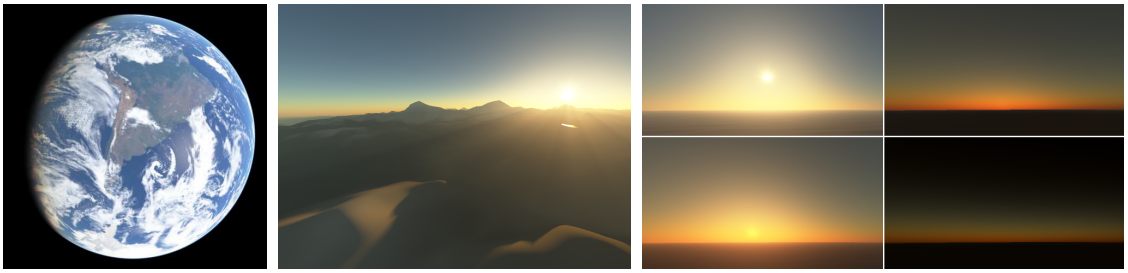
Precomputed Atmospheric Scattering

Eric Bruneton, Fabrice Neyret

Eurographics Symposium on Rendering 2008
Steve Marschner and Michael Wimmer
(Guest Editors)

Volume 27 (2008), Number 4

Precomputed Atmospheric Scattering



Eric Bruneton and Fabrice Neyret

EVASION – LJK / Grenoble Universités – INRIA

Abstract

We present a new and accurate method to render the atmosphere in real time from any viewpoint from ground level to outer space, while taking Rayleigh and Mie multiple scattering into account. Our method reproduces many effects of the scattering of light, such as the daylight and twilight sky color and aerial perspective for all view and light directions, or the Earth and mountain shadows (light shafts) inside the atmosphere. Our method is based on a formulation of the light transport equation that is precomputable for all view points, view directions and sun directions. We show how to store this data compactly and propose a GPU compliant algorithm to precompute it in a few seconds. This precomputed data allows us to evaluate at runtime the light transport equation in constant time, without any sampling, while taking into account the ground for shadows and light shafts.

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

1. Introduction

Atmospheric effects are very important to increase the realism of outdoor scenes in many applications. The sky color gives key indications about the hour of the day, and the aerial perspective gives an important cue to evaluate distances. Rendering these effects in real time, continuously from ground to space, is desirable in many games or applications, such as flight simulators or Earth browsers like Google Earth. This is especially true for applications that target realism, such as Celestia or Nasa WorldWind. However, these applications currently use very basic models to render these effects, which do not give realistic images.

In this paper we propose a method to render these effects in real time, from any viewpoint from ground to space. This method accounts for multiple scattering, which is important to correctly render twilight, or the shadow of the Earth inside the atmosphere (see Figure 8). It is based on moderate simplifying assumptions that allow us to get a better approximate solution of the rendering equation (compared to previous work), in which most terms can be precomputed. Our method is the first real-time method accounting for all viewpoints, all view and sun directions, and multiple scattering.

The next sections are organized as follows. Section 2 introduces the physical model and the rendering equation and reviews the related work. Section 3 presents our resolution method to get a precomputable formulation. Sections 4 and 5 present our precomputation and rendering algorithms. Section 6 gives implementation details and presents our results.

2. Atmospheric models

Rendering atmosphere illumination relies on two aspects: a physical model of the local medium properties, and a simulation of the global illumination exchanges up to the viewer eyes. This includes exchanges with the ground, which can be modeled as a Lambertian surface with a height field of reflectance $\alpha(\mathbf{x},\lambda)$, normal $\mathbf{n}(\mathbf{x})$, etc.

Most computer graphics (CG) papers, starting with [NSTN93], rely on a physical model of the medium comprising air molecules and aerosol particles, summarized in

Section 2.1. However, the classical rendering equation for participating media is rarely completely accounted for in atmospheric CG models, especially for interactive rendering. We restate the general model in Section 2.2 and we present its approximations in previous CG models in Section 2.3.

2.1. Physical model

The physical model commonly used in CG is a clear sky model based on two constituents, air molecules and aerosol particles, in a thin spherical layer of decreasing density between $R_g = 6360$ *km* and $R_t = 6420$ *km* (see Figure 1).

At each point, the proportion of light that is scattered θ degrees away from its incident direction is given by the product of a scattering coefficient β^s and of a phase function P. β^s depends on the particule density and P describes the angular dependency. For air molecules β^s and P are given by the Rayleigh theory [TS99]:

$$\beta^s_R(h,\lambda) = \frac{8\pi^3(n^2-1)^2}{3N\lambda^4} e^{-\frac{h}{H_R}} \quad (1)$$

$$P_R(\mu) = \frac{3}{16\pi}(1+\mu^2) \qquad \text{where } \mu = \cos\theta \quad (2)$$

where $h = r - R_g$ is the altitude, λ the wavelength, n the index of refraction of air, N the molecular density at sea level R_g, and $H_R = 8$ *km* is the thickness of the atmosphere if its density were uniform. As in [REK*04], we use $\beta^s_R = (5.8, 13.5, 33.1)10^{-6}\ m^{-1}$ for $\lambda = (680, 550, 440)$ *nm*. Aerosols also have an exponentially decreasing density, with a smaller height scale $H_M \simeq 1.2$ *km*. Their phase function is given by the Mie theory, approximated with the Cornette-Shanks phase function [TS99]:

$$\beta^s_M(h,\lambda) = \beta^s_M(0,\lambda) e^{-\frac{h}{H_M}} \quad (3)$$

$$P_M(\mu) = \frac{3}{8\pi}\frac{(1-g^2)(1+\mu^2)}{(2+g^2)(1+g^2-2g\mu)^{3/2}} \quad (4)$$

Unlike air molecules, aerosols absorb a fraction of the incident light. It is measured with an absorption coefficient β^a_M, which gives the extinction coefficient $\beta^e_M = \beta^s_M + \beta^a_M$ (see Figure 6 for typical values – $\beta^e_R = \beta^s_R$ for air molecules). Note that the variation of the index of refraction with altitude causes a small bending of rays (less than 2 degrees [HMS05]). We ignore it for simplicity.

2.2. Rendering equation

We recall here the rendering equation in a participating medium, applied to the atmosphere. We note $L(\mathbf{x},\mathbf{v},\mathbf{s})$ the radiance of light reaching $\mathbf{x}$ from direction $\mathbf{v}$ when the sun is in direction $\mathbf{s}$, and $\mathbf{x}_o(\mathbf{x},\mathbf{v})$ the extremity of the ray $\mathbf{x}+t\mathbf{v}$ (see Figure 1). Note that $\mathbf{x}_o$ is either on the ground or on the top atmosphere boundary $r = R_t$. The *transmittance* T between $\mathbf{x}_o$ and $\mathbf{x}$, the radiance $\mathcal{I}$ of light reflected at $\mathbf{x}_o$, and the radiance $\mathcal{J}$ of light scattered at $\mathbf{y}$ in direction $-\mathbf{v}$ are defined as

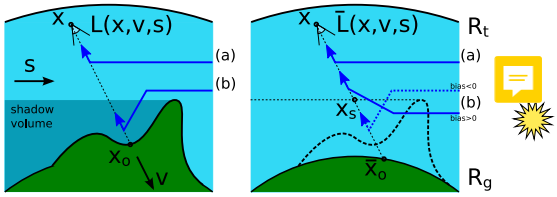


Figure 1: ***Our method***. Left: *the reference solution includes single-scattering (a) and multiple-scattering (b) integrated from* $\mathbf{x}$ *to* $\mathbf{x}_o$*, all accounting for occlusion.* Right: *our approximation. Integration is done from* $\mathbf{x}$ *to* $\mathbf{x}_s$*, ignoring occlusion (implicit via the use of* $\mathbf{x}_s$*). (a) is unchanged. (b) is affected by ignoring occlusion of secondary scatters (this yields both positive and negative bias, and effect is small anyway).*

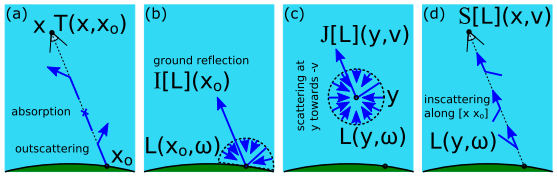


Figure 2: ***Definitions***. (a) *the atmospheric transparency* T *results from absorption and out scattered light.* (b) $\mathcal{I}[L]$ *is the light* L *reflected at* $\mathbf{x}_o$*. It is null on the top atmosphere boundary.* (c) $\mathcal{J}[L]$ *is the light* L *scattered at* $\mathbf{y}$ *in direction* $-\mathbf{v}$*.* (d) $\mathcal{S}[L]$ *is the light scattered towards* $\mathbf{x}$ *between* $\mathbf{x}_o$ *and* $\mathbf{x}$*, from any direction.*

follows (see Figure 2):

$$T(\mathbf{x},\mathbf{x}_o) = \exp\left(-\int_{\mathbf{x}}^{\mathbf{x}_o} \sum_{i\in\{R,M\}} \beta^e_i(\mathbf{y})dy\right) \quad (5)$$

$$\mathcal{I}[L](\mathbf{x}_o,\mathbf{s}) = \frac{\rho(\mathbf{x}_o)}{\pi}\int_{2\pi} L(\mathbf{x}_o,\boldsymbol{\omega},\mathbf{s})\boldsymbol{\omega}.\mathbf{n}(\mathbf{x}_o)d\omega, \text{ or } 0 \quad (6)$$

$$\mathcal{J}[L](\mathbf{y},\mathbf{v},\mathbf{s}) = \int_{4\pi}\sum_{i\in\{R,M\}} \beta^s_i(\mathbf{y})P_i(\mathbf{v},\boldsymbol{\omega})L(\mathbf{y},\boldsymbol{\omega},\mathbf{s})d\omega \quad (7)$$

Note that $\mathcal{I}$ is null on the top atmosphere boundary. With these notations the rendering equation is [TS99]:

$$L(\mathbf{x},\mathbf{v},\mathbf{s}) = (L_0 + \mathcal{R}[L] + \mathcal{S}[L])(\mathbf{x},\mathbf{v},\mathbf{s}) \quad (8)$$

$$L_0(\mathbf{x},\mathbf{v},\mathbf{s}) = T(\mathbf{x},\mathbf{x}_o)L_{sun}, \text{ or } 0 \quad (9)$$

$$\mathcal{R}[L](\mathbf{x},\mathbf{v},\mathbf{s}) = T(\mathbf{x},\mathbf{x}_o)\mathcal{I}[L](\mathbf{x}_o,\mathbf{s}) \quad (10)$$

$$\mathcal{S}[L](\mathbf{x},\mathbf{v},\mathbf{s}) = \int_{\mathbf{x}}^{\mathbf{x}_o} T(\mathbf{x},\mathbf{y})\mathcal{J}[L](\mathbf{y},\mathbf{v},\mathbf{s})dy \quad (11)$$

where L_0 is the direct sunlight L_{sun} attenuated before reaching $\mathbf{x}$ by $T(\mathbf{x},\mathbf{x}_o)$. L_0 is null if $\mathbf{v} \neq \mathbf{s}$, or if the sun is occluded by the terrain, *i.e.*, if $\mathbf{x}_o$ is on the ground. $\mathcal{R}[L]$ is the light reflected at $\mathbf{x}_o$ and also attenuated before reaching $\mathbf{x}$, and $\mathcal{S}[L]$ is the *inscattered* light, *i.e.*, the light scattered towards $\mathbf{x}$ between $\mathbf{x}$ and $\mathbf{x}_o$ (see Figure 2).

2.3. Previous rendering methods

Equation 8 is very complex to solve. Hence, many simplifying assumptions have been made in CG to find approxi-

mate solutions that are easier to compute (see [Slo02] for a survey). Most real-time methods ignore multiple scattering. In this case Equation 8 reduces to $L = L_0 + \mathcal{R}[L_0] + \mathcal{S}[L_0]$. However, even $\mathcal{S}[L_0]$ is quite complex to solve. Some authors propose analytical solutions at the price of idealisations: flat Earth with constant atmosphere density [HP02], or without Mie scattering [REK*04]. The flat Earth hypothesis limits them to observers on the ground. Otherwise, $\mathcal{S}[L_0]$ is generally computed by numerical integration [NSTN93], which can be done in real time using low sampling [O'N05]. A notable exception is [SFE07] who rely on precomputations of this integral. However, in order to reduce the number of parameters, they only take into account the view and sun zenith angles, and neglect the angle between the view and sun directions. Hence they cannot reproduce, *e.g.*, the Earth's shadow inside the atmosphere.

Ignoring multiple scattering as above is acceptable for daylight but not for twilight [HMS05]. This is because sunlight traverses much less atmosphere during the day than during sunset or sunrise. Hence, some authors propose methods to account for multiple scattering. [PSS99] fit the results of a double scattering Monte-Carlo simulation with an analytical model, but their model is only valid for an observer on the ground. [NDKY96] and [HMS05] use volume radiosity algorithms to compute multiple scattering, but their methods are far from real-time (minutes to hours per image).

In this paper we propose a new method to render the sky and the aerial perspective in real time, from *all viewpoints* from ground to space, while taking *multiple scattering* into account. It is inspired by [SFE07] and extends it with multiple scattering, with the previously ignored view-sun angle parameter, with a better parameterization for the precomputed tables, and with a new method for light shafts.

3. Our method

For efficiency and realism, our goal is to precompute L as much as possible, with only minimal approximations. Our solution is based on an exact computation for zero and single scattering, and uses an approximation of occlusion effects to compute multiple scattering. In fact we take the detailed ground shape into account for zero and single scattering, in order to get correct ground colors, shadows and light shafts. But we approximate it with a perfect sphere of constant reflectance to compute multiple scattering, to allow for precomputation.

Notations Before presenting our method we need some notations and auxiliary functions. We note $\bar{L} = \bar{L}_0 + (\bar{\mathcal{R}} + \bar{\mathcal{S}})[\bar{L}]$ the solution of Equation 8 for the case of a perfectly spherical ground of constant reflectance $\bar{\alpha}$. $\bar{L}_0$, $\bar{\mathcal{R}}$, $\bar{\mathcal{S}}$, $\bar{\mathbf{x}}_o$, $\bar{\mathcal{I}}$ and so on are defined as before, but for this spherical ground. Note that thanks to the ground's spherical symmetry $\mathbf{x}$ and $\mathbf{v}$ can be reduced to an altitude and a view zenith angle. Hence functions of $\mathbf{x}, \mathbf{v}, \mathbf{s}$ such as $\bar{L}$ or $\bar{\mathcal{S}}[\bar{L}]$ can be reduced to functions of 4 parameters (2 for $\mathbf{x}, \mathbf{v}$ and 2 for $\mathbf{s}$). Note also that L (resp. $\bar{L}$) can be expressed with a series in the linear operators $\mathcal{R}$ and $\mathcal{S}$ (resp. $\bar{\mathcal{R}}$ and $\bar{\mathcal{S}}$), where the i^{th} term corresponds to light reflected and/or scattered exactly i times:

$$\begin{aligned} L &= L_0 + (\mathcal{R}+\mathcal{S})[L_0] + (\mathcal{R}+\mathcal{S})\left[(\mathcal{R}+\mathcal{S})[L_0]\right] + \ldots \\ &= L_0 + L_1 + L_2 + \ldots = L_0 + L_* \end{aligned} \quad (12)$$

Zero and single scattering We compute L_0 and $\mathcal{R}[L_0]$ exactly, during rendering. For this we use a shadowing algorithm to compute the sun occlusion (see Equation 9), and a precomputed table for the transmittance T, which depends on only 2 parameters (see Section 4). $\mathcal{S}[L_0]$ is more complicated. It is an integral between $\mathbf{x}$ and $\mathbf{x}_o$ but, due to the occlusion term in L_0, the integrand is null at all points $\mathbf{y}$ that are in shadow (this is what gives light shafts). We suppose here that these points are between $\mathbf{x}_s$ and $\mathbf{x}_o$ (see Figure 1 – the general case is discussed in Section 5). Then the integral can be reduced to the lit segment $[\mathbf{x}, \mathbf{x}_s]$. Moreover, occlusion can be ignored since it is already accounted for via $\mathbf{x}_s$, *i.e.*, L_0 can be replaced with $\bar{L}_0$. This shows that $\mathcal{S}[L_0] = \int_{\mathbf{x}}^{\mathbf{x}_s} T\mathcal{J}[\bar{L}_0]$. By rewriting this as $\int_{\mathbf{x}}^{\bar{\mathbf{x}}_o} T\mathcal{J}[\bar{L}_0] - \int_{\mathbf{x}_s}^{\bar{\mathbf{x}}_o} T\mathcal{J}[\bar{L}_0]$, extending an idea introduced in [O'N05] and reused in [SFE07], we finally get a formulation using precomputable functions of 2 and 4 parameters, T and $\bar{\mathcal{S}}[\bar{L}_0]$:

$$\mathcal{S}[L_0](\mathbf{x}, \mathbf{v}, \mathbf{s}) = \bar{\mathcal{S}}[\bar{L}_0](\mathbf{x}, \mathbf{v}, \mathbf{s}) - T(\mathbf{x}, \mathbf{x}_s)\bar{\mathcal{S}}[\bar{L}_0](\mathbf{x}_s, \mathbf{v}, \mathbf{s}) \quad (13)$$

Multiple scattering As shown above L_0 and L_1 can be computed exactly despite the occlusion. Unfortunately accounting for occlusion in the other terms $L_2 + \ldots = \mathcal{R}[L_*] + \mathcal{S}[L_*]$ is much more difficult. Hopefully, in this case the occlusion can be approximated. Indeed, multiple scattering effects are small compared to single scattering during the day, while the ground contribution is small when it is not directly lit by the sun. So we approximate occlusion effects in $\mathcal{S}[L_*]$ by integrating the contribution of multiple scattering, computed without occlusion, between $\mathbf{x}$ and $\mathbf{x}_s$. This yields both positive and negative bias (see Figure 1). Mathematically, this approximation gives $\mathcal{S}[L_*] \simeq \int_{\mathbf{x}}^{\mathbf{x}_s} T\mathcal{J}[\bar{L}_*]$. We also approximate occlusion effects in $\mathcal{R}[L_*]$ with the ambient occlusion of an horizontal hemisphere due to the ground's tangent plane, $\frac{1+\mathbf{n}.\bar{\mathbf{n}}}{2}$. This gives $\mathcal{R}[L_*] \simeq \hat{\mathcal{R}}[\bar{L}_*]$ with:

$$\hat{\mathcal{R}}[\bar{L}_*] = T(\mathbf{x}, \mathbf{x}_o)\frac{\alpha(\mathbf{x}_o)}{\pi}\frac{1+\mathbf{n}(\mathbf{x}_o).\bar{\mathbf{n}}(\mathbf{x}_o)}{2}\bar{\mathcal{E}}[\bar{L}_*](\mathbf{x}_o, \mathbf{s}) \quad (14)$$

$$\bar{\mathcal{E}}[\bar{L}_*](\mathbf{x}_o, \mathbf{s}) = \int_{2\pi} \bar{L}_*(\mathbf{x}_o, \boldsymbol{\omega}, \mathbf{s})\boldsymbol{\omega}.\bar{\mathbf{n}}(\mathbf{x}_o)d\omega, \text{ or } 0 \quad (15)$$

By using the same rewriting rule as for Equation 13, and by noting $\bar{\mathcal{S}}[\bar{L}]_{|\mathbf{x}} = \bar{\mathcal{S}}[\bar{L}](\mathbf{x}, \mathbf{v}, \mathbf{s})$, we finally get:

$$L \simeq L_0 + \mathcal{R}[L_0] + \hat{\mathcal{R}}[\bar{L}_*] + \bar{\mathcal{S}}[\bar{L}]_{|\mathbf{x}} - T(\mathbf{x}, \mathbf{x}_s)\bar{\mathcal{S}}[\bar{L}]_{|\mathbf{x}_s} \quad (16)$$

where the first three terms can be quickly computed with the help of precomputed 2D tables for T and $\bar{\mathcal{E}}[\bar{L}_*]$, and where $\bar{\mathcal{S}}[\bar{L}]$ can be precomputed in a 4D table. We now show how to precompute them, in tables of a reasonnable size.

Algorithm 4.1: PRECOMPUTE(n_{orders})

$$
\begin{aligned}
&\mathbb{T}(\mathbf{x},\mathbf{v}) \leftarrow T(\mathbf{x},\bar{\mathbf{x}}_o(\mathbf{x},\mathbf{v}))\\
&\Delta\mathbb{E}(\mathbf{x},\mathbf{s}) \leftarrow \bar{\mathcal{E}}[\bar{L}_0](\mathbf{x},\mathbf{s})\\
&\Delta\mathbb{S}(\mathbf{x},\mathbf{v},\mathbf{s}) \leftarrow \bar{\mathcal{S}}[L_0](\mathbf{x},\mathbf{v},\mathbf{s})\\
&\mathbb{E}(\mathbf{x},\mathbf{s}) \leftarrow 0\\
&\mathbb{S}(\mathbf{x},\mathbf{v},\mathbf{s}) \leftarrow \Delta\mathbb{S}(\mathbf{x},\mathbf{v},\mathbf{s})\\
&\textbf{for } i \leftarrow 1 \textbf{ to } i < n_{orders}\\
&\quad \textbf{do} \begin{cases}
\Delta\mathbb{J}(\mathbf{x},\mathbf{v},\mathbf{s}) \leftarrow \mathcal{J}[T\frac{\alpha}{\pi}\Delta\mathbb{E}+\Delta\mathbb{S}](\mathbf{x},\mathbf{v},\mathbf{s})\\
\Delta\mathbb{E}(\mathbf{x},\mathbf{s}) \leftarrow \bar{\mathcal{E}}[T\frac{\alpha}{\pi}\Delta\mathbb{E}+\Delta\mathbb{S}](\mathbf{x},\mathbf{s}) = \bar{\mathcal{E}}[\Delta\mathbb{S}](\mathbf{x},\mathbf{s})\\
\Delta\mathbb{S}(\mathbf{x},\mathbf{v},\mathbf{s}) \leftarrow \int_{\mathbf{x}}^{\bar{\mathbf{x}}_o} T(\mathbf{x},\mathbf{y})\Delta\mathbb{J}(\mathbf{y},\mathbf{v},\mathbf{s})dy\\
\mathbb{E}(\mathbf{x},\mathbf{s}) \leftarrow \mathbb{E}(\mathbf{x},\mathbf{s})+\Delta\mathbb{E}(\mathbf{x},\mathbf{s})\\
\mathbb{S}(\mathbf{x},\mathbf{v},\mathbf{s}) \leftarrow \mathbb{S}(\mathbf{x},\mathbf{v},\mathbf{s})+\Delta\mathbb{S}(\mathbf{x},\mathbf{v},\mathbf{s})
\end{cases}
\end{aligned}
$$

4. Precomputations

We precompute $T(\mathbf{x},\bar{\mathbf{x}}_o(\mathbf{x},\mathbf{v}))$ for all $\mathbf{x},\mathbf{v}$ in a 2D table $\mathbb{T}(\mathbf{x},\mathbf{v})$. Due to spherical symmetry, $\mathbb{T}$ depends only on $r=\|\mathbf{x}\|$ and $\mu=\mathbf{v}.\mathbf{x}/r$ [O'N05]. As [O'N05], we then use the identity $T(\mathbf{x},\mathbf{y})=\mathbb{T}(\mathbf{x},\mathbf{v})/\mathbb{T}(\mathbf{y},\mathbf{v})$, with $\mathbf{v}=(\mathbf{y}-\mathbf{x})/\|\mathbf{y}-\mathbf{x}\|$.

We precompute $\bar{\mathcal{E}}[\bar{L}_*]$ and $\bar{\mathcal{S}}[\bar{L}]$ in two tables $\mathbb{E}$ and $\mathbb{S}$ with an algorithm that computes each scattering order $\bar{L}_i$ one after the other. This algorithm uses three intermediate tables $\Delta\mathbb{E}$, $\Delta\mathbb{S}$ and $\Delta\mathbb{J}$ containing after each iteration i $\bar{\mathcal{E}}[\bar{L}_i]$, $\bar{\mathcal{S}}[\bar{L}_i]$ and $\mathcal{J}[\bar{L}_i]$. $\Delta\mathbb{E}$ and $\Delta\mathbb{S}$ are added to the result tables $\mathbb{E}$ and $\mathbb{S}$ at the end of each iteration ($\bar{\mathcal{R}}$ is computed with the identity $\bar{\mathcal{R}}[L](\mathbf{x},\mathbf{v},\mathbf{s}) = T(\mathbf{x},\bar{\mathbf{x}}_o)\frac{\alpha}{\pi}\bar{\mathcal{E}}[L](\bar{\mathbf{x}}_o,\mathbf{s})$ – see Algorithm 4.1).

Angular precision Since $\mathbb{S}$ is a 4D table its size increases very quickly with resolution. So we can only use a limited angular resolution for $\mathbf{v}$. This poses a precision problem, which is however limited to the strong forward Mie scattering. In order to solve it we separate the single Mie scattering term from all the others in $\mathbb{S}$, so as to apply the phase function at runtime. For this we rewrite $\bar{\mathcal{S}}[\bar{L}]$ as $P_M\bar{\mathcal{S}}_M[\bar{L}_0]+P_R\bar{\mathcal{S}}_R[\bar{L}_0]+\bar{\mathcal{S}}[\bar{L}_*]$. We then store $C_M=\bar{\mathcal{S}}_M[\bar{L}_0]$ and $C_*=\bar{\mathcal{S}}_R[\bar{L}_0]+\bar{\mathcal{S}}[\bar{L}_*]/P_R$ separately, which requires 6 values per entry in $\mathbb{S}$. If necessary, for efficiency, this can be reduced to 4 values per entry by storing only the red component $C_{M,r}$ of C_M. In this case the other components can be approximated with a proportionality rule between $\bar{\mathcal{S}}_M[\bar{L}_0]$ and $\bar{\mathcal{S}}_R[\bar{L}_0]$, which gives $C_M \simeq C_* \frac{C_{M,r}}{C_{*,r}}\frac{\beta^s_{R,r}}{\beta^s_{M,r}}\frac{\beta^s_M}{\beta^s_R}$.

Parameterization In order to store $\bar{\mathcal{S}}[\bar{L}]$ into $\mathbb{S}$ we need a mapping from $(\mathbf{x},\mathbf{v},\mathbf{s})$ into table indices in $[0,1]^4$. A simple solution is to use $r=\|\mathbf{x}\|$ and the cosinus of the view zenith, sun zenith, and view sun angles, $\mu=\mathbf{v}.\mathbf{x}/r$, $\mu_s=\mathbf{s}.\mathbf{x}/r$ and $\nu=\mathbf{v}.\mathbf{s}$ (mapped linearly from $[R_g,R_t]\times[-1,1]^3$ to $[0,1]^4$).

The problem of this parameterization is that it requires a very high resolution in μ to get a good sampling for the aerial perspective. Consider for instance an observer near the ground looking horizontally, with a mountain at distance d (see Figure 3). The aerial perspective is given by Equation 16 as $\mathbb{S}(\mathbf{x},\mathbf{v},\mathbf{s}) - T(\mathbf{x},\mathbf{x}_s)\mathbb{S}(\mathbf{x}_s,\mathbf{v},\mathbf{s})$. Then $\mu=0$ for $\mathbf{x}$,

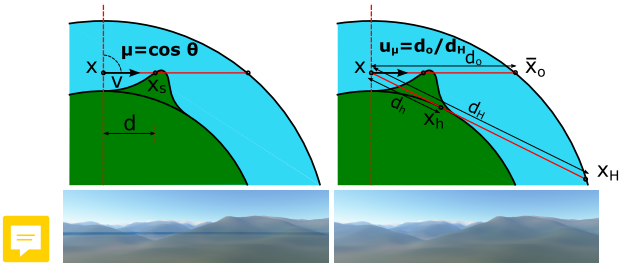


Figure 3: ***Viewing angle parameter***. Left: *using μ gives artifacts.* Right: *using $u_\mu = d_o/d_h$ or d_o/d_H solves the problem (using 128 values for μ or u_μ in the precomputed sky radiance table $\mathbb{S}$).*

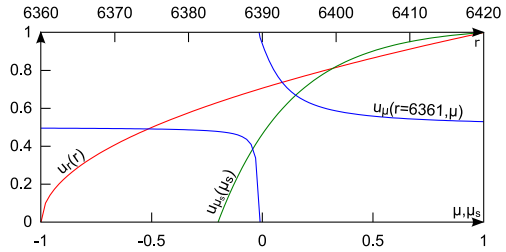


Figure 4: ***Parameterization***. *u_r, u_μ, u_{μ_s} as functions of r, μ, μ_s.*

and $d/\sqrt{r^2+d^2}$ for $\mathbf{x}_s$, which gives $\Delta\mu = 0.016 \ll 1$ for $d=100$ *km*. This too small value gives visible artifacts (see Figure 3). In order to solve this problem we rely on a better parameterization. We replace μ with u_μ, defined as the ratio between the distance $d_o=\|\bar{\mathbf{x}}_o-\mathbf{x}\|$ and the distance d_h (resp. d_H) from $\mathbf{x}$ to the horizon (resp. to the atmosphere boundary "behind" the horizon – see Figure 3). In the previous example $d_H \simeq (R_t^2-R_g^2)^{1/2}$ for $\mathbf{x}$ and $\mathbf{x}_s$, while $d_o \simeq d_H$ for $\mathbf{x}$ and d_H-d for $\mathbf{x}_s$, which gives $\Delta u_\mu = 0.11 \gg 0.016$ for $d=100$ *km*. With this mapping 128 samples for u_μ are sufficient to avoid the above artifacts.

Another problem is that $\mathbb{S}$ is discontinuous at the horizon, due to the discontinuity of the length of the viewing ray here. Hence a continuous mapping yields linear interpolations across this discontinuity, which causes artifacts. We solve this problem by ensuring that u_μ is itself discontinuous at the horizon (see Figure 4). Finally, we use an ad hoc non linear mapping for r and μ_s, chosen so as to get a better precision near the ground and for sun zenith angles near 90°. So our mapping from $(\mathbf{x},\mathbf{v},\mathbf{s})$ into $[0,1]^4$ is finally defined as follows:

$$
\begin{aligned}
u_r &= \rho/H\\
u_\mu &= 1/2+(r\mu+\sqrt{\Delta})/(2\rho) \qquad \text{if } r\mu<0 \text{ and } \Delta>0\\
&\quad\ 1/2-(r\mu-\sqrt{\Delta+H^2})/(2\rho+2H) \qquad \text{otherwise}\\
u_{\mu_s} &= (1-e^{-3\mu_s-0.6})/(1-e^{-3.6})\\
u_\nu &= (1+\nu)/2
\end{aligned}
$$

with $\rho=(r^2-R_g^2)^{1/2}$, $H=(R_t^2-R_g^2)^{1/2}$, and $\Delta=r^2\mu^2-\rho^2$.

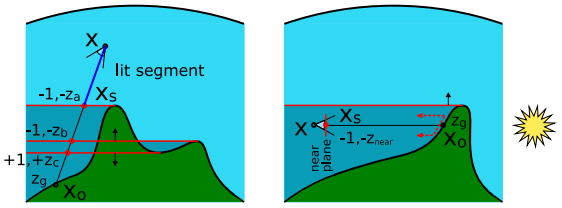


Figure 5: ***Evaluation of*** *l. Left: due to the false boundaries b and c, the computed length $\Delta z - \Delta n.z_g = z_g - z_a + z_c - z_b$ is larger than l. Clamping this value to $z_g - z_{min}$ fixes the problem. Right: viewpoint in shadow. Using only the extruded edges, $\mathbf{x}_o$ would be seen as lit and l would be equal to 0 instead of $z_g - z_{near}$. Projecting the back faces (dashed line) on the near plane solves the issue [HHLH05].*

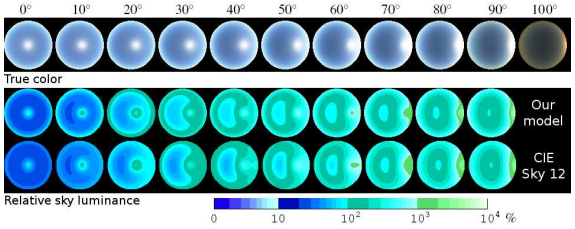


Figure 6: ***Validation.*** *The sky luminance $\mathbb{S}$ in fisheye view for several sun zenith angles, in color, and relatively to the zenith luminance. With $\bar{\alpha} = 0.1$, $\beta_M^s = 2\,10^{-5}\ m^{-1}$, $\beta_M^s/\beta_M^e = 0.9$, $g = 0.76$ and $H_M = 1.2$ km we get the CIE clear sky model, fitted from actual measurements (source [ZWP07]).*

5. Rendering

In order to render the sky and the aerial perspective we evaluate Equation 16 at each pixel. $\bar{L}_0$ can be efficiently computed using $\mathbb{T}$. Computing $\mathcal{R}[\bar{L}_0]$ involves $\mathbb{T}$, $\alpha(\mathbf{x}_o)$ and $\mathbf{n}(\mathbf{x}_o)$, and a shadow test to determine whether $\mathbf{x}_o$ is lit. Finally $\mathbb{E}$ and $\mathbb{S}$ are used to compute $\hat{\mathcal{R}}[\bar{L}_*]$ and $\bar{\mathcal{S}}[\bar{L}]$. As in [SFE07], $\mathbf{x}$ is the camera position or, if in space, the nearest intersection of the viewing ray with the atmosphere boundary. The only remaining non-trivial parameter is $\mathbf{x}_s$, which depends on the terrain shadows and gives light shafts.

Most light shaft algorithms use sampling or slicing to perform a numerical integration along the viewing ray, with a shadow map to find which samples are lit. Up to 100 samples per ray must be used to eliminate the artifacts due to the discrete sampling [IJTN07]. We propose here a new method inspired from shadow volumes [HHLH05]. It does not rely on numerical integration, and therefore does not suffer from these artifacts. We first show that an exact computation is possible but not adapted to the GPU. We then present an approximate solution better adapted to the GPU. Our idea is to use the precomputed integral $\mathbb{S}$ to compute the inscattered light due to each lit segment $[\mathbf{x}_i, \mathbf{x}_{i+1}]$ along the viewing ray, which is given by $T(\mathbf{x}, \mathbf{x}_i)\mathbb{S}_{|\mathbf{x}_i} - T(\mathbf{x}, \mathbf{x}_{i+1})\mathbb{S}_{|\mathbf{x}_{i+1}}$. By definition the points $\mathbf{x}_i$ are on the boundaries of the shadow volume of the terrain. Hence they can be found with a shadow volume algorithm such as [HHLH05]. This algorithm extrudes the silhouette edges of objects, as seen from the light; it also projects these objects on the near plane to get correct results despite clipping. However these algorithms also generate many surfaces that do not correspond to a boundary between light and shadow (see Figure 5). These false boundaries must be ignored when computing the inscattered light, otherwise a wrong result is obtained. Unfortunately, detecting them is a non local operation that is not adapted to GPU (requiring, *e.g.*, the use of multiple passes, or list structures).

Our solution is to use the shadow volume algorithm to compute the total length l of the shadowed segments, and to replace them with a single segment of this length at the "ground" end of the ray (see Figure 5). The false boundaries still cause problem, *i.e.*, an overestimation of l. Here however l can be clamped to the distance between the nearest and farthest faces of the shadow volume. This gives the correct result in most cases, and an approximate value in the others. Our detailed algorithm is the following. We associate with each pixel 4 values Δn, Δz, z_{min}, z_{max} initialized to 0, 0, ∞, 0. In a first step we decrement (resp. increment) Δn by 1 and Δz by the fragment depth z, and update z_{min} and z_{max} with z, for each front (resp. back) face of the shadow surface. In a second step we use (see Figure 5)

$$\tilde{l} = \text{clamp}(\Delta z - \Delta n.z_{ground}, 0, z_{ground} - z_{min})$$
$$L \simeq L_0 + \mathcal{R}[L_0] + \hat{\mathcal{R}}[\bar{L}_*] + \mathbb{S}_{|\mathbf{x}} - T(\mathbf{x}, \mathbf{x}_s)\mathbb{S}_{|\mathbf{x}_s = \mathbf{x}_o - \tilde{l}\mathbf{v}} \quad (17)$$

when looking at the ground or, when looking at the sky:

$$\tilde{l} = \text{clamp}(\Delta z, 0, z_{max})$$
$$L \simeq L_0 + \mathcal{R}[L_0] + \hat{\mathcal{R}}[\bar{L}_*] + T(\mathbf{x}, \mathbf{x}_s)\mathbb{S}_{|\mathbf{x}_s = \mathbf{x} + \tilde{l}\mathbf{v}} \quad (18)$$

6. Implementation, results and discussion

Precomputations We have implemented the precomputation algorithm on GPU, with fragment shaders processing the numerical integration. This is not mandatory but it allows us to quickly change atmospheric parameters, and it saves disk space (indeed 5 scattering orders are computed in 5 *seconds* on a NVidia 8800 GTS). We store $\mathbb{T}(r, \mu)$ and $\mathbb{E}(r, \mu)$ in 32×256 and 16×64 textures. We store $\mathbb{S}(u_r, u_\mu, u_{\mu_s}, u_\nu) = [\ldots_*, C_{M,r}]$ in a $32 \times 128 \times 32 \times 8$ table, seen as 8 3D tables packed in a single $32 \times 128 \times 256$ RGBA texture (using a manual linear interpolation for the 4^{th} coordinate). Thanks to our optimized parameterization our 4D table has a better precision and uses less space than the 3D table of [SFE07] (8 MB for $\mathbb{S}$ with 16 bits floats *vs* 12 MB for their 128^3 texture).

Rendering The rendering is done in four passes:

- we draw the terrain in the depth buffer only;
- we draw the shadow volume of the terrain into a Δn, Δz, z_{min}, z_{max} texture. For this we use the ADD and MAX blending functions, disable depth write, and use a geometry shader that extrudes the silhouette edges (as seen from the sun). This shader also projects on the near plane along $-\mathbf{s}$

the back faces (as seen from the sun) that are between this plane and the sun [HHLH05];
- we draw the terrain and the other objects with aerial perspective, as well as the sky, using Equations 17 and 18. If there are transparent objects such as clouds, aerial perspective must be computed for each object, before blending. We use Δn to compute occlusion in $\mathcal{R}[L_0]$, and $\tilde{l}$ computed as above to get $\mathbf{x}_s$ (see Section 5);
- we finally apply a global tone mapping function.

Results We run several tests with a height field and a reflectance texture from Nasa's Earth Observatory [SVS*05]. Results are depicted in Figures 8 and 9. As shown in Figures 6 and 7 our model can reproduce with good accuracy the CIE clear sky model, fitted from actual measurements at the ground level [DK02]. Since the sky color and aerial perspective are computed with a few texture fetches per pixel (< 10), our algorithm is quite fast. For instance, for the right view in Figure 8 in 1024×768, we get 125 fps without light shafts on a NVidia 8800 GTS. This includes 5 ms for the unshaded terrain, 0.4 ms for the first three terms in Equations 17 and 18 and 2.6 ms for the remaining terms (including 1 ms to evaluate the non linear parameterization). We get 25 fps with light shafts (*i.e.*, the first two rendering passes cost a lot, about 32 ms). By comparison, we get 50 fps with our reimplementation of [O'N05], using ten samples per ray (to get the same quality for single scattering, without shafts).

Limitations A limitation of our method is that the aerosol properties are assumed constant, depending only on altitude, whereas in fact they can greatly change depending on the atmospheric conditions [Slo02]. Since our precomputations are very fast we can change these properties quickly, but they remain uniform.

7. Conclusion

We have presented the first real-time method to render the sky and the aerial perspective from all viewpoints, with multiple scattering, terrain shadows and light shafts, and correct variation with all view and sun angles. This method is based on minimal simplifying assumptions that allow us to get an approximate solution of the rendering equation, in which most terms can be precomputed. This method can easily be extended to more complex physical models, with more constituents or more wavelengths.

As future work we would like to model the effect of clouds on the ground illuminance and on the aerial perspective, to remove the clear sky hypothesis. Indeed with many clouds the interreflections between the ground and the clouds should be taken into account [BNL06]. And their effect on aerial perspective should also be considered. To our knowledge, this has never been done.

The source code of our implementation is available at `http://evasion.inrialpes.fr/~Eric.Bruneton/`.

Acknowledgments This work was partially funded by the Natsim ANR ARA project. We would like to thank Antoine Bouthors and Cyril Soler for proofreading.

References

[BNL06] BOUTHORS A., NEYRET F., LEFEBVRE S.: Real-time realistic illumination and shading of stratiform clouds. In *Eurographics Workshop on Natural Phenomena* (sep 2006).

[DK02] DARULA S., KITTLER R.: CIE general sky standard defining luminance distributions. *eSim* (2002).

[HHLH05] HORNUS S., HOBEROCK J., LEFEBVRE S., HART J. C.: ZP+: correct Z-pass stencil shadows. In *ACM Symposium on Interactive 3D Graphics and Games (I3D)* (April 2005), ACM, ACM Press.

[HMS05] HABER J., MAGNOR M., SEIDEL H.-P.: Physically-based simulation of twilight phenomena. *ACM Trans. Graph. 24*, 4 (2005), 1353–1373.

[HP02] HOFFMAN N., PREETHAM A. J.: Rendering outdoor light scattering in real time. *Proceedings of Game Developer Conference* (2002).

[IJTN07] IMAGIRE T., JOHAN H., TAMURA N., NISHITA T.: Anti-aliased and real-time rendering of scenes with light scattering effects. *Vis. Comput. 23*, 9 (2007), 935–944.

[NDKY96] NISHITA T., DOBASHI Y., KANEDA K., YAMASHITA H.: Display method of the sky color taking into account multiple scattering. In *Proceedings of Pacific Graphics* (1996), pp. 117–132.

[NSTN93] NISHITA T., SIRAI T., TADAMURA K., NAKAMAE E.: Display of the Earth taking into account atmospheric scattering. In *SIGGRAPH 93* (1993), ACM, pp. 175–182.

[O'N05] O'NEIL S.: Accurate atmospheric scattering. In *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation* (2005), Addison-Wesley Professional.

[PSS99] PREETHAM A. J., SHIRLEY P., SMITS. B. E.: A practical analytic model for daylight. In *SIGGRAPH 99* (1999).

[REK*04] RILEY K., EBERT D. S., KRAUS M., TESSENDORF J., HANSEN C. D.: Efficient rendering of atmospheric phenomena. In *Rendering Techniques* (2004), pp. 374–386.

[SFE07] SCHAFHITZEL T., FALK M., ERTL T.: Real-time rendering of planets with atmospheres. In *WSCG International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision* (2007).

[Slo02] SLOUP J.: A survey of the modelling and rendering of the Earth's atmosphere. In *SCCG '02: Proceedings of the 18th spring conference on Computer graphics* (2002), ACM, pp. 141–150.

[SVS*05] STOCKLI R., VERMOTE E., SALEOUS N., SIMMON R., HERRING D.: The Blue Marble Next Generation – a true color Earth dataset including seasonal dynamics from MODIS. *NASA Earth Observatory* (2005).

[TS99] THOMAS G. E., STAMNES K.: *Radiative transfer in the atmosphere and ocean*. Cambridge Univ. Press, 1999.

[ZWP07] ZOTTI G., WILKIE A., PURGATHOFER W.: A critical review of the Preetham skylight model. In *WSCG 2007 Short Communications Proceedings I* (Jan. 2007), pp. 23–30.

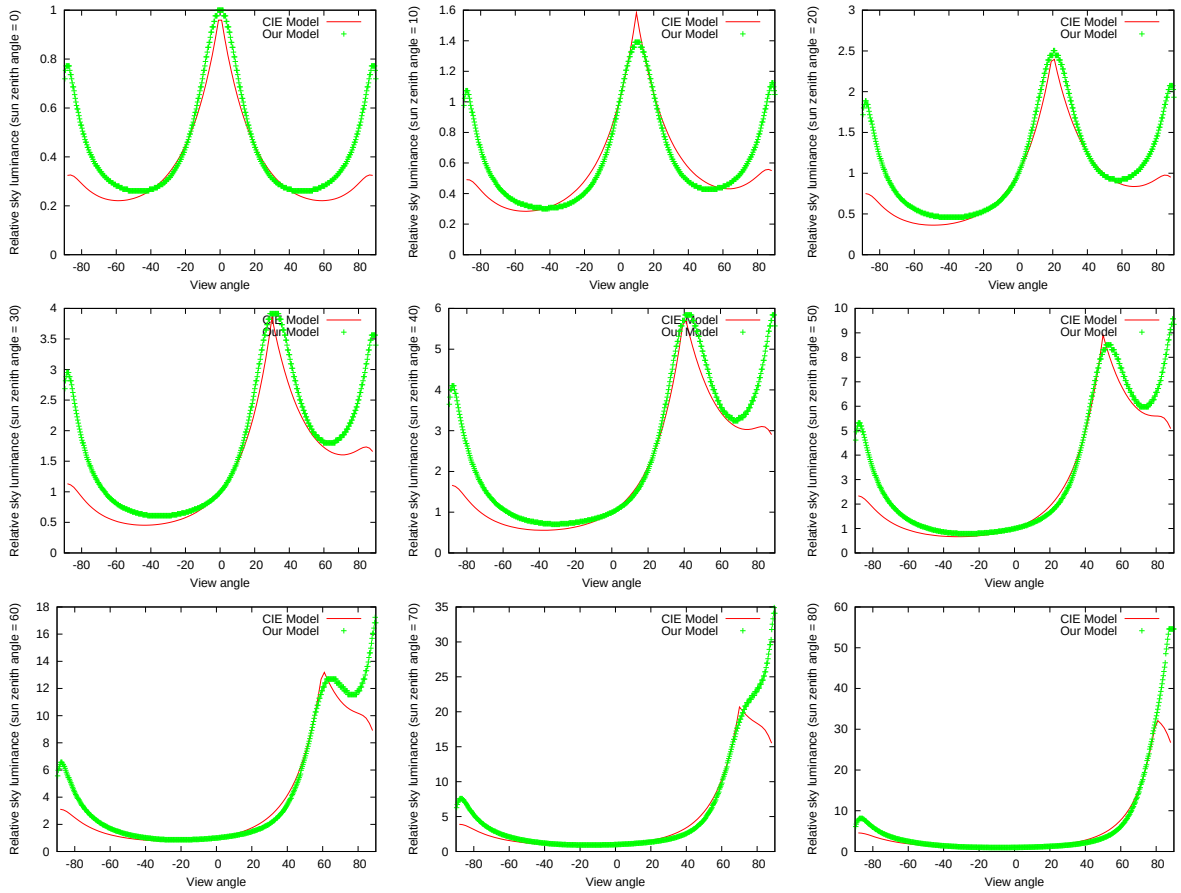


Figure 7: ***Validation.*** *The sky luminance relatively to the zenith luminance for several sun zenith and view zenith angles (and null azimuth between view and sun directions). Comparison between our model (with* $\bar{\alpha} = 0.1$, $\beta_M^s = 2.2\,10^{-5}\ m^{-1}$, $\beta_M^s/\beta_M^e = 0.9$, $g = 0.73$ *and* $H_M = 1.2$ *km) and the CIE sky model 12 (based on actual measurements). We note an overestimation near the horizon (view angles near 90 and -90), which is also visible in Figure 6. As shown in [ZWP07] the Preetham model [PSS99] also suffers from this problem, which probably comes from the physical models currently used in CG.*

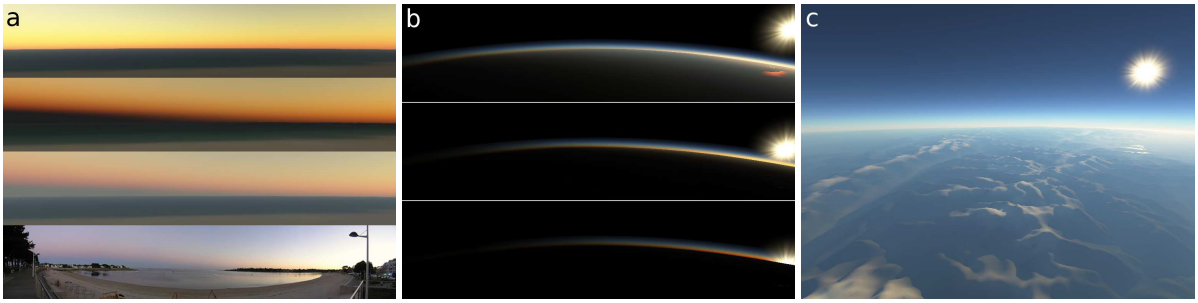
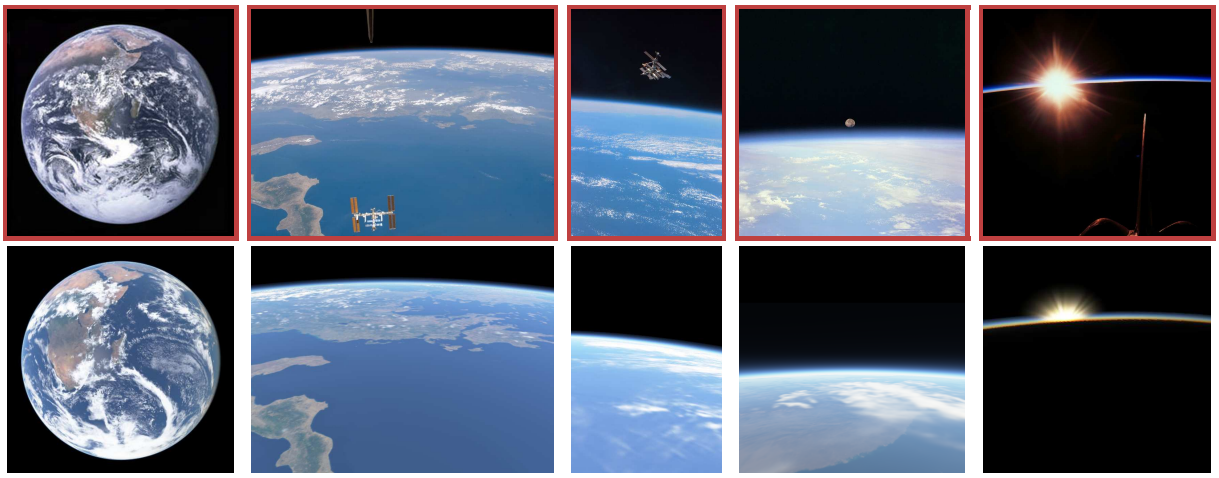
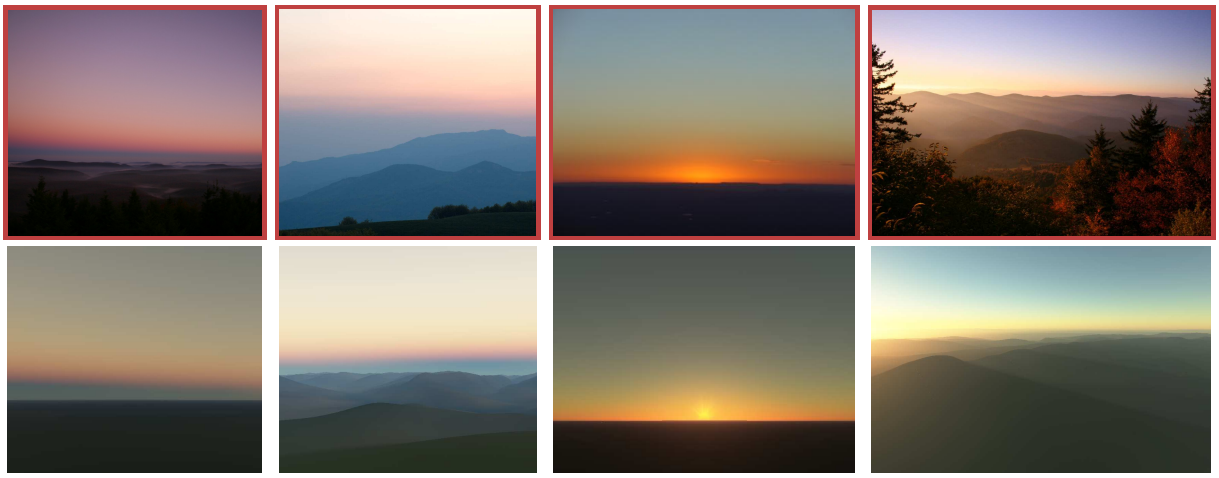


Figure 8: ***Results.*** (a), from top to bottom: *[SFE07], single scattering, multiple scattering and photo. With [SFE07] the shadow does not appear due to the missing* ν *parameter. It is too dark with single sattering only.* (b) *sunset viewed from space.* (c) *the view used for our performance measurements.*

Figure 9: ***Results.*** *Our results (*no frames*) compared with real photographs found on the Web (*red frames*). The tone mapping may explain the sky hue differences on some images compared with the uncalibrated photographs.*



Views from space for various altitudes and sun positions.



Views from the ground showing, from left to right, the Earth shadow, the aerial perspective after sunset, sunset, and light shafts at sunrise.

Aerial perspective during the day, and mountain shadows for various view and sun angles.