Ocean waves synthesis using a spectrum-based turbulence function

Sébastien Thon, Jean-Michel Dischler, Djamchid Ghazanfarpour
Laboratory MSI
University of Limoges
ENSIL Technopole, 87068 Limoges, France
thon@ensil.unilim.fr, dischler@ensil.unilim.fr, ghazanfa@ensil.unilim.fr

Abstract

The representation of ocean waves is not a resolved problem in Computer Graphics yet. There is still no existing method that allows one to simply describe an agitated surface of any size that is visually sufficiently realistic, without using entirely physical models that are usually very complex. Here we present a simple method to represent and animate an ocean surface in deep water by considering it as a procedural texture. This texture is defined by a combination of two levels of details. The first one is a superposition of 2D trochoids whose parameters are determined by ocean waves characteristics in frequency domain. In order to increase the visual complexity of this model and to reduce computation, we incorporate a 3D turbulence function to provide a second level of detail. This turbulence function is also determined by frequency characteristics of ocean waves. Since our synthesized ocean waves spectrum approaches a real ocean waves spectrum, we obtain realistic water waves in the spatial domain. The animation of our model is performed by shifting the phase of the trochoids and by moving into the 3D turbulence function. Since our definition is procedural and continuous, it permits to obtain any size of water surface with any level of detail as well as a simple, direct, antialiasing method. Our model can be used to generate ocean waves using 2D textures or bump maps as well as 3D textures.

1. Introduction

The representation of more or less agitated water surfaces is not yet a resolved problem in Computer Graphics. In this paper, we propose to represent non-breaking agitated ocean waves in deep water without taking into account wave refraction and reflection.

Ocean waves are propagating disturbances of the water surface, principally generated by the wind. There are basically two kinds of approaches for modeling and animating the waves: empirical approaches which describe explicitly the wave appearance, and physical models generally based on the use of fluid mechanic equations.

Many empirical approaches describe explicitly the appearance of water waves by use of parametric functions. The simplest model is Airy's (1845), which supposes that a wave has a sinusoidal shape. Max [12] used this model to represent low amplitude ocean waves by superposing many sinusoidal functions. Peachey [14] improves this method by changing the waves shapes according to the ocean floor. More complex parametric equations have been used by Fourier and Reves [3] and Gonzato and Le Saec [6], based on the water wave model of Gerstner established in 1802. This model describes the motion of each water particle as a circle around a fixed point. The resulting profile of the water wave is a trochoid with more or less sharpened crests. The animation of all of these empirical models based on sinusoidal equations is done by changing the phase according to the time. This phase modification entails a displacement of the water wave.

These empirical models are well designed for modeling propagating water wave fronts, but they cannot easily represent an agitated water surface because too many functions would be necessary. In order to represent such a surface, some methods use an empirical noise function [15] [17]. Perlin [15] used the noise for placing the centers of concentric water waves, and for perturbing the surface by modification of the normals. However, it is difficult to determine the noise function parameters. Linear combinations of noises are used to directly represent the water waves [17], but the results are uncontrollable and a realistic water motion is impossible to achieve. In order to avoid the difficulties raised by these empirical methods for a realistic water modelling, we can use physical equations of the fluid mechanics, such as the Navier-Stokes equations (1827). These equations are used in scientific simulations and solved by using numerical models necessitating a huge finite-element grid [10] [9].

These equations are well adapted for scientific simulations needing a high degree of precision (often not necessary for a visual realism), but they are computationally very expensive. Therefore, several simplifications of these equations have been proposed in computer graphics [8] [1] to reduce the computation time. In spite of a realistic modeling of water behavior with these simplified physical equations, a designer has no control on the desired result. In fact, the designer can only give initial conditions, and the system evolves automatically according to the equations. Another physical approach is a spectral-based one. Mastin *et al* [11] used the Pierson-Moskowitz filter [16], obtained from a real ocean waves spectrum, for filtering a white noise image in the frequency domain. By its inverse fast Fourier transformation (inverse FFT) they obtained a finite dimensions image in the spatial domain that they directly used as a height field of water waves. This representation of the water as a finite set of heights is the main drawback of this method, because the dimensions of the resulting water surface are limited and it is impossible to obtain any level of details. If we want a higher level of detail, the filtering of a higher resolution white noise picture will be needed. Moreover, this filter cannot create realistic ocean waves with trochoidal shapes.

We propose a model between these two families of methods that describes in a very compact and simple formulation the shape and animation of water waves. Although it is based on an explicit description of the wave shape and animation, our model is partially a physical one because, as [11], it uses parameters from the Pierson-Moskowitz filter [16] which describes a real ocean waves spectrum. Our model is used as a procedural texture to compute continuously the height of the water at any point of an infinite surface. This simple parametric model allows the representation of the main structure of an ocean surface by superposing 2D trochoids. The parameters of these functions are obtained in the frequency domain. We build a spectrum that has the same characteristics as a real ocean waves spectrum, and we select a finite number of the most representative (highest amplitudes) frequency components. A similar method of components selection in the frequency spectrum has already been used by the authors [4] to synthesize procedural textures. Trochoid frequencies and amplitudes in our method are determined according to those of the frequency components. We show how the superposition of a 3D turbulence function to this reduced set of 2D trochoids permits us to better approximate the previously sampled real ocean spectrum, allowing a more realistic surface. The parameters of this turbulence function are not determined empirically as it is the case of [15] and [17], but are again determined according to the ocean waves spectrum. The splitting of the model into a main structure (trochoids) and a detail level (turbulence) permits good control over the agitation state of the water. Moreover, our model produces a more realistic and computationally efficient representation than a simple superposition of many trochoids. In addition, it is more simple and compact than a complete physical model. It can be easily animated by shifting the phase of the trochoids to displace the main structure of water waves and by moving into the 3D turbulence function to animate the little water waves that it represents. We propose three methods for rendering the generated ocean waves: a 2D texture, a bump map, and a 3D texture considered as an implicit surface. We will also show how the procedural definition of our texture allows using a simple direct antialiasing technique.

In the next sections, we first present the model itself, the base model and its spectral generation, and the addition of a controlled turbulence. Then, we show how to animate this model and finally, we present three different rendering techniques for our ocean waves model.

2. Ocean waves

2.1. The base model (coarse level)

The convention used is that our world is orientated such that the xy plane is the plane of the water surface, with the z axis pointing upward. For the modeling of ocean waves, we propose a procedural texture model, defined by the superposition of 2D functions. For these functions, we could use simple sinusoidal waves, since Airy (1845) found out that the profile of a swell wave has a sinusoidal shape:

$$h\left(x,t\right)=A.\cos\left(k.x-\omega.t\right) \qquad (1)$$

However, this model only apply to relatively small waves. It is suitable for a quiet water surface, but will not be realistic for an ocean. In fact, larger waves tend to have sharp crests and rounded troughs, a type of curve known as a trochoid, as described by Gerstner (1804). So, we use this model, which describes the motion of each water particle in deep water as a circle of radius r in a vertical plane around a fixed point (x_0, z_0):

$$\begin{aligned} x &= x_0 + r.\sin\left(k.x_0-\omega.t\right) \\ z &= z_0 - r.\cos\left(k.x_0-\omega.t\right) \end{aligned} \qquad (2)$$

The water wave profile obtained is a trochoid, allowing for more or less sharpened crested waves according to the product $k.r$ (figure 1). This kind of shape is more realistic in the case of an agitated ocean surface. We pre-compute a look-up table of trochoid heights by using equation (1) for a given number of x points over a trochoid period. We build a function v that takes into parameter a x value and that returns the corresponding z height of the trochoid by using the look-up table. In order to replace the sinusoidal shape of water waves by a trochoid shape, we replace the cosine function of equation (1) by the function v. We extend this 1D trochoid shape equation to 2D. The water height z on

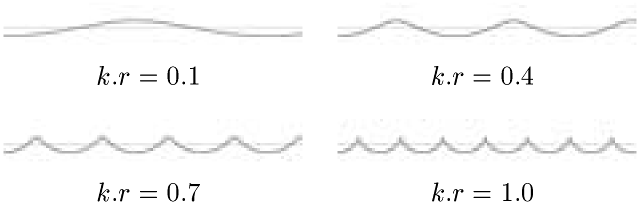


Figure 1. Shape of a trochoid according to the product $k.r$

any point (x, y) of a surface, according to the time, is given by:

$$h(x, y, t) = A.v\left(k\left(x.\cos\theta + y.\sin\theta\right) - \omega.t + \varphi\right) \quad (3)$$

where A is the amplitude, k is the wave number ($k = 2\pi/\lambda$, λ is the wavelength), ω is the pulsation ($\omega = 2\pi.f$, f is the frequency), θ is the direction angle of the wave front in the xy plane, and φ its phase. The surface of an ocean is made up of many of these waves fronts moving in a range of directions centered around the wind main direction. The waves interfere with one another, creating complex interference wave patterns. The superposition of n of these 2D trochoids gives a parametric surface defined by:

$$h(x, y, t) = \sum_{i=1}^{n} A_i.v\left(k_i\left(x.\cos\theta_i + y.\sin\theta_i\right) - \omega_i.t + \varphi_i\right) \quad (4)$$

However, this trochoid wave model is not true for an highly agitated ocean surface, because it can not represent breaking waves.

2.2. Spectral generation

Now, the crucial problem is to determine what amplitude, frequency, phase and direction values must be used for the trochoids of our model. We propose to obtain these values by using an ocean spectrum, given by the Pierson-Moskowitz filter.

2.2.1 The Pierson-Moskowitz filter

Pierson and Moskowitz proposed in 1964 [16] a filter describing the profile of an ocean spectrum (figure 2), obtained from ship-recorded measurements. This model can only be used for fully developed seas[1] in deep water, because it cannot take into account effects such as water wave refraction or reflection that happen in shallow water. This filter is defined as:

$$F_{PM}(f) = \frac{a.g^2}{(2\pi)^4 f^5} e^{-\frac{5}{4}\left(\frac{f_m}{f}\right)^4} \quad (5)$$

where f is the frequency, a is the Phillips constant ($a = 0.0081$), g is the gravity constant, and f_m is a peak of frequency that depends on the wind speed U_{10} at a height of 10 meters above the sea surface according to :

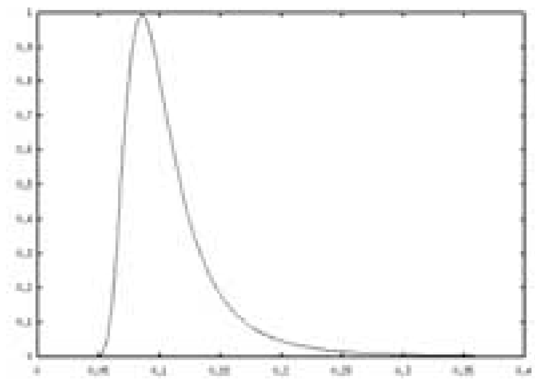


Figure 2. Pierson-Moskowitz 1D filter for a wind speed $U_{10} = 15\ m/s$

$$f_m = \frac{0.13g}{U_{10}} \quad (6)$$

Hasselman *et al* [7] proposed to extend this 1D filter to a 2D one:

$$F(f, \alpha) = F_{PM}(f).D(f, \alpha) \quad (7)$$

where $D(f, \alpha)$ is a directional factor that weights the filter according to an angle α with respect to the direction of the wind θ:

$$D(f, \alpha) = N_p^{-1} \cos^{2p}\left(\frac{\alpha}{2}\right) \quad (8)$$

with:

$$p = 9.77\left(\frac{f}{f_m}\right)^{\mu}, \quad \mu = \begin{cases} 4.06 & \text{if } f < f_m \\ -2.34 & \text{if } f > f_m \end{cases},$$

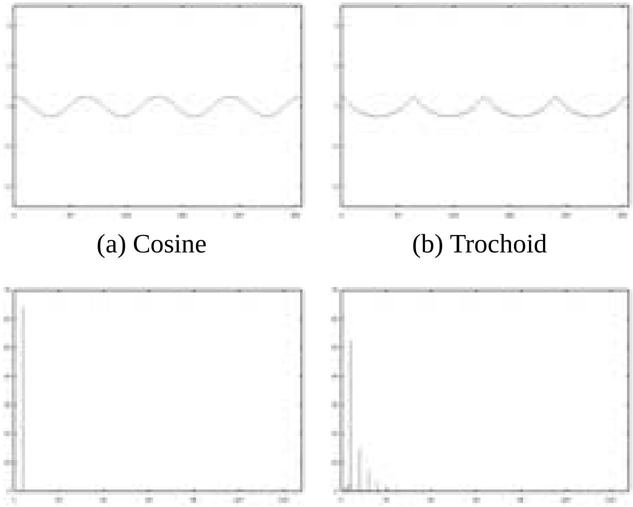
$$N_p = \frac{2^{1-2p}\pi.\Gamma(2p+1)}{\Gamma^2(p+1)}$$

(Γ is Euler's Gamma function)

2.2.2 Using the filter

Mastin *et al* [11] used the Pierson-Moskowitz filter to compute a height field, by filtering a white noise image in the

[1] A fully developed sea is a state in the formation of the waves by the wind where, given a constant wind speed, a distance over which the wind blows in the same direction, and duration, the spectrum will no longer increase.



(a) Cosine (b) Trochoid

(c) Cosine (a) spectrum (d) Trochoid (b) spectrum

Figure 3. Comparison between a cosine (a) and a trochoid (b), and their respective spectrum (c) and (d)

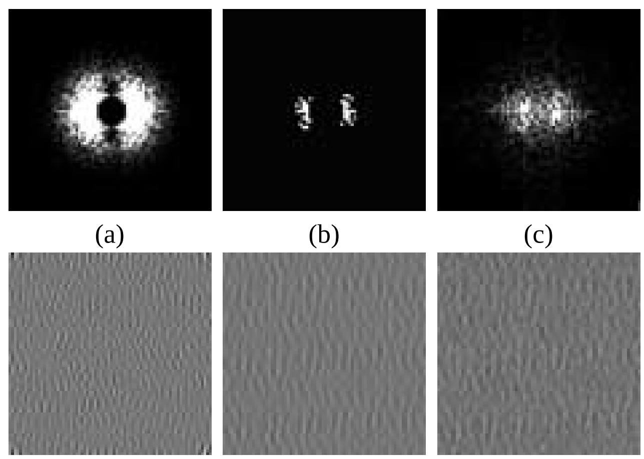
frequency domain, then by using an inverse FFT for returning into the spatial domain. This procedure produces only a finite set of water heights. Instead, our model is continuous because it remains in the frequency domain. We sample the ocean spectrum in a small matrix (we use a size of 64x64) to have a finite set of frequency components (figure 4a). In this set, we select some of the most representative components in terms of amplitude (figure 4b), and we use the values of these components (frequencies, amplitudes, directions) for the trochoids of our model. We can use these values for trochoids instead of cosines, because these two kinds of functions have similar spectrum, a trochoid main frequency corresponding to the cosine frequency (figure 3).

Unlike [11], the water waves are reconstituted by directly superposing the 2D functions of the set (figure 4e), without using an inverse FFT. Since each of these functions is continuous, we obtain all levels of detail. Moreover, we obtain realistic water wave shapes according to Gerstner's theory, because we directly superpose trochoids. It is impossible to achieve such shapes with the method used by [11], because the Pierson-Moskowitz filter does not allow the reconstitution of trochoids. The main advantage of our model is that it is entirely procedural, we compute water heights with our texture function when it is needed, without storing any height-field like [11] or huge finite-element grid like physical-based models.
However, the selection of only a few number of frequency components in the spectrum, results in a lack of precision compared to an image (figure 4d) obtained by an inverse FFT which takes into account all frequencies. By selecting the components having the highest amplitudes, we can reproduce the main structure of the water waves, but it does not take into account the little variations that perturb this structure and that give it a less regular aspect, in particular a more agitated aspect. A solution would consist in selecting a greater number of components in the spectrum, but the model would be more computationally expensive because a summation of more trochoids will be necessary. The alternative that we propose consists in replacing all these neglected components by an appropriate turbulence function.

2.3. Adding a controlled turbulence function

Many natural phenomena can be described as a combination of well-defined structures and little random variations. This is also true for ocean waves. We propose to enhance the basic waves model describing the "main structure" by adding a controlled 3D turbulence function. Thus, we introduce economically the neglected energy, obtaining a more realistic surface. We use Perlin's turbulence function [15], defined as the sum of noise functions with different "amplitudes" over a range of different "frequencies":

$$Turbulence\,(P,m) = \sum_{i=0}^{n_o-1} \frac{Noise\,(P.m^i)}{m^i} \qquad (9)$$



(a) (b) (c)

(d) (e) (f)

Figure 4. The use of a turbulence function allows the ocean waves spectrum reconstitution

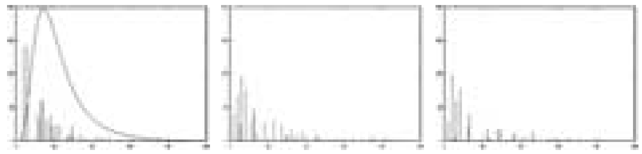
where n_o is the number of frequency levels to sum up, P is a point of $\mathbb{R}^3$ (in our implementation) and m is the frequency multiplier. This turbulence function has already been used to represent water [15], but empirically, because the determination of the parameters for this function is not easy. In our model, we propose to use this turbulence function and to set its parameters so that the spectrum of this turbulence approaches a real ocean waves spectrum. Thus, by adding this turbulence to the heights obtained by superposing the trochoids (figure 4e), we obtain a wavy surface (figure 4f) whose spectrum (figure 4c) approximates a real ocean waves spectrum (figure 4a) given by the Pierson-Moskowitz filter.

Consequently, this wavy surface given by our model is similar to the one (figure 4d) given by a direct FFT of a real ocean waves spectrum. However, by adding this turbulence function in the spatial domain, we introduce low frequencies that are not present in a real ocean waves spectrum. Indeed, frequencies lower than the peak frequency fm (equation 6) are attenuated or deleted by the Pierson-Moskowitz filter in a "real" spectrum (figure 4a). We cannot avoid these low frequencies, but their influence is negligible compared to the amplitudes of the trochoids spectrum frequency components.

The final formulation of our ocean waves model is:

$$h(x, y, t) = \left[\sum_{i=1}^{n} A_i.v\left(k_i\left(x.\cos\theta_i + y.\sin\theta_i\right) - \omega_i.t + \varphi_i\right)\right] + A_t.Turbulence\left(x_t(t), y_t(t), z_t(t), m\right) \qquad (10)$$

where A_t is the "amplitude" of the turbulence, m is the frequency multiplier, and $x_t(t)$, $y_t(t)$, $z_t(t)$ are time functions for displacement into the 3D turbulence. In the first part, in order to build the water waves main structure by trochoids superposition, we selected a few frequency components in a "real" spectrum by choosing the most representatives components, e.g. with the highest amplitudes. Consequently, these frequency components have been chosen in the neighborhood of the frequency peak f_m. Now, in a second step, with the turbulence function, we try to approach the "real" spectrum part where frequencies are greater than the highest frequency f_{max} selected in the first step. To do so, we compare the 1D turbulence spectrum to the 1D Pierson-Moskowitz filter. We only take into account frequencies higher than f_{max}. The amplitude A_t and frequency multiplier m values of the turbulence are determined so that these two spectrum parts are similar. To compare these two parts, we compare their amplitude and slope values in f_{max}. By modifying the frequency multiplier m in equation (9), we can change the turbulence spectrum slope (figure 5). The frequency multiplier is chosen so that the 1D turbulence spectrum slope in f_{max} and the 1D Pierson-Moskowitz filter slope in f_{max} are similar (figure 5a). The right value for the frequency multiplier is determined by a relaxation method, by varying its value until the slopes are similar. A similar relaxation technique in the frequency domain has been used in [2]. Then, in order to scale the turbulence spectrum, the turbulence amplitude value A_t is chosen so that the turbulence spectrum and Pierson-Moskowitz filter amplitudes in f_{max} are the same.



(a) $m = 1.5$ (b) $m = 2.5$ (c) $m = 5.0$

Figure 5. 1D turbulence spectrum with different frequency multiplier m values.

2.4. Antialiasing

Ray traced images are prone to aliasing because ray tracing is a point sampling method. Aliasing problems (in the form of moir patterns on textures) mainly occur in areas of image in which the compression rates due to the perspective projection on the screen are high [5] (for example, near the horizon in figure 6a).

The compression induces a frequency increase of the texture. Consequently, highly compressed areas of the texture can represent frequencies higher than the famous Nyquist limit. Since our model is based on the superposition of trochoidal functions, we can deduce a simple and direct method for antialiasing: when we compute the water height on a point of the screen, instead of superposing all of the functions, we can neglect or progressively reduce the amplitudes of the trochoids that have frequencies too high to be represented on this point (figure6). This can be compared to the application of a low pass filter to the texture. This technique has two advantages: firstly, it permits to reduce efficiently the aliasing in a simple and direct way, and secondly, it reduces the computation time, because fewer trochoids are taken into account for the summation. This technique is close to the "clamping" method [13] for 2D textures, and this principle has also been used by [15] for the turbulence, as well as [4] for 3D textures.

3. Animation

Animating ocean waves is very easy with our model. We consider separately the animation of the water waves main structure, constituted by the trochoids, and the animation of small-scale perturbations introduced by the turbulence function. The trochoids animation is simply done by modifying their phases with respect to time, entailing the propagation of the water waves. Small water waves represented by the turbulence are chaotic, their motion is not distinct.

(a)

(b)

Figure 6. Image without (a) and with antialiasing (b)

Thus, their animation is achieved by a simple displacement into the 3D turbulence function. In the xy plane, the direction of this displacement is given by the angle θ of wind direction. We additionally apply a linear displacement along the z axis in order to change the turbulence appearance, giving the impression that all small waves are oscillating. The displacement speed in the turbulence is low, because the turbulence represents high frequencies waves, and the propagation speed of a water wave is inversely proportional to its frequency.

4. Results

The water surface described by our model can easily be rendered. It can be used as a 2D texture, a bump map, or as a 3D texture (figure 7). In the case of a 3D texture, a very fast way consists of sampling our continuous model to obtain a finite height map, rendered with classical methods, as a set of triangles with OpenGL for example. The inconvenience of this finite height field method is that it needs a very high sampling for taking into account the small details such as the small variations due to the turbulence. Thus, high memory storage capacities are required. The computation time will also be increased. Another way to use our model as a 3D texture is to process it as an implicit surface, using its procedural and continuous definition. Hence, any level of details is available, and there is no need for huge memory storage. In fact, only a few kilobytes are needed, to store the precomputed trochoid tables and the mathematical formulation of our procedural texture. To render this implicit surface, we have implemented our model in a raytracer, using a raymarching technique to determine the ray-implicit surface intersection. Figures 6, 7 and 8 have been rendered by using this technique.

5. Conclusions

We have presented a simple ocean waves model based on two levels of detail. The first one is the water waves main structure, consisting of a 2D trochoids superposition. The major innovation of our model is that the trochoids parameters values are determined by ocean waves characteristics in frequency domain, and that a spectrum controlled 3D turbulence function is used as a second level of detail for a better approximation of a real ocean waves spectrum. The animation of this model is very easy, because it is directly described in its very compact formulation, by shifting trochoids phases and by moving into the 3D turbulence function. Any level of detail and any size of water surface can be achieved thanks to the continuous definition of our model. The frequency-based definition of our model allows a simple, direct and efficient antialiasing method. It can be used as a procedural continuous texture: as a 2D texture, as a bump map or as a 3D texture rendered as an implicit surface.

6. Future works

We plan on working on both modeling and rendering problems in the future. To improve our model, a better turbulence function is needed for a better approximation of a real ocean waves spectrum. The formulation of the water waves shapes needs to be modified in order to take into account interactions between the water and other objects of the scene, such as waves refraction on the ocean floor or reflection on the shore. The use of turbulence functions for

(a) (b) (c)

Figure 7. Three rendering methods for our ocean waves model: 2D texture *(a)*, **bump mapping** *(b)* **and 3D texture using an implicit surface** *(c)*.

the representation of more agitated water surfaces such as rivers or torrents will be studied. Aside from these modeling issues, there are still water rendering problems because a correct interaction model between light and ocean water needs to be defined.

7. Acknowledgements

We would like to thank the Regional Council of Limoges for their financial support of this work.

References

[1] J. Chen and N. Lobo. Toward interactive-rate simulation of fluids with moving obstacles using navier-stokes equations. *Graphical Models and Image Processing*, pages 107–116, 1995.

[2] J. M. Dischler and D. Ghazanfarpour. A procedural description of geometric textures by spectral and spatial analysis of profiles. *Computer Graphics Forum*, 16(3):129–139, 1997.

[3] A. Fournier and W. Reeves. A simple model of ocean waves. *Computer Graphics*, 20(3):75–84, 1986.

[4] D. Ghazanfarpour and J. M. Dischler. Spectral analysis for automatic 3d texture generation. *Computers and Graphics*, 19(3):413–422, 1995.

[5] D. Ghazanfarpour and B. Peroche. A high quality fitering using forward texture mapping. *Computers and Graphics*, 15(4):569–577, 1990.

[6] J. C. Gonzato and B. le Saec. A phenomenological model of coastal scenes based on physical considerations. *Computer Animation and Simulation*, pages 137–148, 1997.

[7] D. Hasselmann, M. Dunckel, and J. Ewing. Directional wave spectra observed during jonswap 1973. *Journal of Physical Oceanography*, pages 1264–1280, 1980.

[8] M. Kass and G. Miller. Rapid, stable fluid dynamics for computer graphics. *Computer Graphics*, 24(4):49–57, 1990.

[9] G. J. Komen, L. Cavaleri, M. Donelan, K. Hasselmann, S. Hasselmann, and P. A. E. M. Janssen. *Dynamics and modelling of ocean waves*. Cambridge University Press, 1996.

[10] Z. Kowalik and T. S. Murty. *Numerical modelling of ocean dynamics*. World Scientific, 1993.

[11] G. A. Mastin, P. A. Watterberg, and J. F. Mareda. Fourier synthesis of ocean scenes. *Computer Graphics and Applications*, 7(3):16–23, 1987.

[12] N. Max. Vectorized procedural models for natural terrain: waves and islands in the sunset. *Computer Graphics*, 15(3):317–324, 1981.

[13] A. Norton, A. Rockwood, and P. Skolmoski. Clamping: a method of antialiasing textured surfaces by bandwidth limiting in object space. *Computer Graphics*, 16(3):1–8, 1982.

[14] D. R. Peachey. Modeling waves and surf. *Computer Graphics*, 20(3):65–74, 1986.

[15] K. Perlin. An image synthesizer. *Computer Graphics*, 19(3):287–296, 1985.

[16] W. J. Pierson and L. Moskowitz. A proposed spectral form for fully developped wind seas based on the similarity theory of s.a. kilaigorodskii. *Journal of Geophysical Research*, pages 5181–5190, 1964.

[17] S. Worley. A cellular texture basis function. *Computer Graphics*, pages 291–294, 1996.

(a) $U_{10} = 0\ m/s$

(b) $U_{10} = 4\ m/s$

(c) $U_{10} = 6\ m/s$

(d) $U_{10} = 8\ m/s$

Figure 8. Our model allows an easy control over the agitation state of the water surface (implicit surface), by simply varying the wind speed.