HAL
open science

Real-time Realistic Rendering and Lighting of Forests

Eric Bruneton, Fabrice Neyret

To cite this version:

Eric Bruneton, Fabrice Neyret. Real-time Realistic Rendering and Lighting of Forests. Computer Graphics Forum, 2012, 31 (2pt1), pp.373-382. 10.1111/j.1467-8659.2012.03016.x . hal-00650120

HAL Id: hal-00650120

https://inria.hal.science/hal-00650120v1

Submitted on 9 Dec 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

EUROGRAPHICS 2012 / P. Cignoni, T. Ertl
(Guest Editors)

Volume 31 (2012), Number 2

Real-time Realistic Rendering and Lighting of Forests

Eric Bruneton[1] and Fabrice Neyret[1]

[1]INRIA Grenoble Rhône-Alpes, Université de Grenoble et CNRS, Laboratoire Jean Kuntzmann

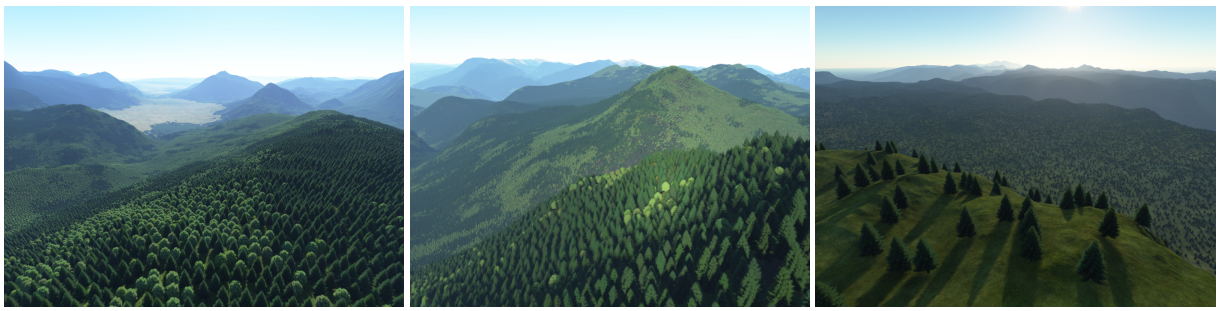

Figure 1: *Some real-time results obtained with our method, showing large forest scenes with a wide range of view distances, various tree densities and lighting conditions. The lighting effects reproduced by our method are shown in Fig. 9.*

Abstract

Realistic real-time rendering and lighting of forests is an important aspect for simulators and video games. This is a difficult problem, due to the massive amount of geometry: aerial forest views display millions of trees on a wide range of distances, from the camera to the horizon. Light interactions, whose effects are visible at all scales, are also a problem: sun and sky dome contributions, shadows between trees, inside trees, on the ground, and view-light masking correlations. In this paper we present a method to render very large forest scenes in real-time, with realistic lighting at all scales, and without popping nor aliasing. Our method is based on two new forest representations, z-fields *and* shader-maps*, with a seamless transition between them. Our first model builds on light fields and height fields to represent and render the nearest trees individually, accounting for all lighting effects. Our second model is a location, view and light dependent shader mapped on the terrain, accounting for the cumulated subpixel effects. Qualitative comparisons with photos show that our method produces realistic results.*

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism—

1. Introduction

Forest rendering is an important topic in Computer Graphics because forests are an essential part of many natural scenes. But it is a difficult problem, in particular in real-time applications like flight simulators or virtual Earth browsers, because real aerial forest views can show millions of trees, each having thousands of leaves. Depicting consistent forest illumination and reflectance at all scales is a challenge per itself, but is essential for realism. Lighting gives important visual cues, up to really far views, that help distinguish trees and understand the trees and terrain shapes, including (see Fig. 9):

- *view-dependent reflectance*: a forest appears brighter when one is facing away from the sun, rather than facing it, because one sees mostly lit parts in the first case, and mostly shadowed parts in the second case.
- *slope-dependent reflectance*: for the same reasons, and for given viewing and lighting conditions, the average reflectance of a forest varies with the terrain slope.
- *opposition effect*: a bright *hotspot* appears opposite to the sun, where all shadows are masked, at all scales (whole trees, groups of leaves, and individual leaves).
- *silverlining*: the silhouettes of backlit trees appear brighter than the rest of the tree because they are optically thinner and thus less opaque.

© 2011 The Author(s)
Computer Graphics Forum © 2011 The Eurographics Association and Blackwell Publishing Ltd. Published by Blackwell Publishing, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main Street, Malden, MA 02148, USA.

- *sky illumination*: due to the masking of sky light by the trees, their base (*resp.* interior) appears darker than their top (*resp.* exterior), and the ground is darker below trees.

Rendering and lighting of individual trees is already well studied. But most forest rendering algorithms use only basic lighting models, which can not reproduce the above effects (mostly due to view-light correlations). Thus, our goal is to propose a new model for the real-time realistic rendering and lighting of forests for medium to far views, supporting transitions with existing models for inside and close views. Our goal is to reproduce the above effects, without necessarily using "exact" equations, but without aliasing nor popping. We also want a consistent lighting at all scales, *i.e.,* the average reflectance of a forest patch must be independent of its screen resolution. Finally, we want to support user control of as many parameters as possible: density and shape of forests, geometry of trees, distribution of species and sizes, etc.

To achieve these goals we propose two new forest representations, called *z-fields* and *shader-maps*, with seamless transitions between them, which are the main contributions of this paper. A z-field combines light field and height field ideas. It is used to render the nearest trees, and supports transitions with 3D models for close views. A shader-map is a texture rendered with view and light dependent shaders, and is used to render the farthest trees.

Our paper is organized as follows. We review related work in Sections 2 and 3. Then, we present an overview of our algorithm in Section 4, our two new forest representations in Sections 5 and 6 and the seamless transition between them in Section 7. We discuss implementation details in Section 8, and then show our results and validations in Section 9. Finally, we discuss the limitations of our algorithm in Section 10, and explore avenues for future work in Section 11.

2. Related work

Trees and forests have been extensively studied in Computer Graphics. Here we exclude the review of modeling and level of detail algorithms for individual trees (see [BMG06] for a survey). Indeed, we start from already modeled trees, and restrict ourselves to medium to far forest views (see Section 4).

Forest representations. A common approach is to represent each tree with one or more billboards [CCDH05, FMU05, BCF+05, AMM07]. This approach is simple, and trees can be distributed arbitrarily. But many billboards are needed to get accurate parallax effects, and popping occurs when circling around intersecting trees. Another approach is to use 3D textures representing whole forest patches, tiled aperiodically and rendered with slices parallel to the terrain [DN04, BCF+05]. This drastically reduces the geometric complexity, but the distribution of trees and the forest boundaries are difficult to control below the scale of patches. The canopy can also be represented with a height field, rendered with relief mapping [MJ06]. User control is easier but only far views are supported. Conversely, point-based rendering [DCSD02, GMN05] is more adapted for close views.

None of these methods can reproduce the lighting effects presented in introduction. [DN04, CCDH05, FMU05, AMM07] simply ignore shadows. [BCF+05] represent the light transport with spherical harmonics coefficients stored in billboard textures, which cannot reproduce shadows from other trees. [MJ06] compute shadows and ambient occlusion using the canopy height field, which cannot account for self-shadowing inside trees. Finally, no method would give a correct lighting for apparent tree sizes smaller than a pixel.

Trees and forest lighting. Direct sun light shadows in individual trees are computed using either the tree geometry [MO95, MNP01] or a participating media approximation, *i.e.,* an exponential attenuation based on the optical depth [RB85, GCRR08, BBP08]. Likewise, direct sky light is (pre)computed using either the geometry [QNTN03] or a participating media approximation [HPAD06, BBP08]. This approximation is also used to estimate indirect lighting in trees [BBP08], and global illumination in forests [GS08] – by solving a radiative transfer equation in a 3D grid, which is not scalable and not real-time. Our method ignores indirect lighting, and combines ideas from [RB85, MO95] and [QNTN03] to estimate the direct sun and sky light in the nearest trees. For the farthest trees we switch to a view and light dependent shader modulated with a density map.

Forest reflectance models. There is no large scale forest reflectance model in Computer Graphics. The fakefur model [Gol97] is quite close to our needs. But hairs are different from trees, and this model ignores view-light correlations and thus does not reproduce the hotspot effect. The multiscale pine tree BRDF in [MN00] is even closer, but does not extend to the forest scale. There are however several forest models in physics, based either on radiative transfer theory [LS93a], or geometric optics [LS85, SJ90, CL97]. We extend the Strahler et al. model [SJ90, LS92, SLS94], presented below, because it is best adapted to our needs.

3. Strahler et al. model

In the context of remote sensing, Strahler et al. [SJ90, LS92, SLS94] modeled the large scale reflectance of forests by using a geometric optical model based on ellipsoids. Using our own notations, the hypothesis and parameters of their model are the following (see Fig. 2). Each tree crown is represented with an *opaque* ellipsoid of horizontal radius R and height H, on top of a trunk of null radius and height B. R, H and B can vary, but the tree proportions $h \stackrel{\text{def}}{=} H/R$ and $b \stackrel{\text{def}}{=} B/R$ are assumed constant. We note A the average horizontal tree area $A \stackrel{\text{def}}{=} E[\pi R^2]$. The ellipsoid centers follow a Poisson distribution of mean Λ (number of trees per unit area) on the ground. We note $\mathbf{n}$ the ground normal and $\mathbf{u}_z$ the vertical. The content of a pixel $\mathcal{P}$ is divided in 4 parts: sunlit ground, shadowed

© 2011 The Author(s)
© 2011 The Eurographics Association and Blackwell Publishing Ltd.

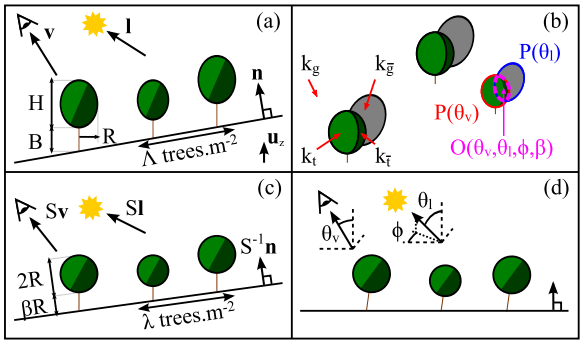

Figure 2: ***Strahler et al. model*** *(from [SJ90, SLS94]). The radiance of a forest patch, modeled with ellipsoids* (a)*, is computed as a weighted sum based on the proportions of sunlit ground k_g, shadowed ground $k_{\bar{g}}$, sunlit crowns k_t and shadowed crowns $k_{\bar{t}}$ visible in a pixel* (b)*. A vertical scaling* (c) *followed by a rotation* (d) *gives spherical trees on a horizontal ground without changing k_g, $k_{\bar{g}}$, k_t and $k_{\bar{t}}$.*

ground, sunlit crowns and shadowed crowns, noted respectively $\mathcal{P}_g$, $\mathcal{P}_{\bar{g}}$, $\mathcal{P}_t$ and $\mathcal{P}_{\bar{t}}$. The radiance of each part, noted L_g, $L_{\bar{g}}$, L_t and $L_{\bar{t}}$, are supposed *constant*, *i.e.,* independent of the view and light vectors $\mathbf{v}$ and $\mathbf{l}$. Thus, the radiance L of $\mathcal{P}$ only depends on the fractions of this pixel covered by each part, noted respectively k_g, $k_{\bar{g}}$, k_t, $k_{\bar{t}}$:

$$L = k_g L_g + k_{\bar{g}} L_{\bar{g}} + k_t L_t + k_{\bar{t}} L_{\bar{t}} \tag{1}$$

with $k_g + k_{\bar{g}} + k_t + k_{\bar{t}} = 1$. The problem is to compute these fractions, depending on $\mathbf{v}$, $\mathbf{l}$, $\mathbf{n}$, Λ, A, h and b.

For this, Strahler et al. proceed as follows. They first use a vertical scaling $\mathcal{S}$ of ratio $2/h$ to get spherical trees, followed by a rotation $\mathcal{R}$ to make the transformed normal $\mathcal{R}\mathcal{S}^{-1}\mathbf{n}$ vertical (see Fig. 2). These transformations do not change the $k_\star$ fractions and allow them to restrict their study to the case of spherical trees on a horizontal ground. More precisely this shows that the $k_\star(\mathbf{v},\mathbf{l},\mathbf{n},\Lambda,A,h,b)$ functions do not have 10 but only 6 independent parameters, namely θ_v, θ_l, ϕ, λ, A and β, defined in transformed space (see Fig. 2):

$$\cos\theta_v \stackrel{\text{def}}{=} u(\mathcal{S}\mathbf{v}) \cdot u(\mathcal{S}^{-1}\mathbf{n}), \quad \text{with } u(\mathbf{v}) = \frac{\mathbf{v}}{\|\mathbf{v}\|} \tag{2}$$

$$\cos\theta_l \stackrel{\text{def}}{=} u(\mathcal{S}\mathbf{l}) \cdot u(\mathcal{S}^{-1}\mathbf{n}) \tag{3}$$

$$\cos\phi \stackrel{\text{def}}{=} \frac{u(\mathcal{S}\mathbf{v}) \cdot u(\mathcal{S}\mathbf{l}) - \cos\theta_v \cos\theta_l}{\sin\theta_v \sin\theta_l} \tag{4}$$

$$\lambda \stackrel{\text{def}}{=} \Lambda \frac{u(\mathcal{S}^{-1}\mathbf{n}) \cdot \mathbf{u}_z}{\mathbf{n} \cdot \mathbf{u}_z} \tag{5}$$

$$\beta \stackrel{\text{def}}{=} 2\frac{b}{h} u(\mathcal{S}^{-1}\mathbf{n}) \cdot \mathbf{u}_z \tag{6}$$

Then Strahler et al. show that the proportion of ground which is visible, $k_g + k_{\bar{g}}$, is equal to $\exp(-\lambda P(\theta_v))$, where $P(\theta_v) = A \sec\theta_v$ is the average area of the projection of a scaled tree on the ground in view direction. Similarly, they show that the proportion of visible and sunlit ground, k_g, is $\exp(-\lambda[P(\theta_v) + P(\theta_l) - O(\theta_v,\theta_l,\phi,\beta)])$, where O is the average area of overlap between the view and sun projections of a tree on the ground (see Fig. 2). They propose an approximate analytical model to compute O in [SJ90]. Finally, they propose in [LS92] an approximate analytical model for the relative proportion of sunlit areas in visible crowns, $\frac{k_t}{k_t + k_{\bar{t}}}$. This is trivial for a single tree, but much more complex when trees occlude each other. Together with the constraint $k_g + k_{\bar{g}} + k_t + k_{\bar{t}} = 1$, this suffice to evaluate Eq. 1.

Symbol	Description
R, H, h	crown radius, height, and proportion H/R
A	average horizontal tree area $E[\pi R^2]$
Λ, λ	tree density on ground, in transformed space
$\mathbf{n}, \bar{\mathbf{n}}, \mathbf{u}_z$	ground normal, its average in $\mathcal{P}$, vertical
$\mathbf{v}, \mathbf{l}$	view and light directions
θ_v, θ_l, ϕ	view and light angles in transformed space
$\mathcal{P}_{g\|\bar{g}\|t\|\bar{t}}$	lit / $\bar{s}$hadowed **g**round / **t**ree parts in a pixel
$k_{g\|\bar{g}\|t\|\bar{t}}$	subpixel fractions corresponding to $\mathcal{P}_{g\|\bar{g}\|t\|\bar{t}}$
$L_{g\|\bar{g}\|t\|\bar{t}}$	constant Strahler et al. radiances for $\mathcal{P}_{g\|\bar{g}\|t\|\bar{t}}$
$\Lambda_h^i(\mathbf{x}), \Lambda_h(\mathbf{x})$	horizontal tree density for species i, total
$\Gamma(\mathbf{x})$	coverage of ground by trees in top view
$z, \bar{z}$	min, max depth z-field channels
α, δ	opacity, ambient occlusion z-field channels
$\mathbf{o}, \mathbf{p}_v, \mathbf{p}_l$	tree center, view and light entry points in tree
α_v, δ_v	opacity, ambient occlusion at $\mathbf{p}_v$
τ, p, ρ	foliage extinction, phase function, albedo
$V(\mathbf{p}_l)$	Sun visibility (computed with shadow maps)
$I_t(\mathbf{p}_v), J_t(\mathbf{p}_v)$	Sun and sky illumination in a tree at $\mathbf{p}_v$
$I_g(\mathbf{x}), J_g(\mathbf{x})$	Sun and sky illumination on the ground at $\mathbf{x}$
a_1, a_2, a_3	user parameters for $I_t(\mathbf{p}_v)$ and $J_g(\mathbf{x})$
$d(\mathbf{p}_v)$	distance of $\mathbf{p}_v$ from top of canopy
δ_e, δ_h	ambient occlusion from other trees, in a hole
δ_g	average ambient occlusion on ground
$\Delta(\mathbf{x})$	empiric contrast factor for δ_g
$r, \bar{r}$	ground BRDF, its average in $\mathcal{P}$
$\bar{I}_g, \bar{J}_g$	averages of $I_g(\mathbf{x})$ in $\mathcal{P}_g$, of $J_g(\mathbf{x})$ in $\mathcal{P}_g \cup \mathcal{P}_{\bar{g}}$
$\bar{I}_t, \bar{J}_t$	averages of $I_t(\mathbf{p}_v)$ in $\mathcal{P}_t$, of $J_t(\mathbf{p}_v)$ in $\mathcal{P}_t \cup \mathcal{P}_{\bar{t}}$
$\mathbb{G}, \mathbb{T}, \mathbb{D}, \mathbb{E}$	precomputed tables
$\tilde{I}_g, \tilde{J}_g, \tilde{I}_t, \tilde{J}_t$	transitions from I_g, J_g, I_t, J_t to $\bar{I}_g, \bar{J}_g, \bar{I}_t, \bar{J}_t$

Table 1: *Important symbols used in this paper.*

The Strahler et al. model reproduces a view and slope-dependent reflectance, and a hotspot at the scale of trees. But opaque ellipsoids cannot reproduce the effects due to lighting *inside* trees: hotspot due to leaves, silverlining. Also this model does not reproduce the sky illumination effects. We extend their model with arbitrary isotropic tree distributions, realistic tree shapes, varying radiances inside each $\mathcal{P}_\star$ part, semi-transparent models, and sky illumination. This allows us to take all the lighting effects into account, in a consistent way with our near tree representation (see Section 6).

© 2011 The Author(s)
© 2011 The Eurographics Association and Blackwell Publishing Ltd.

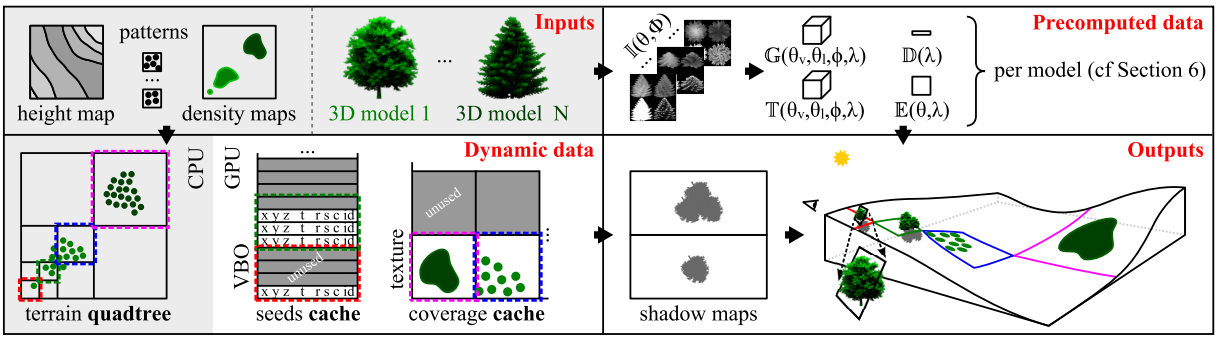

Figure 3: ***Overview***. *For each new visible terrain quad* (bottom-left) *we generate from the input maps* (top-left)*, in two caches, a coverage map tile* (purple and blue) *or, for the nearest quads* (red and green)*, tree seeds (position, size, etc). We use this, together with textures precomputed from 3D trees* (top-right)*, to render the nearest trees with parallax and lighting effects – with only one OpenGL quad per tree – and the farthest trees with a shader modulated by the coverage map tiles.*

4. Our Model

As shown in Section 2, current forest representations are not sufficient to achieve our goals. First, methods using 3D forest patch textures allow only limited user control, and parallax and lighting effects are difficult to simulate with billboards. Second, both types of methods do not scale to really far views (many trees projecting in a single pixel). We solve the first problem by drawing the nearest trees individually, with our new z-field tree representation accounting for parallax, and local and global lighting (although it uses only one quad per tree). We solve the second problem by drawing the farthest trees with our shader-map representation, which does not use any geometry. Indeed, we render the trees as tiny disks in a coverage map, applied on the terrain and rendered with a view and light dependent shader.

This section presents an overview of our algorithm. Our new representations are presented in Sections 5 and 6, and the transition between them in Section 7.

Inputs. Our algorithm requires a set of positions for the nearest trees, and a tree density map for the farthest ones. In this paper we chose to generate positions from the density map (the converse is also possible). For this we use an aperiodic tiling of point distribution patterns [CSHD03], and remove points where needed to match the desired density. Thus, the inputs of our algorithm are (see Fig. 3):

- a height map, *i.e.,* a digital terrain elevation model.
- a set of coarse tree density maps $\Lambda_h^i(\mathbf{x})$ describing the number of trees of each species i per unit *horizontal* area.
- a set of tileable point distribution patterns. Each point has a 2D coordinate and a random rotation, scale, color, etc.
- one or more 3D mesh models per tree species.

Data structures. In order to support very large scenes, we need to load on GPU only the data that is needed for the current view, at the proper resolution. For this, we rely on the framework of [BN08]. This method uses a dynamic quadtree on CPU, with GPU producers and caches to generate and store the terrain data for the currently visible quads. We extend it with new producers and caches for our forest data. More precisely, for each new visible terrain quad, and depending on its level in the quadtree (see below), we produce on GPU either a set of *seeds* to instantiate trees, or a *coverage map tile* for our shader-map representation (see Fig. 3):

- Each seed contains the 3D position of a tree, its species, and a rotation, scale, and color. We generate them from an aperiodic tiling of the input point distribution patterns.
- A coverage map tile has one density channel $\Lambda_h^i(\mathbf{x})$ per species, loaded from the input maps. It also has one channel $\Gamma(\mathbf{x})$ containing the percentage of ground covered by trees, in top view. This channel is generated by drawing one disk per tree or, to avoid drawing too many disks, as the total density $\Lambda_h(\mathbf{x}) = \sum_i \Lambda_h^i(\mathbf{x})$ times the average tree area A, when the disk area is less than $\frac{1}{16}$ of a texel.

Once produced on GPU, we store the seeds and the coverage map tiles in two caches (see Fig. 3), so that they can be reused as long as the quad remains visible. The first cache is a large vertex buffer object, divided in regions of equal size, each storing the seeds for a quad. The second cache is a texture array, each layer storing one coverage tile.

It remains to set the LOD threshold, *i.e.,* to decide when to generate seeds *vs* coverage map tiles. Since a coverage tile is mapped on the terrain it does not add any details to its silhouettes, unlike instantiated trees. Thus, to avoid popping at transitions, we should switch between the two representations when the apparent tree size is about one pixel. In our implementation we use 3 pixels, as a compromise to reduce the number of instantiated trees. In summary, if l is the smallest quadtree level at which the apparent tree size is larger than 3 pixels, then we generate coverage map tiles at all levels from 0 to $l-1$, and seeds at level l (the other quads reuse the data of their ancestor at level l).

© 2011 The Author(s)
© 2011 The Eurographics Association and Blackwell Publishing Ltd.

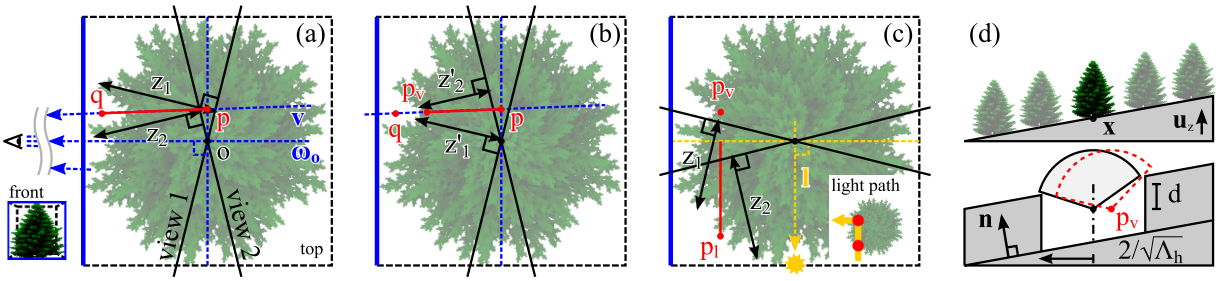

Figure 4: ***Tree rendering.*** *We render each near tree with a camera facing proxy quad* (blue)*. We reconstruct iteratively the intersection* $\mathbf{p}_v$ *of each view ray* $\mathbf{v}$ *with the tree* (a and b)*, and the corresponding entry point* $\mathbf{p}_l$ *of the light ray reaching* $\mathbf{p}_v$ (c)*, using precomputed depth maps. We estimate the direct sun illumination at* $\mathbf{p}_v$ *with an exponential extinction* $\exp(-\tau\|\mathbf{p}_l - \mathbf{p}_v\|)$*, and the direct sky illumination with the ambient occlusion precomputed for an isolated tree, times the ambient occlusion on the axis of a cylindrical hole approximating the environment* (d)*.*

Rendering. We render each frame in three steps, with the help of textures precomputed from the 3D tree models - see Fig. 3 and Sections 5 and 6. First, we render the terrain and the nearest trees, using the cached seeds and our z-field tree representation, into cascaded shadow maps [ZSXL06]. Second, we draw the terrain, using the shadow maps for the shadows of the nearest trees, and the cached coverage map tiles and our shader-map representation for the farthest trees. Third, we draw the nearest trees with our z-field representation, by using the cached seeds and the shadow maps. Note that this order enables early z-culling optimizations.

5. Nearest trees: representation and rendering

To support fine grained user control, we want to render the nearest trees individually. We also want to render them as efficiently as possible, while accounting for self-shadows, hotspot, silverlining and sky illumination effects. Finally, to enable transitions with 3D tree models for close views, we need accurate parallax effects on each detail. For this, we propose a new representation called a *z-field*. It is based on the precomputed depth maps representation of Max et al. [MO95], extended with improved rendering and lighting models inspired from [RB85, QNTN03, TRSK07].

We now describe our representation, its construction, and how we use it to reconstruct the tree shape, evaluate the sun and sky illumination, and the illumination on the ground. Note that we consider 3 levels of light interactions: between trees, between tree parts, and between leaves. All handle view-light correlations and thus hotspot effects.

Precomputations. We precompute a set of low resolution views for each input tree mesh (see Fig. 3). Each view contains 4 values per texel: the minimal and maximal depths z and $\bar{z}$, an ambient occlusion δ, and an opacity α (z, $\bar{z}$ and δ are α-premultiplied). Depths are computed as distances from the plane perpendicular to the view direction passing through the tree center $\mathbf{o}$. Ambient occlusion is precomputed for an isolated tree on a horizontal ground. In our implementation we use 181 view directions $\boldsymbol{\omega}_i$ uniformly distributed on the upper hemisphere Ω^+. Each view has 256×256 RGBA8 texels, allowing us to support apparent tree sizes up to 256 pixels without artefacts (using 45 MB per model).

Run-time shape reconstruction. Max et al. [MO95] render a tree by interpolating 3 or 4 precomputed depth views, but this gives ghosting artefacts. To solve this, we use an iterative method to reconstruct seamless 3D tree shapes, inspired from [TRSK07]. We found that, with 181 views, two iterations for view rays, and one for light rays, were sufficient. So we render each tree with a camera facing proxy quad covering its 3D bounding box, as follows (see Fig. 4):

1. Find the 3 precomputed view directions $\boldsymbol{\omega}_i$ closest to $\boldsymbol{\omega}_o$, and the weights w_i such that $\boldsymbol{\omega}_o \propto \sum w_i \boldsymbol{\omega}_i$ and $\sum w_i = 1$.
2. Find the intersection $\mathbf{p}$ of the view ray $\mathbf{v}$ with the plane passing through $\mathbf{o}$ and perpendicular to $\boldsymbol{\omega}_o$.
3. Iteration 1: project $\mathbf{p}$ orthogonally in the 3 nearest views, yielding 3 $(z_i, \bar{z}_i, \delta_i, \alpha_i)$ texels. Interpolate them to compute $\mathbf{q} = \mathbf{p} + (\sum w_i z_i)/(\sum w_i \alpha_i)\mathbf{v}$.
4. Iteration 2: project $\mathbf{q}$ in the 3 nearest views, yielding 3 $(z'_i, \bar{z}'_i, \delta'_i, \alpha'_i)$ texels. Set $\mathbf{p}_v = \mathbf{p} + (\sum w_i z'_i)/(\sum w_i \alpha'_i)\mathbf{v}$. Use $\mathbf{p}_v$ to set the fragment depth in the z-buffer.
5. Repeat steps 1 to 3 with the light direction $\mathbf{l}$ and the light ray passing through $\mathbf{p}_v$, to get $\mathbf{p}_l$.

This gives ghosting-free view and light rays entry points $\mathbf{p}_v$ and $\mathbf{p}_l$, as well as the opacity and ambient occlusion at $\mathbf{p}_v$, $\alpha_v = \sum w_i \alpha'_i$ and $\delta_v = \sum w_i \delta'_i / \alpha_v$, used for lighting.

Lighting from the sun. We model the foliage as a participating medium with an extinction coefficient τ, a phase function p and an albedo ρ (in our implementation we use an isotropic phase function). We also assume that light scattering occurs mostly at the tree "surface" ($\tau R \gg 1$) so that we only consider the light path from $\mathbf{p}_v$ to $\mathbf{p}_l$. The sun light I_t scattered at $\mathbf{p}_v$ is thus $\rho p(\mathbf{v}, \mathbf{l}) \exp(-\tau\|\mathbf{p}_l - \mathbf{p}_v\|) L_{sun}$. This gives silverlining and self-shadows (and thus a hotspot) at the scale of groups of leaves. To account for self-shadows at

© 2011 The Author(s)
© 2011 The Eurographics Association and Blackwell Publishing Ltd.

the scale of leaves, we add an empiric hotspot term F [CC97] whose amplitude a_1 and width a_2^{-1} are set by the user:

$$I_t(\mathbf{p}_v) = \rho p(\mathbf{v},\mathbf{l}) \exp(-\tau\|\mathbf{p}_l - \mathbf{p}_v\|) F(\mathbf{v},\mathbf{l}) V(\mathbf{p}_l) L_{sun} \quad (7)$$

$$F(\mathbf{v},\mathbf{l}) \stackrel{\text{def}}{=} 1 - a_1[1 - \exp(-a_2 \cos^{-1}(\mathbf{v}\cdot\mathbf{l}))] \quad (8)$$

with $V(\mathbf{p}_l)$ the sun visibility, computed with cascaded shadow maps [ZSXL06] – we render these maps with step 5 only, and with $\bar{z}$ instead of z to avoid shadow acne (*i.e.,* we render the "backside" of trees). We also store the opacity α in these maps, to get "soft" shadows [BBP08].

Lighting from the sky. We approximate the sky radiance J_t reflected at $\mathbf{p}_v$ with the average sky radiance L_{sky} times an ambient occlusion $\delta(\mathbf{p}_v)$. Assuming that inter and intra tree occlusions are uncorrelated, we get $\delta = \delta_v\delta_e$, where δ_e is the ambient occlusion due to other trees. We approximate δ_e with the ambient occlusion δ_h on the axis of a cylindrical hole around the tree (see Fig. 4d), of radius $2/\sqrt{\Lambda_h(\mathbf{x})}$ – the average distance to the nearest tree. Finally, we approximate δ_h with the ambient occlusion in a hole of depth d and radius r in a horizontal ground, $1/(1+d^2r^{-2})$, times the one on a slanted plane, $(1+\mathbf{n}\cdot\mathbf{u}_z)/2$. The result is a rough approximation, but sufficient to get "sky illumination" effects:

$$J_t(\mathbf{p}_v) = \frac{\rho}{2}\left(\int_{\Omega^+} p(\mathbf{v},\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\right) \frac{\delta_v(\mathbf{p}_v)(1+\mathbf{n}\cdot\mathbf{u}_z)}{1+d^2(\mathbf{p}_v)\Lambda_h(\mathbf{x})/4} L_{sky} \quad (9)$$

Lighting on the ground. To get darker ground areas under trees, we modulate the average ambient occlusion due to trees on the ground, noted δ_g (see Section 6) with an empiric term $\Delta(\mathbf{x})$ increasing the contrast close to trees. $\Delta(\mathbf{x})$ must be less than 1 under trees, but its average must be 1. For this, we compute it from the coverage map tiles with:

$$\Delta(\mathbf{x}) \stackrel{\text{def}}{=} 1 - a_3\left[\Gamma(\mathbf{x}) - \Lambda_h(\mathbf{x})A\right] \quad (10)$$

where a_3 is a user defined contrast factor. Thus, if r is the ground BRDF, we compute the sun and sky light reflected by the ground, respectively I_g and J_g, with:

$$I_g(\mathbf{x}) = r(\mathbf{v},\mathbf{l},\mathbf{n})\ \max(\mathbf{n}\cdot\mathbf{l},0)\ V(\mathbf{x})\ L_{sun} \quad (11)$$

$$J_g(\mathbf{x}) = \delta_g\ \Delta(\mathbf{x}) \int_{\Omega^+} r(\mathbf{v},\boldsymbol{\omega},\mathbf{u}_z)\ \mathbf{u}_z\cdot\boldsymbol{\omega}\ \mathrm{d}\boldsymbol{\omega}\ L_{sky} \quad (12)$$

6. Farthest trees: representation and rendering

To ensure scalability in terms of memory and performance, we render the farthest trees with a *shader-map* representation, *i.e.,* shaders modulated by a terrain map. The terrain map stores the proportion of tree *vs* ground in top view (we build it by rendering the trees as tiny disks). From this, we reconstruct each screen pixel color with a forest shader computing the 4 view-light dependent pixel fractions $k_g, k_{\bar{g}}, k_t, k_{\bar{t}}$ and the corresponding radiances $L_g, L_{\bar{g}}, L_t, L_{\bar{t}}$, in the spirit of Strahler et al. (see Section 3 and Fig. 5). In fact, we revisit and extend their model to take into account arbitrary isotropic tree distributions, realistic tree shapes, intra-tree lighting and sky illumination, as follows.

Pixel radiance. Strahler et al. rely on constant values for the radiances of sunlit or shadowed ground and trees, which is not sufficient for Computer Graphics applications. Moreover, we need to ensure a consistent transition with our detailed lighting model used for near trees, which is view and light dependent. For this, we introduce

- $\bar{I}_g$, average of $I_g(\mathbf{x})$ over the visible and sunlit ground $\mathcal{P}_g$.
- $\bar{J}_g$, average of $J_g(\mathbf{x})$ over the visible ground $\mathcal{P}_g \cup \mathcal{P}_{\bar{g}}$.
- $\bar{I}_t$, average of $I_g(\mathbf{p}_v)$ over the visible and sunlit trees $\mathcal{P}_t$.
- $\bar{J}_t$, average of $J_g(\mathbf{p}_v)$ over the visible trees $\mathcal{P}_t \cup \mathcal{P}_{\bar{t}}$.

and we redefine $L_g = \bar{I}_g + \bar{J}_g$, $L_{\bar{g}} = \bar{J}_g$, and similarly for trees. Then the reconstructed pixel radiance in Eq. 1 becomes

$$L = k_g\bar{I}_g + (k_g + k_{\bar{g}})\bar{J}_g + (1 - k_g - k_{\bar{g}})\left[\frac{k_t}{k_t + k_{\bar{t}}}\bar{I}_t + \bar{J}_t\right] \quad (13)$$

We now detail how we compute $k_g, k_{\bar{g}}, k_t, k_{\bar{t}}$ and $\bar{I}_g, \bar{J}_g, \bar{I}_t, \bar{J}_t$, depending on $\mathbf{v}$, $\mathbf{l}$, $\mathbf{n}$, $\Lambda_h(\mathbf{x})$, τ, p, ρ, r, etc. *We assume here that many trees project in a pixel*. The case of apparent tree sizes of about 1 pixel is discussed in Section 7.

Ground fractions. Strahler et al. compute k_g and $k_{\bar{g}}$ by using the fact that random 2D shapes (here the projection of trees on the ground), of average area P, distributed with a Poisson law of mean λ, leave a fraction $f = \exp(-\lambda P)$ of the plane uncovered. But this is no longer true with other tree distributions. For instance, $f = 1 - \lambda P$ with a Poisson-disk distribution, if the shapes are contained in the disks. Also, f generally depends on the precise shapes considered. Thus, to support arbitrary isotropic tree distributions and realistic tree shapes, we replace their analytic model with a precomputed table. For this, we render a forest patch with our z-field representation, scaled vertically by a factor $2/h$, and we measure k_g for many values of θ_v, θ_l, ϕ and λ. We store the result in a 4D table $\mathbb{G}$. At runtime, we compute

$$k_g = \mathbb{G}(\theta_v, \theta_l, \phi, \lambda) \quad (14)$$

$$k_g + k_{\bar{g}} = \mathbb{G}(\theta_v, \theta_v, 0, \lambda) \quad (15)$$

where $\lambda = \Lambda_h(\mathbf{x})\ u(\mathcal{S}^{-1}\mathbf{n})\cdot\mathbf{u}_z$ (cf Eq. 5).

Tree fractions. Strahler et al. compute the relative proportion of sunlit areas in visible crowns, $\frac{k_t}{k_t+k_{\bar{t}}}$, by using a binary test (sunlit or not), only valid for opaque objects. To account for self-shadows inside trees and silverlining, we replace this binary test with our exponential attenuation term $\exp(-\tau\|\mathbf{p}_l - \mathbf{p}_v\|)$. Then, as above, we measure $\frac{k_t}{k_t+k_{\bar{t}}}$ for many views and tree densities, and store the result in a 4D table $\mathbb{T}$. At runtime, we compute

$$\frac{k_t}{k_t + k_{\bar{t}}} = \mathbb{T}(\theta_v, \theta_l, \phi, \lambda) \quad (16)$$

Ground radiances. The sun component of the visible and sunlit ground radiance, $\bar{I}_g$, is the average of Eq. 11 over $\mathcal{P}_g$, where $V(\mathbf{x}) = 1$. Assuming that the ground BRDF r is uncorrelated with the tree distribution, so that its averages over $\mathcal{P}_g$ or over the whole pixel $\mathcal{P}$ are the same, we

© 2011 The Author(s)
© 2011 The Eurographics Association and Blackwell Publishing Ltd.

get $\bar{I}_g = \int_{\mathcal{P}} r(\mathbf{v},\mathbf{l},\mathbf{n})\ \max(\mathbf{n}\cdot\mathbf{l},0)\ \mathrm{d}\mathbf{x}\ L_{sun}$. For a Lambertian ground this can be approximated with:

$$\bar{I}_g \approx \bar{r}\max(\bar{\mathbf{n}}\cdot\mathbf{l},0)L_{sun} \tag{17}$$

with $\bar{r}$ and $\bar{\mathbf{n}}$ the average ground reflectance and normals in the pixel.

The sky component of the visible ground radiance, $\bar{J}_g$, is the average of Eq. 12 over $\mathcal{P}_g \cup \mathcal{P}_{\bar{g}}$. To compute it, we first need to compute the ambient occlusion δ_g. We approximate it as the average ambient occlusion due to trees on a horizontal ground, times the ambient occlusion on a slanted ground, $(1+\mathbf{n}\cdot\mathbf{u}_z)/2$, which gives:

$$\delta_g = \frac{1+\bar{\mathbf{n}}\cdot\mathbf{u}_z}{2}\frac{1}{\pi}\int_{\Omega^+}\mathbb{G}(\theta_v,\theta_v,0,\lambda)\mathbf{v}\cdot\mathbf{u}_z\mathrm{d}\mathbf{v} \tag{18}$$

We precompute the integral and store it in a 1D table $\mathbb{D}(\lambda)$. Then, returning to the remaining terms in Eq. 12, we need to compute the average of $\Delta(\mathbf{x})$ over $\mathcal{P}_g \cup \mathcal{P}_{\bar{g}}$. We approximate it with its average over the whole pixel, 1 by construction. We finally replace r with its average $\bar{r}$, which gives for a Lambertian ground:

$$\bar{J}_g \approx \frac{1+\bar{\mathbf{n}}\cdot\mathbf{u}_z}{2}\mathbb{D}(\lambda)\bar{r}\pi L_{sky} \tag{19}$$

Tree radiances. The sun component of the visible and sunlit trees radiance, $\bar{I}_t$, is the average of Eq. 7 over $\mathcal{P}_t$, where $V(\mathbf{p}_l)=1$. The only other term depending on $\mathbf{p}_v$ in this equation is our exponential attenuation term, but it is already taken into account in $\mathbb{T}$. The other terms are constant over $\mathcal{P}_t$, which gives:

$$\bar{I}_t = \rho p(\mathbf{v},\mathbf{l})F(\mathbf{v},\mathbf{l})L_{sun} \tag{20}$$

Finally, the sky component of the visible trees radiance, $\bar{J}_t$, is the average of Eq. 9 over $\mathcal{P}_t \cup \mathcal{P}_{\bar{t}}$. The only term depending on $\mathbf{p}_v$ in this equation is $\delta_v/(1+d^2\Lambda_h/4)$. We precompute its average on the visible tree areas by rendering trees shaded with this term, as we do for $\mathbb{G}$ and $\mathbb{T}$. We store the result in a 2D table $\mathbb{E}(\theta_v,\lambda)$. At runtime, we compute

$$\bar{J}_t = \rho\left(\int_{\Omega^+} p(\mathbf{v},\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\right)\frac{1+\bar{\mathbf{n}}\cdot\mathbf{u}_z}{2}\ \mathbb{E}(\theta_v,\lambda)\ L_{sky} \tag{21}$$

7. Seamless transition between representations

A sudden switch from our z-field to our shader-map representation would be very noticeable, for several reasons. First, at large distances, the visual fidelity of our z-field representation decreases: the length $\|\mathbf{p}_l - \mathbf{p}_v\|$ computed with our iterative algorithm becomes very imprecise, because then it is computed on very coarse MIP-mapped depth maps. Also, the exponential of this "average" length is not what we want, *i.e.,* the average of the exponential over the visible tree pixels. The cascaded shadow maps become also imprecise at this distance, as well as the occlusion effects. Second, although tree locations fit precisely between the 2 models, our 4 color components (tree *vs* ground, lit *vs* unlit) are computed on a statistical basis and thus cannot match a given tree instance. This would give a visible color discontinuity at the transition. Third, our shader-map model neglects forest thickness. Even if we use it when trees are at most 3 pixels tall, this would yield popping on the terrain silhouettes. To solve these problems, we propose a seamless transition scheme divided in three parts: one transition inside each representation, and a transition between them.

Transition in z-field representation. Although the radiances $I_t(\mathbf{p}_v), J_t(\mathbf{p}_v), I_g(\mathbf{x})$ and $J_g(\mathbf{x})$ become imprecise in the distance, we know the values toward which they should converge. Indeed, we computed them in Section 6. Thus, to solve the problem, we simply force a transition of these values toward their expected average. Concretely, if s is the distance to the viewer, and s_{max} the maximal distance at which the z-field representation is used, then we render the trees and the ground with:

$$\tilde{I}_t(\mathbf{p}_v) \stackrel{\text{def}}{=} (1-\nu)I_t(\mathbf{p}_v) + \nu\bar{I}_t\frac{k_t}{k_t+k_{\bar{t}}} \tag{22}$$

$$\tilde{J}_t(\mathbf{p}_v) \stackrel{\text{def}}{=} (1-\nu)J_t(\mathbf{p}_v) + \nu\bar{J}_t \tag{23}$$

$$\tilde{I}_g(\mathbf{x}) \stackrel{\text{def}}{=} (1-\nu)I_g(\mathbf{x}) + \nu\bar{I}_g\frac{k_g}{k_g+k_{\bar{g}}} \tag{24}$$

$$\tilde{J}_g(\mathbf{x}) \stackrel{\text{def}}{=} (1-\nu)J_g(\mathbf{x}) + \nu\bar{J}_g \tag{25}$$

where $\nu = \text{smoothstep}(0, 0.8, s/s_{max})$.

Transition in shader-map representation. Up to now, we have only used the coarse densities $\Lambda_h(\mathbf{x})$ in our forest model components $k_g, k_{\bar{g}}, k_t, k_{\bar{t}}$. This gives only an average forest radiance which cannot match the spatial variations obtained with our z-field model at the transition. Instead, we want the tree disks in the coverage channel $\Gamma(\mathbf{x})$ shaded only with the tree radiance components, and the ground between them only with the ground components. That is, we want to have $k_g = k_{\bar{g}} = 0$ when $\Gamma(\mathbf{x}) = 1$. For this, we simply replace $\Lambda_h(\mathbf{x})$ with $\Gamma(\mathbf{x})/A$ in Eqs. 14 and 15. Indeed, this changes nothing for really far trees (where $\Gamma(\mathbf{x}) = \Lambda_h(\mathbf{x})A$), but gives $k_g = k_{\bar{g}} = 0$ inside distinguishable disks, as desired (provided we ensure that $\mathbb{G}(\cdot,\cdot,\cdot,1/A) = 0$).

Transition between representations. To avoid popping on the terrain silhouettes, we progressively fade out the trees rendered with our z-field tree representation. For this, we multiply the opacity α_v with $1-\text{smoothstep}(0.8, 1, s/s_{max})$, while replacing the ground radiance in this transition region with our forest radiance model, modified as follows:

$$L = (1-\mu)\left[\frac{k_g}{k_g+k_{\bar{g}}}\bar{I}_g + \bar{J}_g\right] + \mu\left[\frac{k_t}{k_t+k_{\bar{t}}}\bar{I}_t + \bar{J}_t\right] \tag{26}$$

$$\mu = (1-k_g-k_{\bar{g}})\ \text{smoothstep}(0.8, 1, s/s_{max}) \tag{27}$$

© 2011 The Author(s)
© 2011 The Eurographics Association and Blackwell Publishing Ltd.

8. Implementation

Implementing our algorithm is relatively easy, thanks to the features of modern GPUs such as *geometry shaders*, *transform feedback* and user defined *subpixel coverage masks*.

The tree seeds producer must remove points from a precomputed stream of points, and must store the result in a VBO cache. We implement this in one pass with a geometry shader and a transform feedback. The coverage map tiles producer must draw tiny disks in a texture. For this, we use a geometry shader to generate sprites from the tree seeds. We draw the cascaded shadow maps in one pass with a geometry shader to select the shadow map layer(s) for each tree. Finally, we also use a geometry shader to cull the trees outside the view frustum, and to generate camera facing proxy quads for the others.

In general, rendering transparent objects requires sorting, which is costly and not always possible. One way to avoid this is to replace opacities with *subpixel coverage masks*. The OpenGL *alpha to coverage* feature does this automatically. But it always converts a given alpha at a given pixel into the same coverage mask. When n trees project in a pixel with a similar α the resulting opacity is thus α instead of $1-(1-\alpha)^n$ — since the tree positions are uncorrelated. This underestimates opacity and introduces a discontinuity at the transition between our two representations. To solve this, we enforce a different coverage mask for each tree. We precompute a *combination table* associating several possible coverage masks for each α value, and we use the tree seed to select one combination at rendering.

9. Results and validation

Results. Our results are shown in Figs. 1, 5 and 9. The components of our lighting model are shown in Fig. 5, and the seamless transition between our two models in Fig. 6. See also the companion video.

Performance. Given an input mesh of a tree and its ambient occlusion, it takes a few seconds to compute the 181 views, and about 10 minutes to precompute the $\mathbb{G},\mathbb{T},\mathbb{D},\mathbb{E}$ tables with a NVidia Geforce 470 GTX (we use 8 samples for λ, and 16 for each θ_v,θ_l,ϕ angle). With this GPU, a typical 1024×768 frame with about 180,000 z-field trees is rendered in 30 ms (33 fps), including 7 ms for the terrain, 0.6 for the shader-map, 4.4 for the shadow maps, and 18 for the z-field trees (respectively 7, 0.6, 1.4 and 10.6 with a 580 GTX – 51 fps).

Validation. We do not target "exact" lighting, so we did not compare our results quantitatively with ground truth images. Instead, we did qualitative comparisons. First with photos: Fig. 9 shows that our method can reproduce all the lighting effects presented in introduction. Second, with the view-dependent plots of a completely different model based on radiative transfer theory (see Fig. 8). Another goal was to get a consistent result at all scales. We validated this by measuring the radiance of a forest patch rendered with our method at several distances and for many view angles (see Fig. 7).

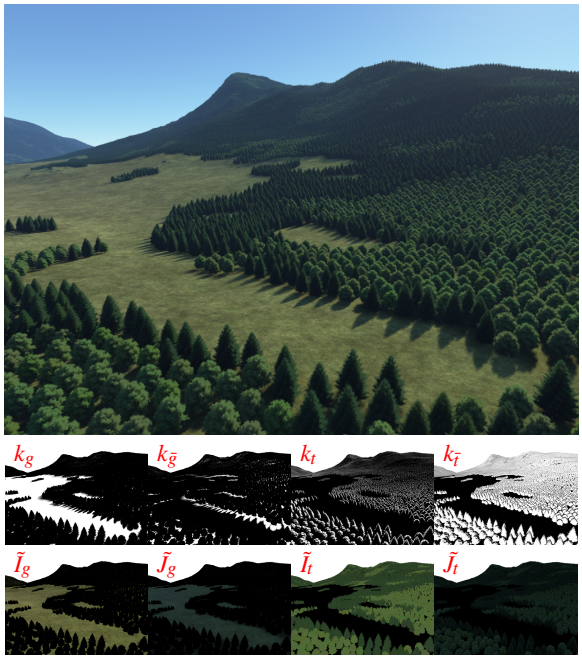

Figure 5: ***Results.*** *The* 8 *components of our lighting model.*

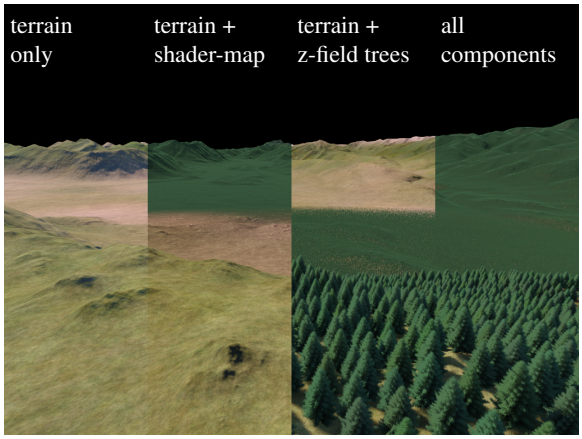

Figure 6: ***Results.*** *Seamless transition between our models.*

10. Discussion

A limitation of our method is the size of the z-field data: 45 MB per tree model. The number of models that can be used simultaneously in a given view is thus limited. For the same reason, trees cannot be animated to move in the wind. On the other hand, adding a normal per texel in each precomputed view is feasible (each model would then take 79 MB).

© 2011 The Author(s)
© 2011 The Eurographics Association and Blackwell Publishing Ltd.

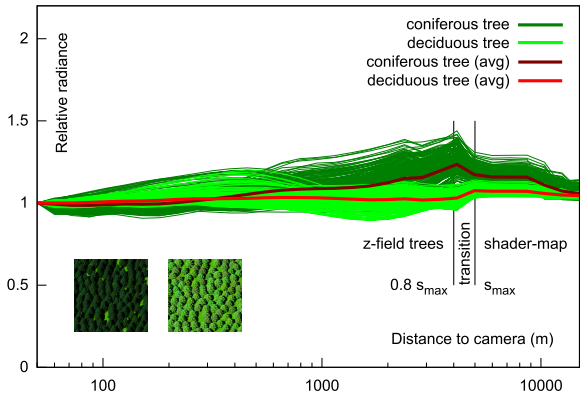

Figure 7: ***Validation of scale consistency.*** *The radiance of a forest patch rendered with our method (*insets*), as a function of the view distance s. Each curve shows the ratio* $L(\theta_v,\phi,s)/L(\theta_v,\phi,50)$ *for a different view direction, where* $L(\theta_v,\phi,s)$ *is the radiance at distance s (*$\theta_l = 45$*). Our results are close to* 1*, the ideal case where L is independent of s.*

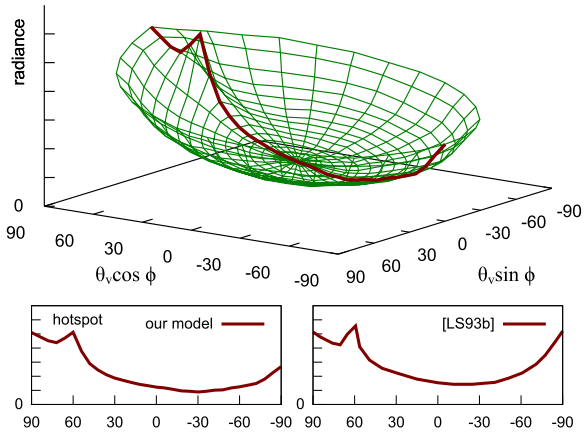

Figure 8: ***Validation with reference plots.*** *The radiance of a forest patch rendered with our method, as a function of the view direction (*$\theta_l = 60$*).* Bottom: *with* $r = 0$ *as in [LS93b],* $a_1 = 0.2$ *and* $a_2 = 20$*, our results are quite similar to theirs.*

This would enable the use of more realistic leaf BRDFs, with specular reflections, for the nearest z-field trees.

The tables $\mathbb{G},\mathbb{T},\mathbb{D},\mathbb{E}$ are precomputed for a given tree model, tree aspect ratio h, tree distribution law, and foliage density τ. But they are so small (131 kB in total) that we can easily use several versions of them to support spatially varying tree distribution laws or tree foliage densities (spatially varying tree colors are trivial since ρ and p are not used in any precomputed data). Likewise, it is easy to extend our method with a spatially varying tree aspect ratio h: it suffice to add this parameter to the 1D and 2D tables $\mathbb{D}$ and $\mathbb{E}$ (the 4D tables $\mathbb{G}$ and $\mathbb{T}$ remain unchanged since they are computed on rescaled trees). It should also be possible to support anisotropic tree distributions (*e.g.,* forests with aligned trees) by adding one or two angle parameters to each table.

11. Conclusion

We presented a method to render large forest scenes in real-time, with a realistic lighting model, consistent at all scales. Comparisons with photos show that our method can reproduce the main lighting effects observed in real forest scenes. In future work, we would like to implement seamless transitions with 3D mesh models for close views, currently not handled. We would also like to study the applicability of our model to other kinds of scene (rocks, grass, etc).

Acknowledgments This work is funded by the ANR 2010 JCJC 0207 01 "SimOne" project. We thank Laurence Boissieux for the 3D tree models and Charles de Rousiers for proofreading this paper.

© 2011 The Author(s)
© 2011 The Eurographics Association and Blackwell Publishing Ltd.

References

[AMM07] Yacine Amara, Sylvain Meunier, and Xavier Marsault. A GPU framework for the visualization and on-the-fly amplification of real terrains. In *Proceedings of the 3rd international conference on Advances in visual computing - Volume Part I*, ISVC'07, pages 586–597, 2007. 2

[BBP08] Kévin Boulanger, Kadi Bouatouch, and Sumant Pattanaik. Rendering trees with indirect lighting in real time. In *Rendering Techniques (Eurographics Symposium on Rendering - EGSR)*, 2008. 2, 6

[BCF+05] Stephan Behrendt, Carsten Colditz, Oliver Franzke, Johannes Kopf, and Oliver Deussen. Realistic real-time rendering of landscapes using billboard clouds. *Computer Graphics Forum (Proceedings of Eurographics'05)*, 24(3):507–516, 2005. 2

[BMG06] Frédéric Boudon, Alexandre Meyer, and Christophe Godin. Survey on Computer Representations of Trees for Realistic and Efficient Rendering. Technical Report RR-LIRIS-2006-003, LIRIS UMR 5205 CNRS/INSA de Lyon/Université Claude Bernard Lyon 1/Université Lumière Lyon 2/École Centrale de Lyon, February 2006. 2

[BN08] Eric Bruneton and Fabrice Neyret. Real-time rendering and editing of vector-based terrains. *Computer Graphics Forum (Proceedings of Eurographics'08)*, 27(2):311–320, 2008. 4

[CC97] J. M. Chen and J. Cihlar. A hotspot function in a simple bidirectional reflectance model for satellite applications. *Journal of Geophysical Research*, 102:25907–25914, November 1997. 6

[CCDH05] Carsten Colditz, Liviu Coconu, Oliver Deussen, and Christian Hege. Realtime rendering of complex photorealistic landscapes using hybrid level-of-detail approaches. In *Trends in Real-Time Landscape Visualization and Participation*, pages 97–106, 2005. 2

[CL97] J.M. Chen and S.G. Leblanc. A four-scale bidirectional reflectance model based on canopy architecture. *IEEE Transactions on Geoscience and Remote Sensing*, 35(5):1316 –1337, sep 1997. 2

[CSHD03] Michael F. Cohen, Jonathan Shade, Stefan Hiller, and

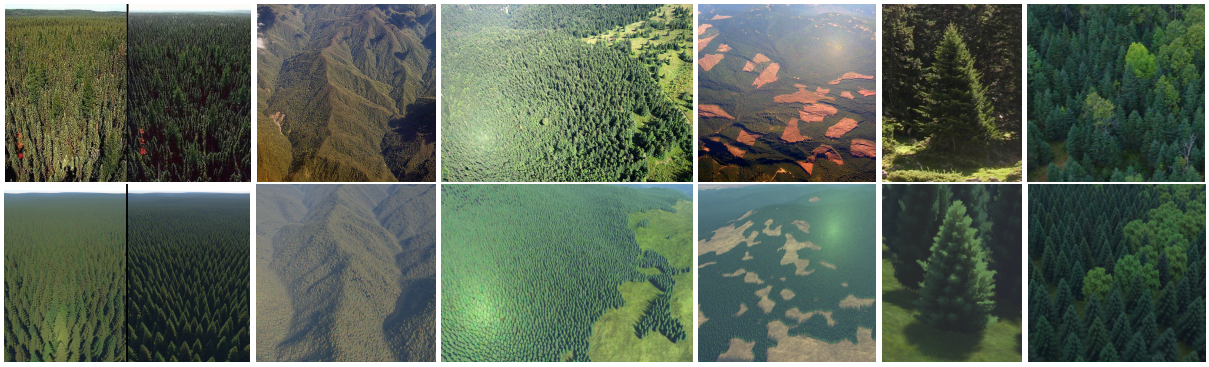

Figure 9: ***Results.*** *Comparisons between photos (*top*) and our results (*bottom*). From left to right: view-dependent reflectance, slope-dependent reflectance, hotspot effect (medium and far view), silverlining and "sky illumination" (overcast sky).*

Oliver Deussen. Wang tiles for image and texture generation. In *ACM Transactions on Graphics (Proceedings of SIGGRAPH'03)*, pages 287–294, 2003. 4

[DCSD02] Oliver Deussen, Carsten Colditz, Marc Stamminger, and George Drettakis. Interactive visualization of complex plant ecosystems. In *Proceedings of the conference on Visualization (VIS'02)*, pages 219–226, 2002. 2

[DN04] Philippe Decaudin and Fabrice Neyret. Rendering forest scenes in real-time. In *Rendering Techniques (Eurographics Symposium on Rendering - EGSR)*, pages 93–102, june 2004. 2

[FMU05] A Fuhrmann, S Mantler, and E Umlauf. Extreme model simplification for forest rendering. *in Proceedings of EGWNP - Eurographics Workshop on Natural Phenomena*, 2005. 2

[GCRR08] Jesus Gumbau, Miguel Chover, Cristina Rebollo, and Inmaculada Remolar. Real-time illumination of foliage using depth maps. In *Proceedings of the 8th international conference on Computational Science, Part II*, ICCS '08, pages 136–145, 2008. 2

[GMN05] Guillaume Gilet, Alexandre Meyer, and Fabrice Neyret. Point-based rendering of trees. In *Eurographics Workshop on Natural Phenomena*, 2005. 2

[Gol97] Dan B. Goldman. Fake fur rendering. In *Computer Graphics (Proceedings of ACM SIGGRAPH'97)*, pages 127–134, 1997. 2

[GS08] R. Geist and J. Steele. A lighting model for fast rendering of forest ecosystems. In *IEEE Symposium on Interactive Ray Tracing (RT'08)*, pages 99 –106, aug. 2008. 2

[HPAD06] Kyle Hegeman, Simon Premože, Michael Ashikhmin, and George Drettakis. Approximate ambient occlusion for trees. In *Symposium on Interactive 3D Graphics and Games (I3D '06)*, pages 87–92, 2006. 2

[LS85] Xiaowen Li and A.H. Strahler. Geometric-optical modeling of a conifer forest canopy. *IEEE Transactions on Geoscience and Remote Sensing*, GE-23(5):705 –721, sept. 1985. 2

[LS92] X. Li and A.H. Strahler. Geometric-optical bidirectional reflectance modeling of the discrete crown vegetation canopy: effect of crown shape and mutual shadowing. *IEEE Transactions on Geoscience and Remote Sensing*, 30(2):276 –292, mar 1992. 2, 3

[LS93a] S. Liang and A.H. Strahler. An analytic brdf model of canopy radiative transfer and its inversion. *IEEE Transactions on Geoscience and Remote Sensing*, 31(5):1081 –1092, sep 1993. 2

[LS93b] S. Liang and A.H. Strahler. Calculation of the angular radiance distribution for a coupled atmosphere and canopy. *IEEE Transactions on Geoscience and Remote Sensing*, 31(2):491 – 502, mar 1993. 9

[MJ06] Stephan Mantler and Stefan Jeschke. Interactive landscape visualization using GPU ray casting. In *Proceedings of Graphite 2006*, pages 2

[MN00] Alexandre Meyer and Fabrice Neyret. Multiscale shaders for the efficient realistic rendering of pine-trees. In *Graphics Interface*, pages 137–144, May 2000. 2

[MNP01] Alexandre Meyer, Fabrice Neyret, and Pierre Poulin. Interactive rendering of trees with shading and shadows. In *Rendering Techniques (Eurographics Workshop on Rendering - EGSR)*, Jul 2001. 2

[MO95] Nelson Max and Keiichi Ohsaki. Rendering trees from precomputed z-buffer views. In *Eurographics Rendering Workshop*, pages 45–54, 1995. 2, 5

[QNTN03] Xueying Qin, Eihachiro Nakamae, Katsumi Tadamura, and Yasuo Nagai. Fast photo-realistic rendering of trees in daylight. *Computer Graphics Forum*, 22(3):243–252, 2003. 2, 5

[RB85] William T. Reeves and Ricki Blau. Approximate and probabilistic algorithms for shading and rendering structured particle systems. *Computer Graphics (Proceedings of ACM SIGGRAPH'85)*, 19:313–322, July 1985. 2, 5

[SJ90] Alan H. Strahler and David L.B. Jupp. Modeling bidirectional reflectance of forests and woodlands using boolean models and geometric optics. *Remote Sensing of Environment*, 34(3):153 – 166, 1990. 2, 3

[SLS94] C.B. Schaaf, Xiaowen Li, and A.H. Strahler. Topographic effects on bidirectional and hemispherical reflectances calculated with a geometric-optical canopy model. *IEEE Transactions on Geoscience and Remote Sensing*, 32(6):1186 –1193, nov 1994. 2, 3

[TRSK07] S. Todt, C. Rezk-Salama, and A. Kolb. Fast (Spherical) Light Field Rendering with Per-Pixel Depth. Technical report, University of Siegen, Germany, 2007. 5

[ZSXL06] Fan Zhang, Hanqiu Sun, Leilei Xu, and Lee Kit Lun. Parallel-split shadow maps for large-scale virtual environments. In *ACM international conference on Virtual reality continuum and its applications (VRCIA'06)*, pages 311–318, 2006. 5, 6

© 2011 The Author(s)
© 2011 The Eurographics Association and Blackwell Publishing Ltd.