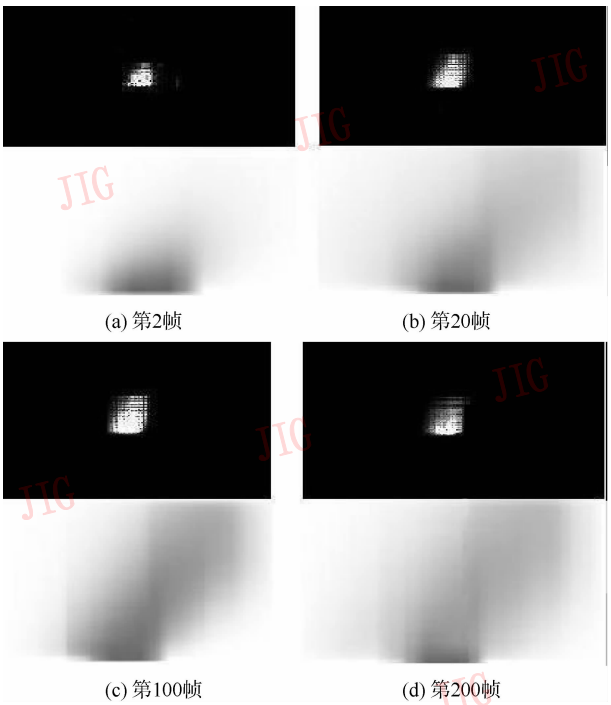
```
float4 textureColor = tex2Dproj(projTexture,texCoordProj)
```

由于将投影纹理映射放在 GPU 上完成,不但解放了 CPU,而且减少了两者之间的通讯,从而有利于提高运算效率,并能保证烟雾阴影模拟的实时性。

4　结果分析

在 Dell Precision380 工作站(奔腾 3.2G CPU,Nvidia Quadro Fx540 显卡,1G 内存)上实现了本文算法,开发平台为开放图形库 OpenGL、可编程图形硬件语言 Cg 与 Visual C++。

如图 7 所示,上方黑色区域内的白色模糊状物体代表 3 维密度场描述的烟雾,其网格划分为 32 × 32 × 32,下方为采用 DPD 算法生成的相应阴影纹理,网格划分为 64 × 64。由于烟雾属于流体物质,并没有明确的轮廓形状,因此阴影纹理所表现出的模糊分布状是合理的。图 7(a)~图 7(d)分别对应模拟过程的第 2、20、100 以及 200 帧图像,其描述了 3 维烟雾在产生、扩散和衰减等不同状态下所生成的相应阴影纹理。由图 7 可见,烟雾刚产生时(第 2 帧),阴影浓度较淡,分布区域也较小;随着烟雾逐渐扩散,阴影浓度值以及分布区域都在增加;在第 100 帧时,烟雾扩散为最大,相应的阴影浓度最深,分布区域也最大;在第 200 帧时,烟雾逐渐减弱,此时的阴影浓度开始变淡,分布区域也有所回缩。由此可见,本文的实验结果与算法思想之间具有较好的吻合性。



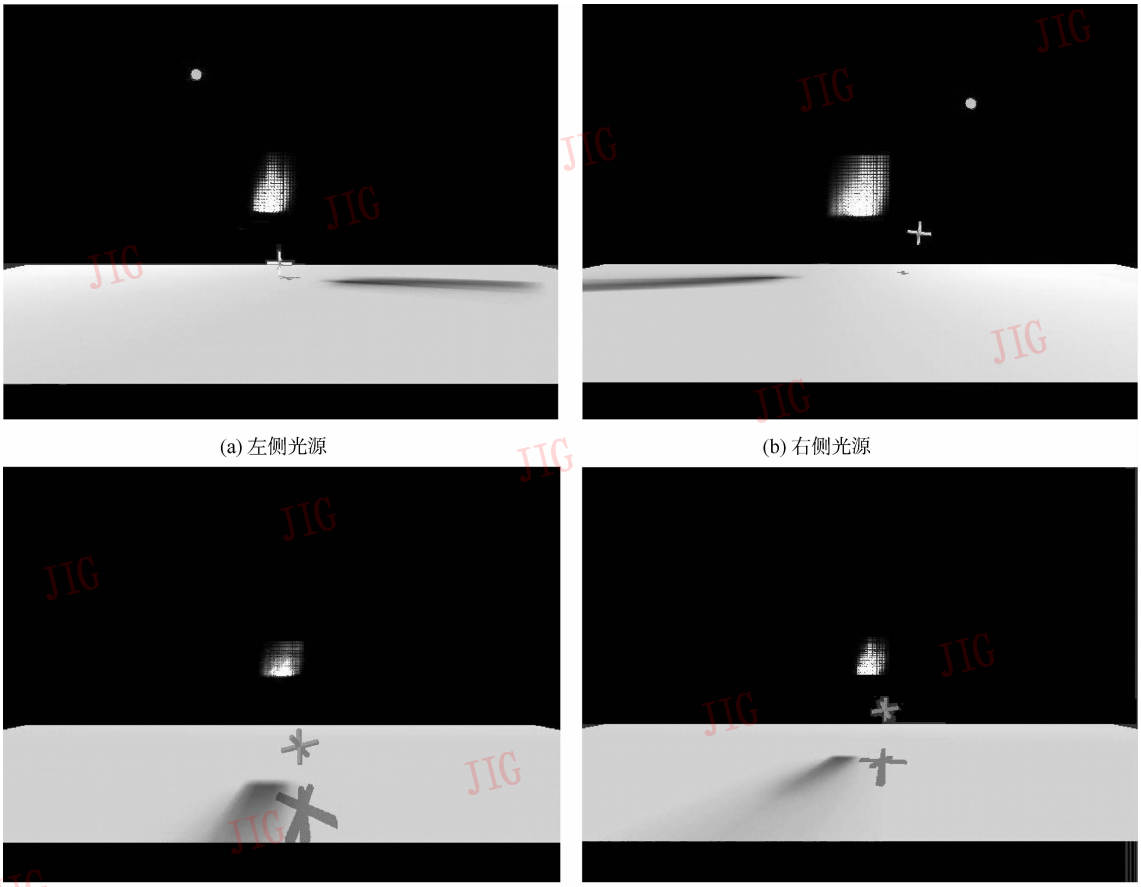
(a) 第2帧　(b) 第20帧

(c) 第100帧　(d) 第200帧

图 7　不同烟雾状态所对应的阴影纹理

Fig. 7　Evolution of shadow texture

图 8 为 3 维场景中,光源分别照射在烟雾和刚体上所生成的实时阴影效果。其中,图上方的小亮点代表光源,白色模糊状物体为使用 3 维密度场表示的烟雾,十字架代表刚体。从图 8 中可以看到,随着光源位置变动以及烟雾形态变化,阴影效果也发生相应变化。



(a) 左侧光源 (b) 右侧光源

(c) 正后侧光源 (d) 斜后侧光源

图 8 3 维场景中烟雾与固体十字架在不同光源照射下的阴影效果

Fig. 8 Shadow simulation of smoke and solid cross using different lightings

DPD 算法中,由于直接投射只需确定投影点的位置以及进行简单的数值累加,因此计算开销不高,而且能量扩散被转移到 2 维平面进行,也有效减少了所需处理的数据量。最后的投影纹理映射是在可编程图形硬件上完成,而高速的 GPU 处理速率则能保证绘制的实时性。表 1 给出了本算法处理不同网格划分的密度场和阴影纹理时,生成每一帧画面的耗时。当 3 维密度场的网格划分不高于 64 × 64 × 64 时,本系统不仅能保持较好的画面连贯性,而且能模拟实时的烟雾阴影效果。但随着网格划分变细,画面会出现跳帧现象,使模拟实时性受到影响。网格精细度与模拟实时性一直是基于物理的流体模拟中的一对矛盾体,目前仍未有彻底的解决方法,只能针对实际的情况进行折中选择。本文的实验结果表明,当密度场和阴影纹理的网格划分分别为 32 × 32 × 32 和 64 × 64 时,就能在视觉真实感与模拟实时性之间取得好的平衡。

表 1 不同网格划分下的耗时比较

Tab. 1 Simulation cost under different voxel division

密度场	阴影纹理	耗时(ms)
32 × 32 × 32	64 × 64	16
64 × 64 × 64	64 × 64	38
100 × 100 × 100	128 × 128	135
256 × 256 × 256	256 × 256	378

5 结 论

本文首先将烟雾描述为 3 维密度场,然后通过

先直接投射,再进行能量扩散的方式来生成阴影纹理,并基于可编程图形硬件采用投影纹理映射来实时模拟烟雾的阴影效果。本文的主要贡献是提出了一种新颖的基于直接投射扩散的阴影纹理生成方法,并基于可编程 GPU 加速了绘制过程。本算法所需处理的数据量要远小于传统的体绘制法,从而保证了模拟的实时性。

未来的工作主要集中在以下 3 个方面:(1)将 DPD 算法应用到其他低反射率参与介质的阴影模拟;(2)研究 DPD 算法是否适用于普通模糊状物体以及高反射率参与介质的体绘制;(3)研究多光源情况下的烟雾阴影模拟,以获得更具有视觉真实感的效果。

参考文献(References)

1 Jensen H W, Christensen P H. Efficient simulation of light transport in scenes with participating media using photon maps[A]. In: Proceedings of SIGGRAPH[C], Orlando, Florida, USA, 1998: 311~320.

2 Crow F C. Shadow algorithm for computer graphics[J]. Computer Graphics, 1977, **11**(3):242~248.

3 Atherton P R, Weiler K, Greenberg D P. Polygon shadow generation [J]. Computer Graphics, 1978, **12**(3):275~281.

4 Catmull E. Computer display of curved surfaces[A]. In: Proceedings of the IEEE Conference on Computer Graphics, Pattern Recognition and Data Structures[C], Los Angeles, California, USA, 1975: 11~17.

5 Cook R L. Shader trees[J]. Computer Graphics, 1984, **18**(3): 223~231.

6 Stam J. Stable fluids[A]. In: Proceedings of SIGGRAPH[C], Los Angeles, California, USA, 1999: 121~128.

7 Fedkiw R, Stam J, Jensen H. Visual simulation of smoke[A]. In: Proceedings of SIGGRAPH[C], Los Angeles, California, USA, 2001: 15~22.

8 Westover L. Footprint evaluation for volume rendering[A]. In: Proceedings of SIGGRAPH[C], Dallas, Texas, USA, 1990: 367~376.

9 Segal M, Korobkin C, Widenfelt R, *et al*. Fast shadows and lighting effects using texture mapping[A]. In: Proceedings of SIGGRAPH[C], Chicago, Illinois, USA, 1992:249~252.

10 Chorin A J, Marsden J E. A Mathematical Introduction to Fluid Mechanics (Texts in Applied Mathematics 4)[M]. Second Edition. New York,USA: Springer-Verlag, 1990.

11 Fernando R, Kilgard M J. The Cg Tutorial-the Definitive Guide to Programmable Real-time Graphics[M]. Bk&CD-Rom edition. New York,USA: Addison-Wesley Professional, 2003.