doi: 103969/j. issn. 0490-6756. 2016. 11. 010

使用 Light Field Rendering 实现森林的实时渲染及光照算法

方　震，杨红雨，林　毅
(四川大学计算机学院视觉合成图形图像技术国防重点学科实验室，成都 610065)

摘　要：森林的实时渲染及光照是视景系统中的一个难题. 基于图像的渲染方法(IBR)由于渲染速度与模型复杂度无关，被广泛应用于场景重建. 基于光流场(Light Field Rendering)的IBR 技术，提出一种迭代投射算法来进行外形重建，实现了具有实时光影特征的森林效果. 实验表明该算法结合了传统迭代、投射算法各自的优点，在质量和效率方面取得了平衡.

关键词：森林渲染及光照；基于图像的渲染方法；光流场；迭代投射算法
中图分类号：TP391. 9　　**文献标识码**：A　　**文章编号**：0490-6756(2016)06-1241-06

Real-time rendering of forests using light field rendering and lighting algorithm

FANG Zhen, *YANG Hong-Yu*, *LIN Yi*
(National Key Laboratory of Fundamental Science on Synthetic Vision,
College of Computer Science, Sichuan University, Chengdu 610065, China)

Abstract: Real-time rendering and lighting of forests is a difficult problem in visual system. Due to uncorrelated of scene complexity, the Image-based rendering techniques have been widely used for the scene reconstruction. An iterative-casting algorithm based on Light Field Rendering (IBR) is put forward to solve the reconstruction pro-blem, and then realizes a real-time forests rendering with lighting effects. The experimental results show that this algorithm has combined the advantages of iterative and casting algorithms, gets a balance between efficiency and quality.

Keywords: Forest rendering and lighting; Image-based rendering; Light field rendering; Iterative-casting algorithm

1　引　言

如何高效地绘制森林一直是虚拟现实中的重要研究方向. 由于现实中的森林及树木结构极其复杂，传统的绘制方法比如几何建模、billboard 等方法都存在效率低下或者是真实度较低的缺点. 要想高效地进行森林绘制，就要求绘制算法与模型复杂度以及场景复杂度无关. 为此，一些学者提出了基于图像的渲染方法(Image-based Rendering, IBR)，由于此方法和模型复杂度不相关，渲染效率高，所以受到广泛应用. 但是由于传统 IBR 方法对其理论基础：全光函数(The Plenoptic Function)的维度进行了限定，导致渲染的结果具有无法响应光照变化、逼真度不够等种种缺点.

2　相关工作

森林的快速渲染一直是计算机图形学的研究热点. 许多学者提出使用一个或多个 billboards 来

收稿日期：2015-12-29
基金项目：国家高技术研究发展规划"863"项目(2015AA016404－3)；中央高校基本科研业务费资助项目 (2015SCU1102)
作者简介：方震(1992－)，男，河南信阳人，硕士生，研究方向为虚拟现实、飞行仿真.
通讯作者：杨红雨. E-mail: yanghongyu@scu. edu. cn.

模拟单棵树木[1,2],这种方法简单高效,但是靠近观察容易出现破绽,而且由于 billboard 的自遮挡[3]导致树木对光照的响应容易出错.

另外一种被广泛使用的方法就是使用 3D 建模来构建森林.文献[4,5]总结了多种 3D 树木建模的方法,包括基于规则、草图、图像以及手工建模等方法,但是在实际应用中发现,3D 建模方法虽然效果好,但很多时候不能满足场景实时绘制的要求.文献[6]提出使用在 GPU 上进行处理的动态方格来渲染森林,并取得了良好的 LOD 效果.

基于图像的渲染技术由来已久.可以快速解决树木外形重建这个问题.IBR 技术的理论基础为全光函数[7],如式(1)所示,假设 t 时刻相机(人眼)坐标 $P=P(V_x,V_y,V_z)$,从方向 $v=v(\theta,\Phi)$ 上射入相机的光线波长为 λ,则这束光线可以表示为

$$p=p(\theta,\Phi,\lambda,t,V_x,V_y,V_z) \tag{1}$$

如式(1),知道这 7 个变量,便可以确定任意位置处的全景信息.但是计算 7 维度的全光函数非常困难,如何降低维数便成为 IBR 技术研究的热点.Todt 等人提出使用球形光流场[8](Spherical Light Field, SLF),通过在球面上不同位置进行采样来降低维度,再结合相应的深度信息提纯算法,取得了良好的效果,但是需要在效果和效率间进行权衡.

在光照方面,赵作林等人[9]提出使用概率模型来模拟树木间接光照,文献[10,11]提出使用多参数化的基于视觉深度的指数衰减算法来计算树木的阴影信息,但是计算过程复杂,且与场景复杂度相关,对森林绘制的效率产生很大影响.另外,使用常见的 Shadow Mapping[12,13] 及其改进算法[14]也可以用来生成森林阴影,但是这些算法均无法计算到树木的内部自阴影.本文采用文献[15]提出的方法,对森林中树木的遮挡进行粗略的建模估计,来达成光照及阴影效果.

3 基于球形光流场 SLF 的外形重建

为了在新视角下重建模型,需要事先处理保存相关采样点下的模型信息.如图 1 所示,首先将树木模型缩放到 1/sqrt(2.0)单位大小,以便相机可以在距离模型中心单位距离的位置上看全整个模型;即模型的包围球半径为 1/sqrt(2.0)单位,相机在半径单位 1 的包围球上运动采样.由于树木模型的上方信息更为有效,为了减少存储消耗,所以只在其上半球 Ω^+ 内采样.在 Ω^+ 中均匀采样 181 个点,针对每一个点,均把相机移动到这个位置上来,对准模型的中心,保存一张 2D RGBz 的纹理,其中 RGB 为颜色信息,z 为深度比例信息.为了降低存储量,z 值保存的是一个深度比例,即 $z/2$,这样一来,z 的变化范围就变成(0,1),可以存入八位的 alpha 通道中.另外,还要保存相机在各个采样位置处的视图投影矩阵 M,供后面使用.

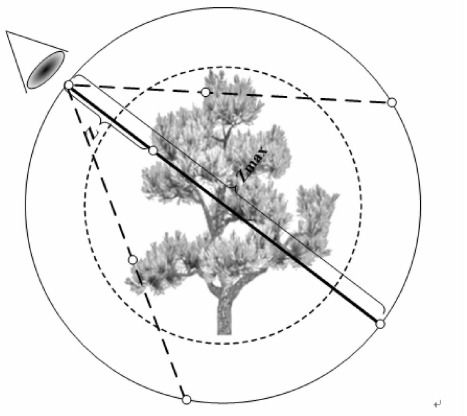

图 1 树的采样
Fig. 1 Tree model sampling

3.1 迭代算法[8]

如图 2 所示,对于一个新的视点观察方向 V,要想得到 E_0 的准确信息,首先取得离这个方向最近的三个采样点 C_0、C_1、C_2.然后使用不断迭代逼近的方法来定义这个视线方向所看到的 E_0 点的颜色信息.

传统的迭代方法[8]对这临近的三个采样点采取投影到视线 $\boldsymbol{VE}$ 上的方法来重建.经过研究发现,视点离某一采样点越近,投影后的长度也越接近其原始长度.为了简化计算过程,在找到三个临近采样点的同时,会根据距离远近同时生成三个对应的权重系数 w_0、w_1、w_2(系数和为 1).将这些权重系数结合使用,可以明显简化计算流程.

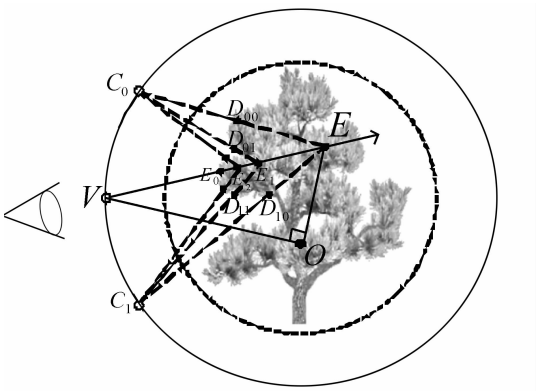

图 2 迭代算法
Fig. 2 Iterative algorithm

具体的迭代算法步骤如下

步骤 1　如图 2，针对一个具体的观察方向 $\boldsymbol{V}$，首先找到其临近的三个采样点 C_0、C_1、C_2（图中未标出），计算出各自的权重系数 w_0、w_1、w_2，并获取各自的视图投影矩阵. 然后作视点和模型中心的连线即 VO，再过 O 点作直线 OE 垂直于 VO 交观察方向于点 E.

步骤 2　如式(2)所示，使用 $\boldsymbol{OE}$ 和各自采样点的视图投影矩阵相乘，获取到对应纹理坐标，读取出采样点在此处的深度值（对应图上的 C_0D_{00}、C_1D_{10}）. i 取 0、1、2，代指临近的三个相机，j 代表迭代次数（下同）.

$$u_{ji} = M_i * \mathbf{OE} \tag{2}$$

$$\mathrm{color}_{ji} = texture\ (\mathrm{tex}_i, \mathrm{u}_{ji}.\mathrm{xy}) \tag{3}$$

步骤 3　在得到个采样点的深度值 $color_{ji}.w$ 后，乘以各自的权重系数和深度缩放比例，得到新的深度值 $depth = VE_1$，连接 OE_1 作为新的投影向量.

$$VE_j = V + \sum_{i=0}^{2} w_i * color_{ji}.w * 2 \tag{4}$$

步骤 4　使用 OE_1 作为新的投影向量，重复步骤 2 和 3，直到两次结果误差达到事先设定的精度范围或迭代次数达到上限（实际使用中上限设为 5）.

$$error = VE_j - VE_{j-1} \tag{5}$$

步骤 5　当迭代达到设定的精度后，结合权重获取最终的颜色信息 $texColor$.

$$texColor = \sum_{i=0}^{2} w_i * color_{ji}.rgb \tag{6}$$

迭代算法不能精确地重建模型外形，因为这种算法在一些情况下是失效的. 如图 3 中 AB 所示，在这两种情况下，临近的三个采样点中，只要有一个无法观察到目标点，迭代过程就会以失败告终，导致结果出现误差. 加大采样密度虽然可以降低这种情况发生的几率，但是仍不能从根源上解决这个问题.

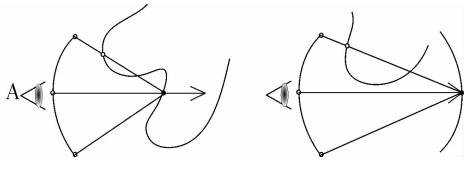

图 3　迭代失效的情形

Fig. 3　Invalid scenes of Iterative algorithm

3.2　光线投射算法[8]

光线投射算法[8]和迭代算法的第一步均相同，即首先要获取到点 E，然后取到视线 VE 和模型包围盒的交点作为起始投影点 E_1，然后将 EE_1 等分成 N 份，产生 $N+1$ 个投影点，对每个投影点进行投影计算

$$color_{ji} = M_i * \mathbf{OE}_j \tag{7}$$

每次得到新的 color_{ji} 后，比如下图 4 中点 E_2，C_0D_{02} 的长度（深度值）即为 $color_{20}.w * 2$，我们要比较 C_0E_2 和 C_0D_{02} 的长度差. 当式(8)中 $error$ 小于事先设定的误差范围，则停止.

$$error = C_iD_{ij} - C_iE_j \tag{8}$$

如果临近的三个采样点都满足此条件，则使用式(6)来计算新的颜色，若有一个采样点不满足此条件，意味着出现了图 3 所示的情况，此时将其权重设为零，并重新调整剩余采样点的权重来计算新的颜色信息. 这样就可以从根源上解决图 3 所示的问题，所以投射算法比迭代算法更精确但是计算开销明显增大，效率低下.

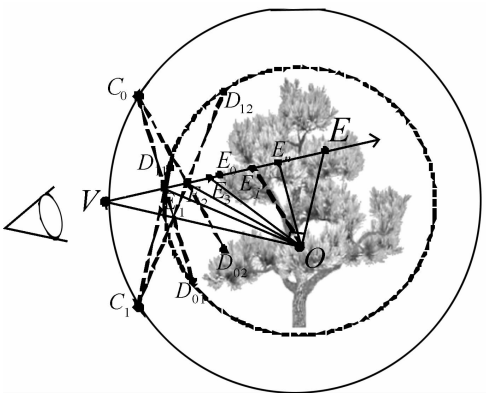

图 4　光线投射算法

Fig. 4　Raycasting algorithm

3.3　本文新算法：迭代-投射算法

迭代算法具有速度快的优点，但是精度低；而投射算法虽然精度高，但是效率低下. 可以使用少数几次的迭代来产生投射算法的起点 E_1，再进行投射计算. 这样会带来一个问题，如图 5(a)所示投射方向向内，图 5(b)图投射方向则向外. 所以对于新的观察方向势必要内外均投射，这样才能产生正确的结果，但是这样一来，算法的时间复杂度和单纯的投射算法并没有很大差别.

本文提出一种算法来结合这两种算法以达到速度与效率的权衡. 经过研究发现，图 3 中所示的问题大多出现在模型形状起伏较大的地方，而对于平坦的区域则很少出现这种情况. 所以，可以在模型起伏变化较小的地方使用迭代算法，而在起伏变化大的地方使用光线投射算法. 新算法的具体流程如下

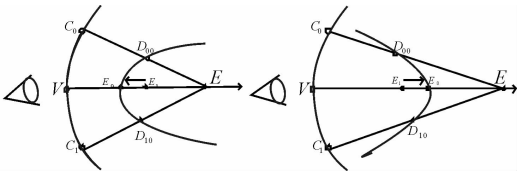

图 5 迭代方向
(a) 外凸点 (b) 内凹点
Fig. 5 Iterative directions
(a) Convex point, (b)Concave point

(1) 如图 6 所示,对所有的点均先进行迭代计算,当迭代结束后,假设当前迭代点为 E_j,然后计算临近三个采样点的误差,即:比较 E_jD_{0j}、E_jD_{1j}、E_jD_{2j},取这三个误差值中最大的一个作为判别误差 *error*.

$$error=\max(E_jD_{0j},E_jD_{1j},E_jD_{2j}) \tag{9}$$

(2) 将 *error* 与事先给定的值相比较,如果超出误差范围则代表模型在此处起伏变化较大,即出现了图 3 所示的状况. 此时我们再调用光线投射算法来精确构建物体的外形特征. 我们可以利用最新生成的点 E_j 作为起始点来降低计算量,虽然投射的方向仍然不确定,但是可以以 E_j 作为起点来进行光线投射过程,先使用 $\boldsymbol{E}_{j-1}\boldsymbol{E}_j$ 作为投射方向,若误差满足条件则结束投射过程,否则再逆向投射. 这样平均会有 50% 的几率来减少大部分的计算量,从而达到提高效率的目的.

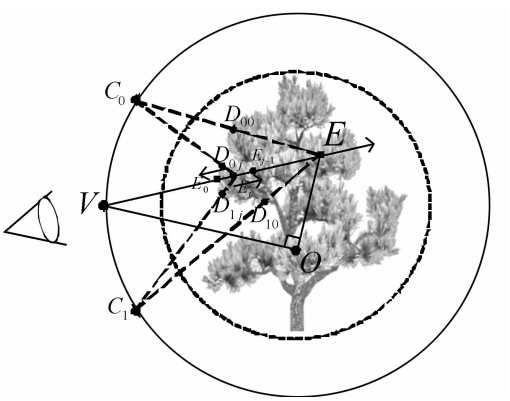

图 6 迭代-投射算法
Fig. 6 Iterative-Raycasting algorithm

4 森林光照算法

为了使模拟出来的森林中的每棵树均具有内部阴影和外部阴影,而且使森林整体具有热点反光等和实际森林更加一致的效果,使用文献[15]提出的方法实现森林的模拟,为了计算树木自身的环境遮挡信息,将采样图片中的颜色信息替换为单棵树木内部的自身遮挡信息. 为了追求效率,这里并不提供准确的树木外形以及光照效果,只是通过近似的方法来模拟. 具体计算如下.

(1) 树木外形实时重建:经过实际研究发现,对于视线方向,使用两次迭代投射算法来获得视线的入口 $\boldsymbol{P}_v$;然后对于光照方向,假设光线通过 $\boldsymbol{P}_v$,再进行一次迭代算法获取光线的入口 $\boldsymbol{P}_l$,此时效果最好. 这样便得到树木外形的重建点 $\boldsymbol{P}_v$,以及光线的传播路径 $\boldsymbol{P}_v\boldsymbol{P}_l$.

(2) 太阳光照计算:假设太阳光线只在树木表面发生散射,这样就只需要考虑光线由 $\boldsymbol{P}_v$ 传向 $\boldsymbol{P}_l$ 的情况. 所以,太阳光在 $\boldsymbol{P}_v$ 处的散射就可以表示为 $\rho p(\boldsymbol{v},\boldsymbol{l})exp(-\tau\|\boldsymbol{P}_l-\boldsymbol{P}_v\|)L_{sun}$,$\boldsymbol{v}$ 代表视线的方向,$\boldsymbol{l}$ 代表光照的方向,ρ 为树木反射率,$p(\boldsymbol{v},\boldsymbol{l})$ 为相函数,τ 为衰减系数(使用时取 10),L_{sun} 为太阳亮度. 实际使用中,我们还要乘以太阳的可见性 $V(\boldsymbol{P}_l)$ 来获取树木之间的遮挡阴影效果. 另外,使用式(11)所示的 $f(\boldsymbol{v},\boldsymbol{l})$[16] 为森林加入一个反光热点,其中 a_1、a_2 均为经验数据. $\boldsymbol{P}_v$ 处太阳光照的计算公式如式(10)所示(以下所有公式及文中的"$*$"符号均代表算术乘法):

$$sun(\mathbf{P}_v)=\rho p(v,l)\exp(-\tau\|\mathbf{P}_l-\mathbf{P}_v\|)*f(\mathbf{v},\mathbf{l})V(\mathbf{P}_l)L_{sun} \tag{10}$$

$$f(\mathbf{v},\mathbf{l})=1-a_1[(1-\exp(-a_2\cos^{-1}(\mathbf{v}\cdot\mathbf{l})))] \tag{11}$$

(3) 天空亮度计算:对于 $\boldsymbol{P}_v$ 处的天空反射,我们使用 $L_{sky}*\delta(\boldsymbol{P}_v)$ 来近似,$\delta(\boldsymbol{P}_v)$ 为 $\boldsymbol{P}_v$ 处的环境遮挡. $\delta(\boldsymbol{P}_v)$ 的近似计算方法如下:假设树内遮挡和树间遮挡互不影响,便有 $\delta(\boldsymbol{P}_v)=\delta_v(\boldsymbol{P}_v)*\delta_e$,$\delta_v(\boldsymbol{P}_v)$ 为树内自身遮挡(迭代时已算出),δ_e 为树间的遮挡. 对于 δ_e,设 $\delta_e=\delta_h*\delta_s$,$\delta_h$ 为水平地面上树木间的遮挡因子,δ_s 为斜坡上的遮挡因子. 假设树木之间的平均距离为 r(可用树木密度来近似),我们使用水平面上的一个半径为 r,深度为 d(用 $\boldsymbol{P}_v.z$ 计算)的洞来模拟计算水平面上的环境遮挡,所以遮挡因子 δ_h 可以近似表示为 $1/(1+d^2r^{-2})$. δ_s 因子则用 $(1+\boldsymbol{n}*\boldsymbol{u}_z)/2$ 来模拟,$\boldsymbol{n}$ 表示地面法线,$\boldsymbol{u}_z$ 表示垂直向上的单位向量. 这样计算的结果便具有了内外阴影的效果. $\boldsymbol{P}_v$ 处天空亮度计算表示为式(12):

$$sky(\mathbf{P}_v)=\frac{\rho}{2}\left(\int_{\Omega^+}p(\mathbf{v},\mathbf{w})\mathrm{d}\,w\right)*\frac{\delta_v(\mathbf{P}_v)(1+\mathbf{n}*\mathbf{u}_z)}{1+d^2r^{-2}} \tag{12}$$

5 实验结果及分析

5.1 实验环境

实验硬件配置为:CPU Intel Core i5-3550 3.

30GHz，8.00GB RAM，NVIDIA GeForce GTX 560 Ti 显卡. 软件环境为：Windows 7 64 位旗舰版，Visual Studio 2010，OpenGL 4.3，着色器语言为 GLSL.

5.2　实验结果及分析

由于树木的外形极其复杂，不利于观察效果. 所以我们使用 bunny 模型进行上述各种外形重建算法的实验. 图 7 (a)为原始模型图像，图 7 (b)为迭代算法产生的结果，图 7 (c)为本全文提出的新算法结果，图 7 (d)为光线投射算法产生的结果. 各算法的帧率如表 1 所示. 通过对比我们可以发现这三种算法中，光线投射算法的重建结果清晰无重影，与源图像差异最小，平均帧率为 61fps；迭代算法的重建结果整体有重影，边界部分(如耳朵)重影尤为明显，平均帧率为 96fps；本文提出新算法的重建结果在图像内部仍有重影，但是边缘清晰，符合人眼对物体边缘信息更敏感的观察规律，平均帧率达到 75fps，较光线投射算法的帧率提高了 23%.

(a)源图像　(b)迭代算法　(c)迭代-投射算法(d)光线投射算法

图 7　bunny 模型的重建结果

Fig. 7　Reconstructions of bunny model

表 1　算法帧率对比

Tab. 1　The frame rate of different algorithms

算法	迭代算法[11]	迭代-投射算法	投射算法[11]
平均帧率(fps)	96	75	61

森林算法的实验结果如图 8、图 9、图 10 所示。在图 8 中可以发现单棵树木对阳光的照射有自身遮挡的效果(黑色方框内所示). 而且树木各部分的明暗变化自然，单棵树木(512×512 分辨率)的渲染帧率为 62；图 9 为森林效果，图 10 为阴影效果. 在图 9 中，使用该算法同时渲染了上百棵树木，帧率仍达到了 60，满足了实时性渲染的要求.

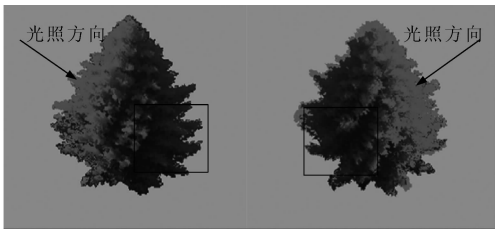

图 8　单棵树效果

Fig. 8　Result of single tree

图 9　森林效果

Fig. 9　Result of forests

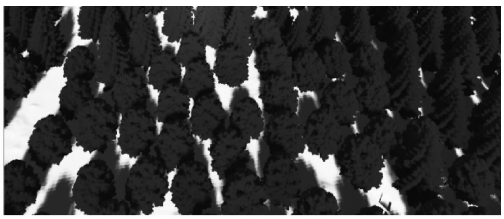

图 10　阴影效果

Fig. 10　Result of shadow

6　结　论

本文通过对传统基于图像渲染算法的研究，提出一种新的基于 LFR 的迭代-投射算法，在提升帧率的同时兼顾了渲染效果. 随后利用 LFR 技术确定树木外形，再使用光照算法实现逼真的实时森林光影效果. 但是，新算法对树木细节的渲染效果不好，近距离观察会出现失真现象，且无法响应外界风速的变化等，这些均是进一步的工作研究内容.

参考文献：

[1]　Behrendt S, Colditz C, Franzke O, *et al*. Realistic real-time rendering of landscapes using billboard clouds[J]. Computer Graphics Forum, 2005, 24(3): 507.

[2]　卞燕山，邹鹏，于荣欢，等. 近界视景导航中森林场景加速绘制[J]. 国防科技大学学报，2014，(6)：31.

[3]　魏厚明，刘冬香，曹卫群，等. 布告板云树木模型的阴影快速生成与绘制[J]. 计算机辅助设计与图形学学报，2011，(5)：771.

[4]　谭云兰，贾金原，张晨，等. 3D 树木建模技术研究

进展[J]. 中国图象图形学报, 2013, 18(11): 1 520.

[5] 刘真余, 芮小平, 董承玮. 森林三维真实感建模与可视化LOD技术研究[J]. 中国科学院研究生院学报, 2011, 28(3): 322.

[6] Liu F, Wei H, Bao H J. GPU-based dynamic quad stream for forest rendering[J]. Science China Information Sciences, 2010, 53(8): 1539.

[7] Adelson E, Bergen J. The plenoptic function and the elements of early vision[C]// Computation Models of Visual Processing. Cambridge, MA: MIT Press, 1991.

[8] Todt S, Rezk-Salama C, Kolb A. Fast (Spherical) light field rendering with per-pixel depth [R]. Siegen: University Of Siegen, 2007.

[9] 赵作林, 李兆民, 杨刚, 等. 基于概率模型的树木间接光照效果的实时模拟[J]. 计算机工程与应用, 2015(15): 161.

[10] Boulanger K, Bouatouch K, Pattanaik S. Rendering trees with indirect lighting in real time[J]. Computer Graphics Forum. 2008, 27(4): 1189.

[11] Gumbau J, Chover M, Rebollo C, *et al*. Real-Time illumination of foliage using depth maps. [M]// Computational Science - ICCS 2008. Berlin Heidelberg: Springer Berlin Heidelberg, 2008.

[12] Williams L. Casting curved shadows on curved surfaces[C]// Proceeding of the 5th annual conference on Computer graphics and interactive techniques. New York: ACM Press, 1978.

[13] Scherzer D, Wimmer M, Purgathofer W. A survey of real-time hard shadow mapping methods [J]. Computer Graphics Fortum, 2011, 30(1): 169.

[14] 罗德宁, 叶万方, 杨红雨. 基于边缘拟合阴影图的实时硬阴影生成[J]. 四川大学学报:自然科学版, 2013, 50(2): 261.

[15] Bruneton E, Neyret F. Real-time realistic rendering and lighting of forests[J]. Computer Graphics Forum, 2012, 31(2): 373.

[16] Chen J M, Cihlar J. A hotspot function in a simple bidirectional reflectance model for satellite applications[J]. Journal of Geophysical Research Atmospheres, 1997, 1022(D22): 25907.