Real-time Realistic Ocean Lighting using Seamless Transitions from Geometry to BRDF

Eric Bruneton, Fabrice Neyret, Nicolas Holzschuch

EUROGRAPHICS 2010 / T. Akenine-Möller and M. Zwicker
(Guest Editors)

Volume 29 (2010), Number 2

Real-time Realistic Ocean Lighting using Seamless Transitions from Geometry to BRDF

Eric Bruneton[1,2] and Fabrice Neyret[1,3] and Nicolas Holzschuch[1,2]

[1] LJK – Grenoble Universités, [2] INRIA, [3] CNRS

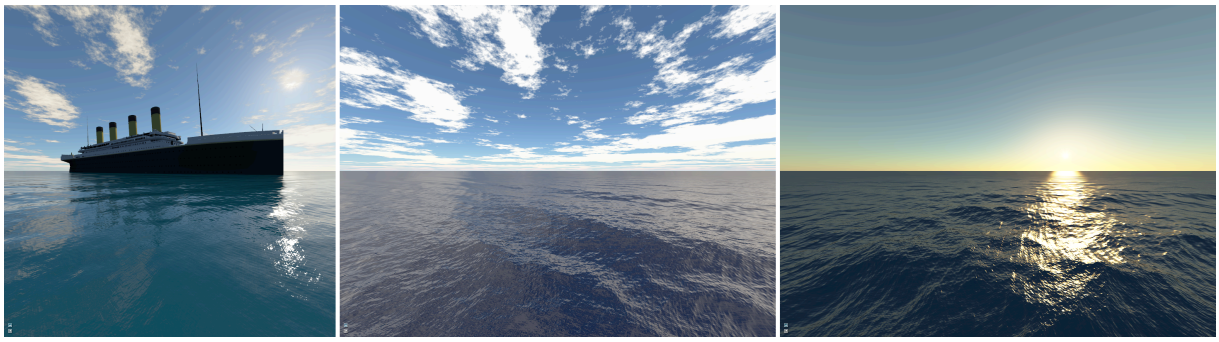

Figure 1: *Some real-time results obtained with our method, showing Sun reflections, sky reflections and local reflections from a boat. The lighting is correct at all distances thanks to accurate transitions from geometry to BRDF.*

Abstract

Realistic animation and rendering of the ocean is an important aspect for simulators, movies and video games. By nature, the ocean is a difficult problem for Computer Graphics: it is a dynamic system, it combines wave trains at all scales, ranging from kilometric to millimetric. Worse, the ocean is usually viewed at several distances, from very close to the viewpoint to the horizon, increasing the multi-scale issue, and resulting in aliasing problems. The illumination comes from natural light sources (the Sun and the sky dome), is also dynamic, and often underlines the aliasing issues. In this paper, we present a new algorithm for modelling, animation, illumination and rendering of the ocean, in real-time, at all scales and for all viewing distances. Our algorithm is based on a hierarchical representation, combining geometry, normals and BRDF. For each viewing distance, we compute a simplified version of the geometry, and encode the missing details into the normal and the BRDF, depending on the level of detail required. We then use this hierarchical representation for illumination and rendering. Our algorithm runs in real-time, and produces highly realistic pictures and animations.

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

1. Introduction

The surface of the sea, with its complex interplay between the waves and with the reflections of the Sun and the sky, plays an important role in our perception of the realism of ocean scenes. The ocean appears in several video games and movies, as well as simulators. There are many algorithms for modelling, animating and rendering the surface of the sea [FR86, PA00, HNC02, Tes01, YPZL05, HVT*06, CC06].

But the surface of the sea is, by its very nature, a highly complex problem for computer graphics: it is a dynamic system, which excludes precomputation. It combines together waves of different scales, ranging from the kilometric to the millimetric, making storage expensive. It is usually viewed at all distances at the same time, from the viewpoint to the horizon, resulting in aliasing issues. The illumination includes a quasi-point light source, the Sun, and a large non uniform area light source, the sky, as well as scatter-

ing effects under the surface. Illumination by a point light source further increases the multi-scale and aliasing issues. Although simulation and rendering of the ocean has been the subject of extensive research (see, *e.g.* [FR86,PA00,HNC02, Tes01,YPZL05,HVT*06,CC06]), there is not, currently, an algorithm that can address both storage, aliasing and illumination issues at the same time, for all scales. In this paper, we present an algorithm for real-time animation, illumination and rendering of the surface of the ocean. Our algorithm uses a combined representation of the surface, runs in real-time and produces highly realistic pictures, including complex lighting effects, without aliasing.

The core of our algorithm is a hierarchical representation of the ocean, combining geometry, normals and BRDF. At each viewing distance, we evaluate the required level of detail for the geometry representation, then encode the missing detail into the normal and the BRDF. The normal represents details that are too distant to make a visible contribution to the silhouette of the waves, but still close enough to make a contribution to their aspect. The BRDF encodes details that are so small (with respect to the viewing distance) that we can apply a micro-facet BRDF model. Our geometric model is a finite sum of wave trains of all wavelengths; the transition from geometry to normal to BRDF depends on the wavelength, for each wave train. Our algorithm is based on the deep water waves model of Pierson and Moskowitz [PM64], and shares the limitations of this model: our algorithm only works for deep water waves, and does not work for coasts and shores.

Our contributions are:

- a hierarchical representation of ocean waves, combining geometry, normals and BRDF, with smooth transitions,
- a fast approximate method to compute the illumination reflected by a glossy BRDF from a hemispherical environment map,
- a simple approximate formula for computing the Fresnel term for anisotropic rough surfaces.

Our paper is organized as follows. We review related work in the next section. We present our hierarchical representation of the ocean surface in Section 3. In Section 4 we present the ocean surface BRDF that we use in our illumination algorithm, presented in Section 5. We present several extensions to the main algorithm in Section 6, then show our results and validation experiments in Section 7. Finally, we conclude and explore avenues of future work in Section 8.

2. Previous Work

Physical ocean models. The ocean waves and the resulting surface statistics have been extensively studied by physicists [CM54,PM64,HDE80,RD07]. We summarize the most important results for our work in Section 3.1.

Computer graphics ocean models. Our work relies on existing methods to synthesize and represent the ocean shape:

- Synthesizing the surface has been done by summing wave trains [FR86,PA00,HNC02] or by using a FFT to convert a frequency spectrum to a surface [Tes01,CC06].
- Adaptive geometric resolution can be provided by a projected grid from screen [HNC02, CC06], by a dynamic quadtree [YPZL05], or by near and far patches [HVT*06].

In this paper, we have used the algorithm of Hinsinger *et al.* [HNC02], but we could have used another algorithm.

Several papers have also addressed the issue of illumination of the ocean: Hu *et al.* [HVT*06] simulated reflection and refraction in real-time using texture maps. Premoze and Ashikhmin [PA00] modelled the diffusion of light inside the water. However, to the best of our knowledge, nobody has addressed the issue of filtering both the ocean shape and lighting according to the viewing distance.

Reflectance models. Many BRDF models have been proposed for computer graphics. Cook and Torrance [CT81] and He *et al.* [HTSG91] proposed isotropic models. Ashikhmin [AS00] and Ward [War92] proposed models for anisotropic surfaces. In the physics literature Ross *et al.* [RDP05] proposed a physically-based anisotropic BRDF based on the surface slope variances. They derived it from a microfacets model taking masking and shadowing into account. In order to get accurate transitions from geometry to BRDF we need a physically-based anisotropic BRDF relying on physical surface parameters. Only the Ward and Ross models meet these requirements. We found that the Ross model was more accurate for the ocean. We therefore used this BRDF in our model. It is presented in section 4.

Multiresolution reflectance models. Transitioning from geometry to BRDF has not been investigated for the ocean case but has been studied in other contexts. The idea was first introduced by Kajiya [Kaj85] as a *hierarchy of scales*. [BM93] use transitions from geometry to bump mapping and then to BRDF. They introduce redistribution bump mapping to take apparent normal distributions (different from the real distribution due to masking effects) into account. [HSRG07] solve the problem in the context of normal maps. Their solution is based on a formulation of normal maps in terms of normal distribution functions, which can be mipmapped linearly. [TLQ*05] and [TLQ*08] compute reflectance mipmaps that can represent complex BRDFs with multiple lobes. All these methods assume a static surface and use precomputations. They are not applicable in our case since the surface is dynamic. Still, our work is inspired from [BM93]. A multiresolution reflectance model of sea surface in infrared was recently proposed by physicists [CFB*07]. Their model is too complex for real-time applications, but has been used to generate offline images.

3. Our ocean model

Multiresolution reflectance models are difficult to design in the general case. In our case the dynamic surface compli-

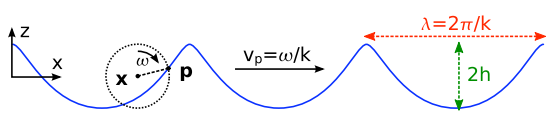

Figure 2: ***Trochoid waves****. A Gerstner wave is defined by* $\mathbf{p} = [x + h\sin(\omega t - kx),\ h\cos(\omega t - kx)]^T$*, where* $\omega = \sqrt{gk}$*.*

cates the problem because it forbids precomputations. On the other hand we have access to both the geometry and the spectrum of the surface [PM64], that is easy to filter, by removing frequencies above the Nyquist limit. In addition, the surface of the ocean has Gaussian statistic properties, at almost all scales. This is the starting hypothesis of many BRDF models [CT81]. This section presents our method to transition from geometry to normals and then to shading based on statistical surface properties. We first recall some physical facts about deep water waves.

3.1. Deep ocean waves

The ocean wave wavelengths vary from a few millimeters (capillary waves) to several hundred meters (gravity waves). The Pierson-Moskowitz spectrum [PM64] gives the energy distribution of gravity waves as a function of their frequency:

$$h(\omega) \propto \sqrt{S(\omega)}, \quad S(\omega) = \frac{\alpha g^2}{\omega^5} \exp\left[-\beta\left(\frac{\omega_0}{\omega}\right)^4\right] \qquad (1)$$

where h is the amplitude, $g = 9.81\ m.s^{-2}$, $\alpha = 8.1 \times 10^{-3}$, $\beta = 0.74$, and where $\omega_0 = g/V_{20}$ depends on the wind velocity V_{20} at 20 m above the surface. Hasselmann [HDE80] extended this model with a wave direction parameter.

The wave directions are anisotropic, which gives an *anisotropic surface*. Cox and Munk [CM54] found that the surface slopes have an elliptical Gaussian distribution whose major axis is aligned with the wind direction. The slope variance can be two times larger in this direction than in the crosswind direction [RD07]. Finally, the shape of waves can be modeled with trochoids (see Figure 2), which are exact solutions to the Euler fluid equations for gravity waves in deep water (found by Gerstner in 1809).

3.2. Model hierarchy

We model the ocean surface with a sum of n trochoid wave trains of amplitude h_i, wavenumber $\mathbf{k}_i$ and angular frequency ω_i sampled from the Pierson-Moskowitz and Hasselmann spectrums ($n = 60$ in our examples). We render the ocean with a regular grid in screen space, projected on the horizontal plane, displaced by waves and projected back to screen, as in [HNC02] (see Figure 3).

In order to transition from geometry to BRDF we represent the ocean surface inside a screen area with three models: an average position $\mathbf{p}$, an average normal $\mathbf{n}$ and a BRDF. As the view distance increases, details filtered out from one model are reintroduced in the next one (see Figure 3).

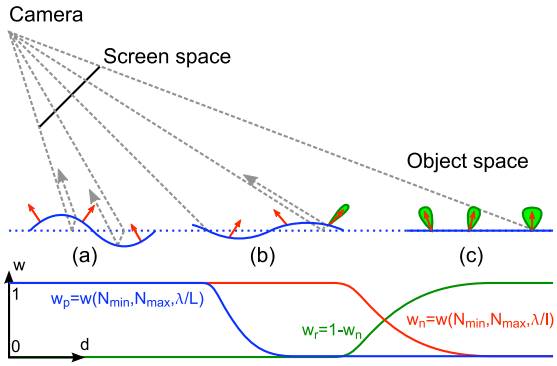

Figure 3: ***Ocean model****. A regular grid in screen space is projected on the horizontal plane, displaced by waves and projected back to screen (gray arrows). Each wave (in blue) is attenuated by a weight* w_p *(bottom) to avoid aliasing and popping. Per pixel normals (in red) are computed and attenuated independently by* w_n*. They are eventually replaced with a distribution of normals,* i.e.*, a BRDF (in green).*

Average positions. We compute the average position inside a grid cell by filtering the trochoids whose wavelength λ is less than N_{min} times the projected grid cell size L in object space ($N_{min} = 2$ according to Nyquist – see Figure 3). For that we scale each trochoid by $w_p = w(N_{min}, N_{max}, \lambda/L)$, with $w(a,b,x) = 3\bar{x}^2 - 2\bar{x}^3, \bar{x} = \text{clamp}(\frac{x-a}{b-a}, 0, 1)$:

$$\mathbf{p} = \begin{bmatrix}\mathbf{x}\\ 0\end{bmatrix} + \sum_1^n w_{p,i}\,\mathbf{t}_i, \quad \mathbf{t}_i = \begin{bmatrix}\frac{\mathbf{k}_i}{\|\mathbf{k}_i\|} h_i \sin(\omega_i t - \mathbf{k}_i \cdot \mathbf{x})\\ h_i \cos(\omega_i t - \mathbf{k}_i \cdot \mathbf{x})\end{bmatrix} \qquad (2)$$

where $\mathbf{x} = [x\ y]^T$ is the ocean surface at rest. This eliminates geometric aliasing as well as popping.

Average normals. We compute the average normal inside a pixel in a similar way:

$$\mathbf{n} = \left(\begin{bmatrix}\frac{\partial \mathbf{x}}{\partial x}\\ 0\end{bmatrix} + \sum_1^n w_{n,i}\frac{\partial \mathbf{t}_i}{\partial x}\right) \wedge \left(\begin{bmatrix}\frac{\partial \mathbf{x}}{\partial y}\\ 0\end{bmatrix} + \sum_1^n w_{n,i}\frac{\partial \mathbf{t}_i}{\partial y}\right) \qquad (3)$$

where the filter weight $w_n = w(N_{min}, N_{max}, \lambda/l)$ is such that wavelengths less than N_{min} times the projected size of a pixel l in object space are canceled (see Figure 3). Note that normals are *not* computed from the average positions. Hence, normals remain exact at longer distances than the geometry itself (see Figure 3, b). However, the *apparent* normals become wrong because masking effects change when the geometry is filtered [BM93]. In particular, we can get backfacing normals ($\mathbf{n}.\mathbf{v} < 0$). We avoid this problem with a simple trick: we reflect backfacing normals with $\mathbf{n} \leftarrow \mathbf{n} - 2(\mathbf{n}.\mathbf{v})\mathbf{v}$.

BRDF. We represent the subpixel surface details with their statistical properties, from which we compute a BRDF. Trochoids with different wavelengths can be viewed as independent random variables. According to the central limit theorem the sum of many of these trochoids gives a surface

whose slopes have a Gaussian distribution whose variance is the sum of the variance of each trochoid slope distribution:

$$\begin{bmatrix}\sigma_x^2\\ \sigma_y^2\end{bmatrix} = \sum_1^n \frac{[k_{i,x}^2 \;\; k_{i,y}^2]^T}{\|\mathbf{k}_i\|^2}\left(1-\sqrt{1-\|\mathbf{k}_i\|^2 w_r^2 h_i^2}\right) \quad (4)$$

where σ_x^2 and σ_y^2 are the slope variances along the x and y axis (see Appendix A), and $w_r = 1 - w_n$ (see Figure 3). In practice we compute these variances along the average wave direction, called the *wind direction*, and in the perpendicular direction. These directions correspond to the axes of the elliptical Gaussian slope distribution [CM54].

According to our experiments Eq. 4 holds when summing at least 10 trochoids. Hence, in theory we should not zoom in too much so that at least 10 trochoids have a wavelength smaller than a pixel. In practice we get good results even with less than 10 trochoids.

4. Ocean BRDF

In the context of ocean optics, Ross *et al*. [RDP05] have recently found a very accurate BRDF model for anisotropic rough surfaces whose slopes and heights follow Gaussian distributions, with uncorrelated heights and slopes (which is the case when summing enough trochoids). They derived their BRDF by computing the probability to see a microfacet of slopes $\boldsymbol{\zeta}$, which is visible from both the viewer $\mathbf{v}$ and the source $\mathbf{l}$ (see Figure 5), using Smith [Smi67] shadowing factors. Their major contribution was to analytically integrate the resulting expression to get a *normalized* visibility probability distribution q_{vn} [RDP05]:

$$q_{vn}(\boldsymbol{\zeta},\mathbf{v},\mathbf{l}) = \frac{p(\boldsymbol{\zeta})\max(\mathbf{v}\cdot\mathbf{f},0)H(\mathbf{l}\cdot\mathbf{f})}{(1+\Lambda(a_v)+\Lambda(a_l))f_z\cos\theta_v}\mathrm{d}^2\boldsymbol{\zeta} \quad (5)$$

$$\mathbf{f}(\boldsymbol{\zeta}) = \begin{bmatrix}f_x\\ f_y\\ f_z\end{bmatrix} = \frac{1}{\sqrt{1+\zeta_x^2+\zeta_y^2}}\begin{bmatrix}-\zeta_x\\ -\zeta_y\\ 1\end{bmatrix} \quad (6)$$

$$p(\boldsymbol{\zeta}) = \frac{1}{2\pi\sigma_x\sigma_y}\exp\left(-\frac{1}{2}\left(\frac{\zeta_x^2}{\sigma_x^2}+\frac{\zeta_y^2}{\sigma_y^2}\right)\right) \quad (7)$$

$$\Lambda(a_i) = \frac{\exp(-a_i^2) - a_i\sqrt{\pi}\,\mathrm{erfc}(a_i)}{2a_i\sqrt{\pi}}, i\in\{v,l\} \quad (8)$$

$$a_i = \left(2\left(\sigma_x^2\cos^2\phi_i+\sigma_y^2\sin^2\phi_i\right)\tan\theta_i\right)^{-1/2} \quad (9)$$

where $\mathbf{f}$ is the normal of the microfacet of slopes $\boldsymbol{\zeta}$, p is the Gaussian distribution of these slopes, and Λ comes from Smith shadowing factors. σ_x^2 and σ_y^2 are the slope variances along x and y, and H is the Heaviside function. In the absence of light source the visible interaction probability becomes:

$$q_{vn}^e(\boldsymbol{\zeta},\mathbf{v}) = \frac{p(\boldsymbol{\zeta})\max(\mathbf{v}\cdot\mathbf{f},0)}{(1+\Lambda(a_v))f_z\cos\theta_v}\mathrm{d}^2\boldsymbol{\zeta} \quad (10)$$

Thanks to the normalization factor $1+\Lambda(a_v)$ Ross *et al*. get:

$$\iint_{-\infty}^{\infty} q_{vn}^e(\boldsymbol{\zeta},\mathbf{v})\mathrm{d}^2\boldsymbol{\zeta} = 1 \quad (11)$$

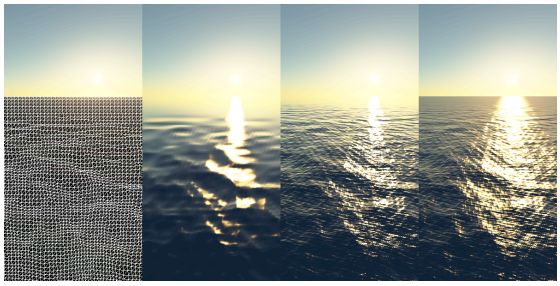

Figure 4: ***Transitions from geometry to BRDF***. From left to right*: screen space grid (typically 8×8 pixels cells), geometry only, geometry with per pixel normals, and geometry with normals and BRDF. The BRDF represents subpixel surface details and ensures a correct shading without aliasing*.

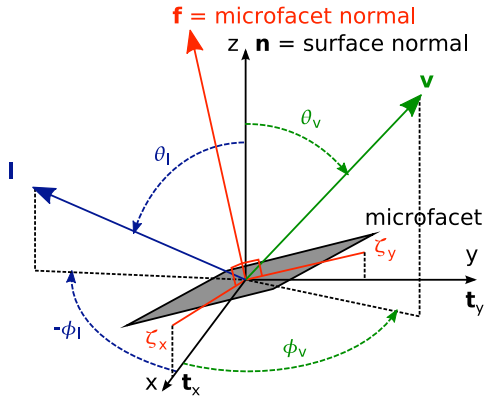

Figure 5: ***BRDF model coordinates*** *(from [RDP05])*. $\mathbf{v}$ *and* $\mathbf{l}$ *are unit vectors towards the viewer and the light*. $\mathbf{f}$ *is the normal of a microfacet whose x and y slopes are* ζ_x *and* ζ_y.

meaning that the probability to see at least one facet is 1, as expected ($\int q_{vn} < 1$ because some facets are shadowed).

If we neglect multiple reflections and assume that each microfacet is a perfect mirror, the BRDF is the probability to see a microfacet of slopes $\boldsymbol{\zeta}_h$ corresponding to the half vector $\mathbf{h}$ between $\mathbf{v}$ and $\mathbf{l}$, times the Fresnel factor F. Using the change of variables [RDP05]

$$\mathrm{d}^2\boldsymbol{\zeta} = \frac{\sin\theta_l\mathrm{d}\theta_l\mathrm{d}\phi_l}{4h_z^3\mathbf{v}\cdot\mathbf{h}} = \frac{\mathrm{d}^2\boldsymbol{\omega}_l}{4h_z^3\mathbf{v}\cdot\mathbf{h}} \quad (12)$$

Ross *et al*. get:

$$\mathrm{brdf}(\mathbf{v},\mathbf{l}) = \frac{q_{vn}(\boldsymbol{\zeta}_h,\mathbf{v},\mathbf{l})F(\mathbf{v}\cdot\mathbf{h})}{4h_z^3\cos\theta_l\mathbf{v}\cdot\mathbf{h}} \quad (13)$$

5. Ocean lighting

This section presents our method to compute the reflected light from the Sun and from the sky dome, and the refracted light from the water (see Figure 6), using the Ross BRDF

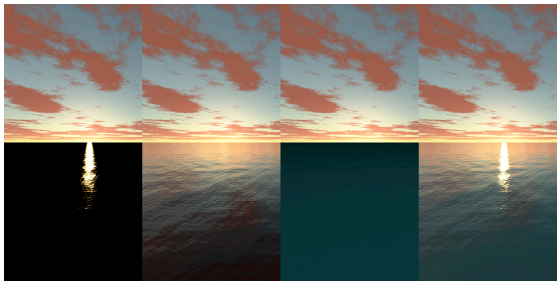

Figure 6: ***Ocean lighting****. From left to right: we add the reflected Sun light, the reflected sky light and the light refracted from the water to get the final result.*

and the slope variances of Eq. 4. We consider distant lighting only, and we ignore multiple reflections (local lighting and multiple reflections are discussed in Section 7.1).

5.1. Sun light

We compute the light reflected from the Sun at $\mathbf{p}$ (see Eq. 2) by applying the BRDF of Eq. 13 in the tangent frame aligned with the average normal $\mathbf{n}$ of Eq. 3 and the wind direction, and with the slope variances of Eq. 4. By doing this we approximate the slope distribution in the tangent space with a Gaussian (this is only true in world space), of the same variances as in world space. This is acceptable if $\mathbf{n}$ is not too far from the vertical (according to Ross *et al.*, the corresponding error is very small [private communication]).

As [RDP05] we consider the BRDF as constant over the Sun solid angle Ω_{sun}. We also use Schlick's model [Sch94] for the Fresnel factor F:

$$F(\mathbf{v}\cdot\mathbf{h}) \approx R + (1-R)(1-\mathbf{v}\cdot\mathbf{h})^5 \tag{14}$$

The reflected Sun radiance $\iint \text{brdf}(\mathbf{v},\mathbf{l}) L_{sun} \cos\theta_l \mathrm{d}^2\boldsymbol{\omega}_l$ is then, using Eq. 13:

$$I_{sun} \approx L_{sun}\Omega_{sun} p(\boldsymbol{\zeta}_h) \frac{R + (1-R)(1-\mathbf{v}\cdot\mathbf{h})^5}{4h_z^4 \cos\theta_v (1+\Lambda(a_v)+\Lambda(a_l))} \tag{15}$$

where L_{sun} is the Sun radiance. When the surface becomes flat the BRDF becomes a Dirac. This would give a punctual Sun specular highlight, instead of a finite disc. To avoid this we simulate the integral of the Dirac BRDF over the solar disc by clamping the slope variances σ_x^2 and σ_y^2 to a minimum value in Eq. 15.

Self-shadowing can be provided with a shadow map for close views. For distant views its effects are already taken into account in the Ross BRDF.

5.2. Sky light

Computing the light reflected from the sky dome is difficult because it requires to integrate the BRDF with the sky radiance $L_{sky}(\mathbf{l})$ over a hemisphere Ω. In our case the BRDF varies from purely specular to directional diffuse. This precludes the use of spherical harmonics. The BRDF is also anisotropic, which excludes spherical radial basis functions. It also has two directional parameters, which is a problem for prefiltered environment maps [KVHS00], as it leads to high dimensional textures. We propose here a fast approximate method for specular to directional diffuse BRDFs assuming an isotropic or anisotropic Gaussian slope distribution. Our method does not require precomputations and uses hardware texture filtering to approximate the lighting integral.

Approximate environment lighting. Microfacet BRDF models [CT81, War92, RDP05] share a similar expression, which denotes the fact that the BRDF is proportional to the fraction of microfacets whose normal is equal to the half-vector (if multiple reflections are neglected). By noting ρ the proportionality coefficient, we get:

$$\text{brdf}(\mathbf{v},\mathbf{l}) = p(\boldsymbol{\zeta}_h)\rho(\mathbf{v},\mathbf{l}) \tag{16}$$

In the surface's tangent space the lighting integral is then:

$$I_{sky} = \iint_\Omega p(\boldsymbol{\zeta}_h)\rho(\mathbf{v},\mathbf{l}) L_{sky}(\mathbf{l}) \cos\theta_l \mathrm{d}^2\boldsymbol{\omega}_l \tag{17}$$

Using the change of variables of Eq. 12, and by posing:

$$\mathbf{r}(\mathbf{v},\boldsymbol{\zeta}) = 2(\mathbf{v}\cdot\mathbf{f}(\boldsymbol{\zeta}))\mathbf{f}(\boldsymbol{\zeta}) - \mathbf{v} = [r_x\ r_y\ r_z]^T \tag{18}$$

$$\rho'(\mathbf{v},\boldsymbol{\zeta}) = 4h_z^3 \mathbf{v}\cdot\mathbf{f}(\boldsymbol{\zeta})\, r_z\, \rho(\mathbf{v},\mathbf{r}(\mathbf{v},\boldsymbol{\zeta})) \tag{19}$$

we get:

$$I_{sky} = \iint_{-\infty}^{\infty} p(\boldsymbol{\zeta})\rho'(\mathbf{v},\boldsymbol{\zeta}) L_{sky}(\mathbf{r}) H(r_z) \mathrm{d}^2\boldsymbol{\zeta} \tag{20}$$

where we replaced $\mathbf{l}$ with $\mathbf{r}$, the view vector reflected by the microfacet of normal $\mathbf{f}(\boldsymbol{\zeta})$, and where $H(r_z)$ restricts the integral to Ω. This integral can be seen as the average of the product of two terms, weighted by p. If we assume that p has a small support, *i.e.*, if the BRDF lobe is narrow, we can approximate it with a product of two averages (*i.e.*, $\int p\rho' L \approx \int p\rho' \int pL$):

$$I_{sky} \approx \bar{F}\bar{L}, \quad \bar{F}(\mathbf{v}) = \iint_{-\infty}^{\infty} p(\boldsymbol{\zeta})\rho'(\mathbf{v},\boldsymbol{\zeta}) H(r_z) \mathrm{d}^2\boldsymbol{\zeta} \tag{21}$$

$$\bar{L}(\mathbf{v}) = \iint_{-\infty}^{\infty} p(\boldsymbol{\zeta}) L_{sky}(\mathbf{r}) H(r_z) \mathrm{d}^2\boldsymbol{\zeta} \tag{22}$$

This approximation is exact when the BRDF is purely specular (p is a Dirac). It becomes less accurate when the BRDF becomes diffuse (see Section 7.2). We now explain how we compute the two averages $\bar{F}$ and $\bar{L}$.

Average Fresnel reflectance. In the case of the Ross BRDF (see Eq. 13), Eq. 21 gives:

$$\bar{F}(\mathbf{v}) = \iint_{-\infty}^{\infty} q_{vn}(\boldsymbol{\zeta},\mathbf{v},\mathbf{r}) F(\mathbf{v}\cdot\mathbf{h}) H(r_z) \mathrm{d}^2\boldsymbol{\zeta} \tag{23}$$

which can be seen as an average or *effective* Fresnel reflectance. We found experimentally that we could approximate $q_{vn}H$ with q_{vn}^e in this integral (see Eq. 10). Then, using

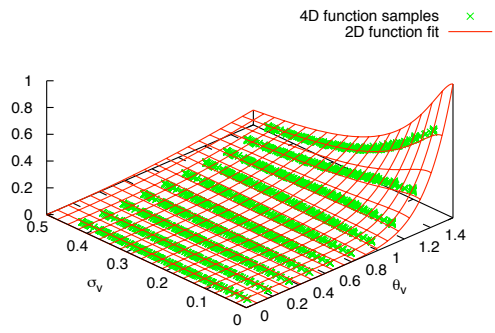

Figure 7: ***Effective Fresnel reflectance***. *Plot of the reflectance of anisotropic rough surfaces* $\bar{F}(\theta_v, \phi_v, \sigma_x, \sigma_y)$ *(in green – Eq. 23), and of our fitting function (in red), as functions of* θ_v *and* σ_v^2*, the slope variance in the view direction.*

Schlick's approximation and Eq. 11, we get:

$$\bar{F}(\mathbf{v}) \approx R + (1-R) \iint_{-\infty}^{\infty} q^e_{vn}(\boldsymbol{\zeta}, \mathbf{v})(1-\mathbf{v}\cdot\mathbf{h})^5 \mathrm{d}^2\boldsymbol{\zeta} \quad (24)$$

The remaining integral depends on σ_x, σ_y and $\mathbf{v}$. However, we found experimentally that for $\sigma_x, \sigma_y < 0.5$ it mainly depends on θ_v and on the slope variance in the view direction

$$\sigma_v^2 = \sigma_x^2 \cos^2\phi_v + \sigma_y^2 \sin^2\phi_v \quad (25)$$

Also when $\sigma_v \to 0$, q^e_{vn} becomes a Dirac and we get back Schlick's formula. We therefore looked for a generalization of this formula that could fit $\bar{F}$. We found the following fitting function (see Figure 7):

$$\bar{F}(\mathbf{v}) \approx R + (1-R)\frac{(1-\cos\theta_v)^{5\exp(-2.69\sigma_v)}}{1+22.7\sigma_v^{1.5}} \quad (26)$$

Average sky radiance. In order to compute the average sky radiance $\bar{L}$ we drop the Heaviside term in Eq. 22. This approximation allows us to compute $\bar{L}$ as a filtering of L_{sky} with the filter kernel p, which can be approximated with an anisotropic texture fetch (as shown below). On the other hand this approximation can give unwanted extra light for grazing view angles (for other angles p already restricts the integral to a domain inside the hemisphere).

Lets assume that L_{sky} is stored in a single texture $\mathbb{L}$ (our method is also valid with multiple textures). We note $\mathbf{u}(\mathbf{v}, \boldsymbol{\zeta}) = \mathcal{U}(\mathcal{R}(\mathbf{r}(\mathbf{v}, \boldsymbol{\zeta})))$ the function mapping microfacets slopes $\boldsymbol{\zeta}$ to texture coordinates $\mathbf{u}$, via reflected view vectors $\mathbf{r}$ in tangent space, transformed to world space with the rotation $\mathcal{R}$ from tangent space to world space. If p is the Gaussian of Eq. 7, its support in slope space is the ellipse of axes $2\sigma_x$ and $2\sigma_y$ centered at 0 (see Figure 8). If the function $\mathbf{u}$ is carefully chosen so as to minimize the distortions from slope space to texture space for any $\mathbf{v}$, we can then approximate p's support in texture space with the ellipse centered at

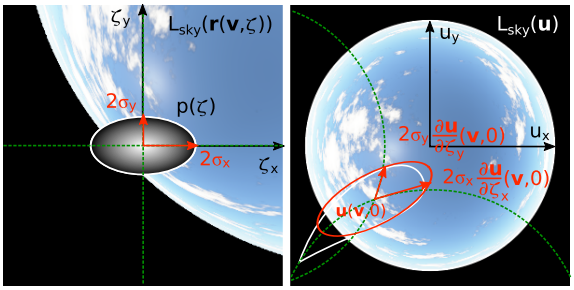

Figure 8: ***Environment map filtering***. Left*: the reflected light* I_{sky} *is an elliptical Gaussian filtering (in white) of the sky light* $L_{sky}(\mathbf{r}(\mathbf{v}, \boldsymbol{\zeta}))$ *in slope space. But it cannot be evaluated with an anisotropic texture fetch because* L_{sky} *cannot be stored in a* $\mathbb{L}(\boldsymbol{\zeta})$ *texture – it also depends on* $\mathbf{v}$. Right*: with a carefully chosen environment map parameterization* $\mathcal{U}(\mathbf{r})$*, the transformed filter (in white) stays close to an ellipse (in red) in the environment map texture space.* I_{sky} *can then be approximated with an anisotropic texture fetch.*

$\mathbf{u}(\mathbf{v}, 0)$ and of axes $2\sigma_x \frac{\partial \mathbf{u}}{\partial \zeta_x}(\mathbf{v}, 0)$ and $2\sigma_y \frac{\partial \mathbf{u}}{\partial \zeta_y}(\mathbf{v}, 0)$ (see Figure 8). This gives:

$$\bar{L} \approx \mathrm{tex2D}(\mathbb{L}, \mathbf{u}(\mathbf{v}, 0), 2\sigma_x \frac{\partial \mathbf{u}}{\partial \zeta_x}(\mathbf{v}, 0), 2\sigma_y \frac{\partial \mathbf{u}}{\partial \zeta_y}(\mathbf{v}, 0)) \quad (27)$$

where tex2D performs an anisotropic texture fetch using an elliptical filter specified by its center and axes in texture space (like the OpenGL `texture2DGrad` function).

We must finally choose a parameterization $\mathbf{u} = \mathcal{U}(\mathbf{r})$ for the environment map, with $\mathbf{r}$ in world space. Since the sky is hemispherical, a single 2D texture can be used to represent it. $\mathcal{U} = [\theta_r\ \phi_r]^T$ or $[r_x\ r_y]^T$ are possible parameterizations, but they give too much distortion to approximate the transformed filter kernel with an ellipse. We found that the stereographic projection $[r_x\ r_y]^T/(1+r_z)$, which also gives the slopes of the half vector between $\mathbf{r}$ and the vertical, was a good choice to minimize these distortions. In this space the zenith is projected at the origin, and the horizon is mapped to the unit circle (see Figure 8).

5.3. Refracted light

The light coming from the Sun and the sky is also refracted inside the water, and can be refracted again to the viewer via multiple scattering in the water and reflections on the sea floor. In deep water multiple scattering dominates. So we consider here that the radiance L_{sea} reaching the surface from below is diffuse (and proportional to the total Sun and sky irradiance). With this hypothesis, and by replacing F with $T = 1 - F$ in the BRDF, the same computations as in the previous section give

$$I_{sea} \approx L_{sea}(1 - \bar{F}) \quad (28)$$

The complete lighting algorithm is summarized in Figure 9.

Algorithm 5.1: SEACOLOR($\mathbf{v}, \mathbf{l}, \mathbf{n}, \mathbf{t}_x, \mathbf{t}_y, \sigma_x, \sigma_y$)

procedure U($\boldsymbol{\zeta}$)
  $\mathbf{f} \leftarrow$ normalize($[-\zeta_x \;\; -\zeta_y \;\; 1]$) *// tangent space*
  $\mathbf{f} \leftarrow f_x\mathbf{t}_x + f_y\mathbf{t}_y + f_z\mathbf{n}$ *// world space*
  $\mathbf{r} \leftarrow 2(\mathbf{f}\cdot\mathbf{v})\mathbf{f} - \mathbf{v}$
  return $[r_x \;\; r_y]/(1+r_z)$

$\mathbf{h} \leftarrow$ normalize$(\mathbf{v}+\mathbf{l})$
$\boldsymbol{\zeta}_h \leftarrow -[\mathbf{h}\cdot\mathbf{t}_x \;\; \mathbf{h}\cdot\mathbf{t}_y]/\mathbf{h}\cdot\mathbf{n}$
$\cos\theta_v \leftarrow \mathbf{v}\cdot\mathbf{n} \quad \phi_v \leftarrow \text{atan}(\mathbf{v}\cdot\mathbf{t}_y, \mathbf{v}\cdot\mathbf{t}_x)$
$\cos\theta_l \leftarrow \mathbf{l}\cdot\mathbf{n} \quad \phi_l \leftarrow \text{atan}(\mathbf{l}\cdot\mathbf{t}_y, \mathbf{l}\cdot\mathbf{t}_x)$
$\sigma_v \leftarrow (\sigma_x^2\cos^2\phi_v + \sigma_y^2\sin^2\phi_v)^{1/2}$
$\bar{F} \leftarrow R + (1-R)(1-\cos\theta_v)^{5e^{-2.69\sigma_v}}/(1+22.7\sigma_v^{1.5})$
$\mathbf{u}_0 \leftarrow$ U$([0 \;\; 0])$
$\Delta\mathbf{u}_x \leftarrow 2\sigma_x(\text{U}([\varepsilon \;\; 0]) - \mathbf{u}_0)/\varepsilon$
$\Delta\mathbf{u}_y \leftarrow 2\sigma_y(\text{U}([0 \;\; \varepsilon]) - \mathbf{u}_0)/\varepsilon$
$I_{sun} \leftarrow L_{sun}\Omega_{sun}\frac{p(\boldsymbol{\zeta}_h)(R+(1-R)(1-\mathbf{v}\cdot\mathbf{h})^5)}{4(\mathbf{h}\cdot\mathbf{n})^4\cos\theta_v(1+\Lambda(a_v)+\Lambda(a_l))}$
$I_{sky} \leftarrow \bar{F}\,\text{texture2DGrad}(L_{sky}, \mathbf{u}_0, \Delta\mathbf{u}_x, \Delta\mathbf{u}_y)$
$I_{sea} \leftarrow L_{sea}(1-\bar{F})$
return $I_{sun} + I_{sky} + I_{sea}$

Figure 9: ***Summary of our lighting algorithm****. The input unit vectors* $\mathbf{v}$, $\mathbf{l}$, $\mathbf{n}$, $\mathbf{t}_x$ *and* $\mathbf{t}_y$ *are in world space.* σ_x^2 *and* σ_y^2 *are the slope variances along the surface tangents* $\mathbf{t}_x$ *and* $\mathbf{t}_y$*. The derivatives of* U *can be computed analytically. Using finite differences is faster, and precise enough with* $\varepsilon = 10^{-3}$.

6. Extensions

Our algorithms can be extended to account for local waves, local and multiple reflections, and planet-scale rendering.

Local waves. Our ocean model supports other waves than trochoids, provided their wavelength and slope variances are known (see an example with Kelvin wakes in Fig. 11). We can also easily change the waves parameters locally, which modifies the Sun specular reflection and the sky reflection (see ground and space view examples in Fig. 11 and 12).

Local reflections. When the viewer is close to a boat or an island the Sun can be shadowed and each point on the ocean sees a different environment above and under water. Shadows are easily handled with a shadow map. Local reflections cannot be handled with one environment map per point on the ocean. Instead, we use a reflection map in screen space [HVT*06] (local refractions can be handled in the same way). Hu *et al*. [HVT*06] used ad-hoc formulas to jitter and filter this map according to the surface roughness. We improve their method by filtering this map with our fast environment lighting method (see Section 5.2), with a mapping $\mathcal{U}$ from microfacet slopes to reflection map coordinates.

Multiple reflections. In order to account for multiple reflections on waves we use a non null radiance in the sky environment map for directions below the horizon. View rays reflected below the horizon are eventually reflected towards the sky or refracted in the water and therefore contribute to the reflected light. We approximate this contribution as an average Fresnel reflectance times the sky irradiance.

Planet-scale rendering. We can render the ocean at all scales from ground to space as follows. For space views we do not need a projected grid nor transitions between levels of details. Instead, we render a sphere with the Ross BRDF directly. The main problem is to compute the reflected sky light, since each point on the sphere sees different sky conditions. We solve that by ignoring clouds. We can then use a set of 2D environment maps indexed by the Sun zenith angle. Each point is then lit with the 2D map corresponding to the local Sun elevation. For altitudes below $20,000\ m$, we switch to the projected grid method, with a projection on the sphere. This poses parameterization and numerical precision issues that can be fixed (we do not have space to discuss them here), but our method is otherwise unchanged.

7. Implementation and Results

7.1. Implementation

We implemented our method in vertex and fragment shaders on GPU. The vertex shader projects the screen space regular grid, displaces it by evaluating Eq. 2, and projects it back. The fragment shader computes the per pixel normals using Eq. 3, and then the Sun, sky and refracted light as described in Section 5. The wave parameters $\{h_i, \mathbf{k}_i, \omega_i\}$ are generated on CPU and stored in a texture. We generate them either with the Hasselmann [PM64, HDE80] spectrum (see Eq. 1 and Fig. 14 in complementary materials), or with an ad-hoc spectrum $h(\omega) \propto \omega^{-2}$. We get accurate results and smooth transitions with both spectrums.

We use a geometric progression for the wavelengths $\lambda_i = 2\pi/\|\mathbf{k}_i\|$, which allows us to optimize the evaluation of Eqs. 2, 3 and 4. We know in advance the indices i for which the weights w_p, w_n or w_r are not null, and restrict the sums to these indices. We also evaluate Eq. 4 by subtracting from the total variance due to all waves (computed on CPU) the variance of the resolved waves. As a result the computation time in the shaders is not proportional to the total number of waves, but only to the number of resolved waves. It is minimal for distant views where no details can be seen.

According to Nyquist theorem, the N_{min} and N_{max} parameters used in the weights w_p, w_n or w_r (see Section 3) should be larger than 2 to avoid aliasing. In practice this gives too much blur, so we use in fact $N_{min} = 1.0$ and $N_{max} = 2.5$.

7.2. Validation

We validated our real-time method by comparing its results with reference images. We computed these reference images

with a very detailed geometric model, using a perfectly specular BRDF. To integrate the subpixel details we divided the view frustum in fixed size areas in object space, and rendered each area with OpenGL, in a constant size buffer. We used about 6000 30×30 m^2 areas rendered with $2 \times 600 \times 600$ triangles in 1024×1024 pixels buffers (at least). We also used several normal samples per pixel – up to 512 – to correctly sample the Sun, which occupies only $1/8000^{th}$ of the hemisphere. We then resized and reassembled all these buffers to get the final images.

We compared the reflected Sun and sky radiance and the refracted radiance separately. For each we compared the energy repartition inside the images. The results are presented in Figure 10 (right column). The largest errors come from the reflected sky radiance, for agitated seas (due to our approximation method). They are mostly due to the approximation made in Eqs. 21 and 22 (the approximation of Eq. 26 is very accurate, and the one in Eq. 27 is also quite good). These errors are low frequency and should not be noticeable without reference images.

7.3. Results

Some results obtained with our method are depicted in Figures 1, 4 (right), 6, 11 and 12, showing various Sun and sky conditions, ground and space views, Kelvin wakes, local reflections, locally varying wave parameters, etc (some images show structured patterns, due to an insufficient sampling of the surface spectrum). We found that the hardware anisotropic filter `texture2DGrad` sometimes caused artifacts near the horizon (slightly visible in Figure 11). Using an average of nine `texture2D` samples inside the elliptical filter support solves the problem but adds about 2.5 *ms* per 1024×768 frame. All results and performance measures were obtained with `texture2DGrad`.

With a NVIDIA 8800 GTS and a 1024×768 resolution, we get 52 fps with 60 trochoids from 2 *cm* to 30 *m*, with a horizontal view at 4 *m* above the surface (with 8×8 pixels cells for the projected grid). This gives 19.2 *ms* per frame, including 11.1 *ms* to compute $\mathbf{p}$, $\mathbf{n}$ and σ_x, σ_y, and 8.1 *ms* for Algorithm 5.1. We get 87 fps at 1000 *m* and 130 fps at 8000 *m*, showing that the computation time of $\mathbf{p}$, $\mathbf{n}$ and σ_x, σ_y is proportional to the number of resolved waves.

8. Conclusion

We presented a real-time method to compute realistic and accurate ocean lighting at all scales, from very short to very long distances. Our method uses a hierarchy of models from geometry to BRDF, without aliasing nor visible transitions between models. In future work we would like to investigate the case of shallow water and coasts, an important feature for planet-size rendering. We also plan to evaluate our fast approximate environment lighting method in other situations. Finally, we would like to generalize our approach based on a model hierarchy to other contexts.

The source code of our implementation, the accompanying video and some supplemental materials are available at `http://evasion.inrialpes.fr/~Eric.Bruneton/`.

Acknowledgments. We thank Vincent Ross for the discussions we had about their BRDF model.

References

[AS00] ASHIKHMIN M., SHIRLEY P.: An anisotropic Phong BRDF model. *Journal of Graphics Tools 5* (2000), 25–32.

[BM93] BECKER B. G., MAX N. L.: Smooth transitions between bump rendering algorithms. *SIGGRAPH 93 27* (1993), 183–190.

[CC06] CHIU Y.-F., CHANG C.-F.: GPU-based ocean rendering. *IEEE International Conference on Multimedia and Expo* (2006), 2125–2128.

[CFB*07] CAILLAULT K., FAUQUEUX S., BOURLIER C., SIMONEAU P., LABARRE L.: Multiresolution optical properties of rough sea surface in infrared. In *Society of Photo-Optical Instrumentation Engineers Conference Series* (2007), vol. 6743.

[CM54] COX C., MUNK W.: Measurement of the roughness of the sea surface from photographs of the Sun's glitter. *Journal of Optical Society of America 44* (1954), 838–850.

[CT81] COOK R. L., TORRANCE K. E.: A reflectance model for computer graphics. *SIGGRAPH 81 15*, 3 (1981), 307–316.

[FR86] FOURNIER A., REEVES W. T.: A simple model of ocean waves. *SIGGRAPH 86 20*, 4 (1986), 75–84.

[HDE80] HASSELMANN D. E., DUNCKEL M., EWING J. A.: Directional wave spectra observed during JONSWAP 1973. *J. Phys. Oceanogr. 10* (1980), 1264–1280.

[HNC02] HINSINGER D., NEYRET F., CANI M.-P.: Interactive animation of ocean waves. In *Symposium on Computer Animation* (2002), pp. 161–166.

[HSRG07] HAN C., SUN B., RAMAMOORTHI R., GRINSPUN E.: Frequency domain normal map filtering. *SIGGRAPH 07 26*, 3 (2007), 28.

[HTSG91] HE X. D., TORRANCE K. E., SILLION F. X., GREENBERG D. P.: A comprehensive physical model for light reflection. *SIGGRAPH 91 25*, 4 (1991), 175–186.

[HVT*06] HU Y., VELHO L., TONG X., GUO B., SHUM H.: Realistic, real-time rendering of ocean waves. *Computer Animation and Virtual Worlds 17*, 1 (2006), 59–67.

[Kaj85] KAJIYA J. T.: Anisotropic reflection models. *SIGGRAPH 85 19*, 3 (1985), 15–21.

[KVHS00] KAUTZ J., VÁZQUEZ P.-P., HEIDRICH W., SEIDEL H.-P.: Unified approach to prefiltered environment maps. In *Rendering Techniques* (2000), pp. 185–196.

[PA00] PREMOZE S., ASHIKHMIN M.: Rendering natural waters. In *Pacific Graphics 00* (2000), pp. 189–200.

[PM64] PIERSON JR. W. J., MOSKOWITZ L.: A proposed spectral form for fully developed wind seas based on the similarity theory of S. A. Kitaigorodskii. *Journal of Geophysical Research 69* (Dec. 1964), 5181–5190.

[RD07] ROSS V., DION D.: Sea surface slope statistics derived from Sun glint radiance measurements and their apparent dependence on sensor elevation. *Journal of Geophysical Research 112*, 11 (Sept. 2007).

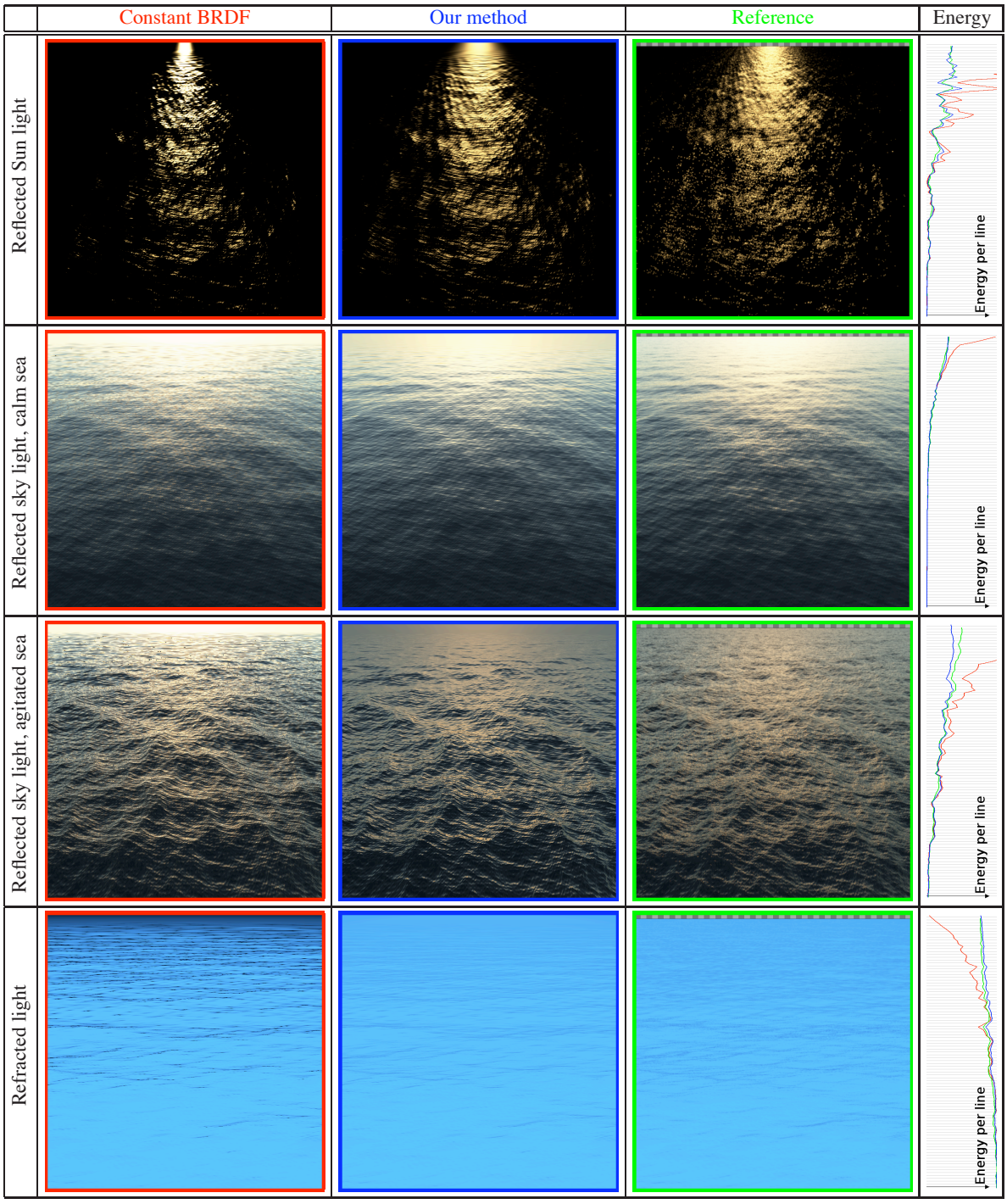

Figure 10: ***Validation***. *Comparison of the reflected Sun light, the reflected sky light and the refracted light obtained with a constant BRDF and with our method, against reference images. The total radiance per image line is shown on the right (each row has its own scale). Both models use averaged positions and normals with the Ross BRDF. However, the basic model uses a constant BRDF, while our model adapts its parameters to the subpixel surface details. All methods give the same result in the foreground, but the basic method rapidly diverges with the view distance. On the contrary our method gives results very similar to the reference images. Note how the reflected sky light changes from a calm sea to an agitated sea – the sky is the same in both cases. Due to our approximations, we get less accurate but still very visually convincing results for agitated seas.*

Figure 11: ***Results***. *Comparison between photos (*top*) and our results (*bottom*). From left to right: locally modified Sun and sky reflections due to a calm area, and three different Sun and sky conditions.*

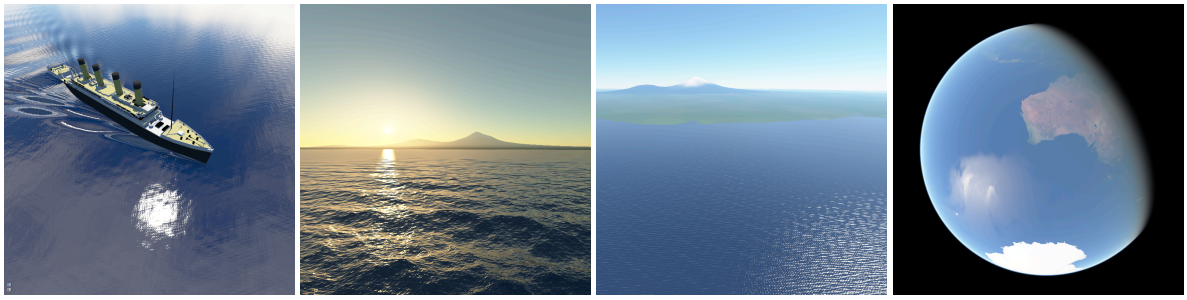

Figure 12: ***Results***. *From left to right: Kelvin wakes, and three more and more distant views in a planet renderer. On the right, locally varying sea conditions give a non uniform Sun glint. See also the accompanying video.*

[RDP05] ROSS V., DION D., POTVIN G.: Detailed analytical approach to the gaussian surface bidirectional reflectance distribution function specular component applied to the sea surface. *Journal of Optical Society of America A 22* (Nov. 2005), 2442–2453.

[Sch94] SCHLICK C.: An inexpensive BRDF model for physically-based rendering. *Computer Graphics Forum 13* (1994), 233–246.

[Smi67] SMITH B.: Geometrical shadowing of a random rough surface. *IEEE Transactions on Antennas and Propagation 15* (Sept. 1967), 668–671.

[Tes01] TESSENDORF J.: Simulating ocean water. In *ACM SIGGRAPH course notes* (2001).

[TLQ*05] TAN P., LIN S., QUAN L., GUO B., SHUM H.-Y.: Multiresolution reflectance filtering. In *Rendering Techniques* (2005), pp. 111–116.

[TLQ*08] TAN P., LIN S., QUAN L., GUO B., SHUM H.: Filtering and rendering of resolution-dependent reflectance models. *IEEE Transactions on Visualization and Computer Graphics 14*, 2 (2008), 412–425.

[War92] WARD G. J.: Measuring and modeling anisotropic reflection. *SIGGRAPH 92 26*, 2 (1992), 265–272.

[YPZL05] YANG X., PI X., ZENG L., LI S.: GPU-based real-time simulation and rendering of unbounded ocean surface. In *International Conference on Computer Aided Design and Computer Graphics* (2005), pp. 428–433.

Appendix A: Trochoid statistics

For a trochoid defined parametrically by $x(s) = s - h\sin(ks)$ and $z(s) = h\cos(ks)$, with $k = 2\pi/\lambda$, the slope is given by:

$$\frac{dz}{dx}(x) = \frac{dz}{ds}(s)\frac{ds}{dx}(x) = \frac{dz}{ds}(s)\left(\frac{dx}{ds}(s)\right)^{-1}$$

Its mean μ_s is 0, and its variance is:

$$\begin{aligned}\sigma_s^2 &= \frac{1}{\lambda}\int_0^\lambda \left(\frac{dz}{dx}(x)\right)^2 dx = \frac{1}{\lambda}\int_0^\lambda \left(\frac{dz}{dx}(s)\right)^2 \frac{dx}{ds}(s)ds \\ &= \frac{1}{\lambda}\int_0^\lambda \left(\frac{dz}{ds}(s)\right)^2 \left(\frac{dx}{ds}(s)\right)^{-1} ds \\ &= \frac{1}{\lambda}\int_0^\lambda \frac{k^2h^2\sin^2(ks)}{1-kh\cos(ks)}ds = 1-\sqrt{1-k^2h^2}\end{aligned}$$